Gwent

2.1.1

Generated by Doxygen 1.9.8

1 Gwent - A Calculator 1

1.1 Overview . . . 1

1.2 Printing analog cards . . . 1

1.3 Documentation . . . 1

2 Hierarchical Index 3

2.1 Class Hierarchy . . . 3

3 Class Index 5

3.1 Class List . . . 5

4 File Index 9

4.1 File List . . . 9

5 Class Documentation 13

5.1 Ability Enum Reference . . . 13

5.1.1 Detailed Description . . . 13

5.1.2 Member Data Documentation . . . 13

5.1.2.1 BINDING . . . 13

5.1.2.2 HORN . . . 14

5.1.2.3 MORAL_BOOST . . . 14

5.1.2.4 NONE . . . 14

5.1.2.5 REVENGE . . . 14

5.2 AddCardDialog Class Reference . . . 14

5.2.1 Detailed Description . . . 16

5.2.2 Constructor & Destructor Documentation . . . 16

5.2.2.1 AddCardDialog() [1/2] . . . 16

5.2.2.2 AddCardDialog() [2/2] . . . 16

5.2.3 Member Function Documentation . . . 16

5.2.3.1 onCreate() . . . 16

5.2.4 Member Data Documentation . . . 17

5.2.4.1 disposables . . . 17

5.2.4.2 pickerAdapter . . . 17

5.2.4.3 row . . . 17

5.2.4.4 soundManager . . . 17

5.3 AppDatabase Class Reference . . . 18

5.3.1 Detailed Description . . . 18

5.3.2 Member Function Documentation . . . 19

5.3.2.1 rows() . . . 19

5.3.2.2 units() . . . 19

5.4 BondDamageCalculatorDecorator Class Reference . . . 19

5.4.1 Detailed Description . . . 20

5.4.2 Constructor & Destructor Documentation . . . 21

5.4.2.1 BondDamageCalculatorDecorator() . . . 21

5.4.3 Member Function Documentation 21
5.4.3.1 calculateDamage() 21
5.4.3.2 isBuffed() 22
5.4.4 Member Data Documentation 22
5.4.4.1 idToSquadSize 22
5.5 BurnDialogUseCase Class Reference 22
5.5.1 Detailed Description 23
5.5.2 Member Function Documentation 23
5.5.2.1 burn() [1/2] 23
5.5.2.2 burn() [2/2] 23
5.5.2.3 getBurnUnits() 24
5.6 ResetAlertDialogBuilderAdapter.Callback Interface Reference 24
5.6.1 Detailed Description 25
5.6.2 Member Function Documentation 25
5.6.2.1 reset() [1/2] 25
5.6.2.2 reset() [2/2] 25
5.7 ChangeFactionDialog.Callback Interface Reference 25
5.7.1 Detailed Description 26
5.7.2 Member Function Documentation 26
5.7.2.1 onThemeSelect() 26
5.8 CardListAdapter Class Reference 26
5.8.1 Detailed Description 28
5.8.2 Constructor & Destructor Documentation 28
5.8.2.1 CardListAdapter() 28
5.8.3 Member Function Documentation 28
5.8.3.1 getItemId() 28
5.8.3.2 onBindViewHolder() 29
5.8.3.3 onCreateViewHolder() 29
5.8.4 Member Data Documentation 29
5.8.4.1 onCopy 29
5.8.4.2 onRemove 30
5.9 CardNumberPickerAdapter Class Reference 30
5.9.1 Detailed Description 31
5.9.2 Constructor & Destructor Documentation 31
5.9.2.1 CardNumberPickerAdapter() 31
5.9.3 Member Function Documentation 32
5.9.3.1 addSelectedUnits() 32
5.9.3.2 getDelayedOnValueChangeListener() 32
5.9.4 Member Data Documentation 32
5.9.4.1 abilityPicker 32
5.9.4.2 damagePicker 32
5.9.4.3 EPIC_UNIT_ABILITIES 33

5.9.4.4 epicPicker 33
5.9.4.5 numberPicker 33
5.9.4.6 squadManager 33
5.9.4.7 squadPicker 33
5.10 CardUiState Class Reference 33
5.10.1 Detailed Description 35
5.10.2 Constructor & Destructor Documentation 35
5.10.2.1 CardUiState() 35
5.10.3 Member Function Documentation 35
5.10.3.1 equals() 35
5.10.3.2 getAbilityImageId() 36
5.10.3.3 getDamageBackgroundImageId() 36
5.10.3.4 getDamageString() 36
5.10.3.5 getDamageTextColor() 37
5.10.3.6 getSquadString() 37
5.10.3.7 getUnitId() 37
5.10.3.8 showAbility() 37
5.10.3.9 showSquad() 38
5.10.4 Member Data Documentation 38
5.10.4.1 abilityImageId 38
5.10.4.2 damageBackgroundImageId 38
5.10.4.3 damageString 38
5.10.4.4 damageTextColor 38
5.10.4.5 DIFF_CALLBACK 39
5.10.4.6 squadString 39
5.10.4.7 unitId 39
5.10.4.8 UNUSED 39
5.11 CardUiStateFactory Class Reference 40
5.11.1 Detailed Description 40
5.11.2 Constructor & Destructor Documentation 40
5.11.2.1 CardUiStateFactory() 40
5.11.3 Member Function Documentation 41
5.11.3.1 createCardUiState() [1/2] 41
5.11.3.2 createCardUiState() [2/2] 41
5.11.4 Member Data Documentation 41
5.11.4.1 damageTextColors 41
5.11.4.2 horn 42
5.11.4.3 weather 42
5.12 CardListAdapter.CardViewHolder Class Reference 42
5.12.1 Detailed Description 43
5.12.2 Constructor & Destructor Documentation 44
5.12.2.1 CardViewHolder() 44

5.12.3 Member Function Documentation 44
5.12.3.1 getAbilityView() 44
5.12.3.2 getBindingView() 44
5.12.3.3 getDamageView() 44
5.12.3.4 setItemId() 45
5.12.4 Member Data Documentation 45
5.12.4.1 abilityView 45
5.12.4.2 bindingView 45
5.12.4.3 damageView 46
5.12.4.4 itemId 46
5.13 ChangeFactionDialog Class Reference 46
5.13.1 Detailed Description 48
5.13.2 Constructor & Destructor Documentation 48
5.13.2.1 ChangeFactionDialog() 48
5.13.3 Member Function Documentation 48
5.13.3.1 getOnThemeClickListener() 48
5.13.3.2 onCreate() 48
5.13.4 Member Data Documentation 49
5.13.4.1 callback 49
5.14 CoinFlipDialog Class Reference 49
5.14.1 Detailed Description 50
5.14.2 Constructor & Destructor Documentation 51
5.14.2.1 CoinFlipDialog() 51
5.14.3 Member Function Documentation 51
5.14.3.1 onCreate() 51
5.14.4 Member Data Documentation 51
5.14.4.1 RANDOM 51
5.15 DamageCalculator.Color Enum Reference 51
5.15.1 Detailed Description 52
5.15.2 Member Data Documentation 52
5.15.2.1 BUFFED 52
5.15.2.2 DEBUFFED 52
5.15.2.3 DEFAULT 52
5.16 DamageCalculator Interface Reference 52
5.16.1 Detailed Description 53
5.16.2 Member Function Documentation 53
5.16.2.1 calculateDamage() 53
5.16.2.2 isBuffed() 54
5.17 DamageCalculatorBuildDirector Class Reference 54
5.17.1 Detailed Description 55
5.17.2 Member Function Documentation 55
5.17.2.1 getCalculator() 55

5.17.2.2 setHorns() 56
5.17.2.3 setMoralBoosts() 56
5.17.2.4 setSquads() 56
5.18 DamageCalculatorBuilder Class Reference 57
5.18.1 Detailed Description 58
5.18.2 Member Function Documentation 58
5.18.2.1 getResult() 58
5.18.2.2 setBond() 58
5.18.2.3 setHorn() 58
5.18.2.4 setMoral() 59
5.18.2.5 setWeather() 59
5.18.3 Member Data Documentation 59
5.18.3.1 calculator 59
5.19 DamageCalculatorDecorator Class Reference 59
5.19.1 Detailed Description 60
5.19.2 Constructor & Destructor Documentation 61
5.19.2.1 DamageCalculatorDecorator() 61
5.19.3 Member Data Documentation 61
5.19.3.1 component 61
5.20 DamageCalculatorUseCase Class Reference 61
5.20.1 Detailed Description 62
5.20.2 Member Function Documentation 62
5.20.2.1 getDamageCalculator() [1/2] 62
5.20.2.2 getDamageCalculator() [2/2] 62
5.21 DamageValuePicker Class Reference 63
5.21.1 Detailed Description 64
5.21.2 Constructor & Destructor Documentation 64
5.21.2.1 DamageValuePicker() 64
5.21.3 Member Function Documentation 64
5.21.3.1 getDisplayString() 64
5.21.3.2 getValue() 65
5.21.3.3 setEpicValues() 65
5.21.3.4 setSelectableValues() 65
5.21.3.5 setValue() 65
5.21.4 Member Data Documentation 66
5.21.4.1 epicValues 66
5.22 ResetDialogUseCase.DialogType Enum Reference 66
5.22.1 Detailed Description 66
5.22.2 Member Data Documentation 67
5.22.2.1 DEFAULT 67
5.22.2.2 MONSTER 67
5.22.2.3 NONE 67

5.23 FactionSwitchListener Class Reference 67
5.23.1 Detailed Description 69
5.23.2 Constructor & Destructor Documentation 69
5.23.2.1 FactionSwitchListener() 69
5.23.3 Member Function Documentation 69
5.23.3.1 getListener() 69
5.23.3.2 onSharedPreferenceChanged() 70
5.23.3.3 setTheme() 70
5.23.4 Member Data Documentation 70
5.23.4.1 ballViews 70
5.23.4.2 cardViews 71
5.23.4.3 factionButton 71
5.23.4.4 THEME_MONSTER 71
5.23.4.5 THEME_NILFGAARD 71
5.23.4.6 THEME_NORTHERN_KINGDOMS 71
5.23.4.7 THEME_PREFERENCE_KEY 72
5.23.4.8 THEME_SCOIATAEL 72
5.23.4.9 unitViews 72
5.24 GameBoardViewModel Class Reference 72
5.24.1 Detailed Description 74
5.24.2 Constructor & Destructor Documentation 74
5.24.2.1 GameBoardViewModel() 74
5.24.3 Member Function Documentation 74
5.24.3.1 getMenuUiState() 74
5.24.3.2 getModel() 75
5.24.3.3 getRepository() 75
5.24.3.4 getRowUiState() 75
5.24.3.5 getSoundManager() 76
5.24.3.6 onBurnButtonPressed() 76
5.24.3.7 onFactionSwitchReset() 76
5.24.3.8 onHornViewPressed() 77
5.24.3.9 onResetButtonPressed() 77
5.24.3.10 onWeatherButtonPressed() 78
5.24.3.11 onWeatherViewPressed() 78
5.24.3.12 reset() 78
5.24.4 Member Data Documentation 79
5.24.4.1 INITIALIZER 79
5.24.4.2 menuUiState 79
5.24.4.3 rowUiStates 79
5.24.4.4 soundManager 80
5.25 GwentApplication Class Reference 80
5.25.1 Detailed Description 81

5.25.2 Member Function Documentation 81
5.25.2.1 getRepository() [1/2] 81
5.25.2.2 getRepository() [2/2] 81
5.25.2.3 onCreate() 82
5.25.3 Member Data Documentation 82
5.25.3.1 database 82
5.25.3.2 repository 82
5.26 HornDamageCalculatorDecorator Class Reference 82
5.26.1 Detailed Description 84
5.26.2 Constructor & Destructor Documentation 84
5.26.2.1 HornDamageCalculatorDecorator() 84
5.26.3 Member Function Documentation 85
5.26.3.1 calculateDamage() 85
5.26.3.2 doubleDamage() 85
5.26.3.3 isBuffed() 85
5.26.4 Member Data Documentation 86
5.26.4.1 unitIds 86
5.27 ImageViewSwitchAnimator Class Reference 86
5.27.1 Detailed Description 87
5.27.2 Member Function Documentation 87
5.27.2.1 animatedSwitch() 87
5.27.2.2 getOverlayView() 87
5.28 IndicatorManager Class Reference 88
5.28.1 Detailed Description 88
5.28.2 Constructor & Destructor Documentation 88
5.28.2.1 IndicatorManager() 88
5.28.3 Member Function Documentation 89
5.28.3.1 updateIndicators() 89
5.28.4 Member Data Documentation 89
5.28.4.1 currentView 89
5.28.4.2 indicators 89
5.29 IntroductionActivity Class Reference 90
5.29.1 Detailed Description 90
5.29.2 Member Function Documentation 91
5.29.2.1 onCreate() 91
5.29.3 Member Data Documentation 91
5.29.3.1 indicatorManager 91
5.30 IntroductionFragment Class Reference 91
5.30.1 Detailed Description 93
5.30.2 Member Function Documentation 93
5.30.2.1 newInstance() 93
5.30.2.2 onCreateView() 93

5.30.3 Member Data Documentation 93
5.30.3.1 ARG_SECTION_NUMBER 93
5.30.3.2 imageIds 94
5.30.3.3 PAGES_COUNT 94
5.30.3.4 stringIds 94
5.31 MainActivity Class Reference 94
5.31.1 Detailed Description 96
5.31.2 Member Function Documentation 96
5.31.2.1 inflateCoinFlipPopup() 96
5.31.2.2 inflateFactionPopup() 96
5.31.2.3 initializeViewModel() 96
5.31.2.4 onCreate() 96
5.31.2.5 onDestroy() 97
5.31.2.6 onResume() 97
5.31.2.7 onWindowFocusChanged() 97
5.31.3 Member Data Documentation 97
5.31.3.1 disposables 97
5.31.3.2 factionSwitchListener 97
5.31.3.3 gameBoardViewModel 97
5.32 MenuUiState Class Reference 98
5.32.1 Detailed Description 98
5.32.2 Constructor & Destructor Documentation 99
5.32.2.1 MenuUiState() 99
5.32.3 Member Function Documentation 99
5.32.3.1 equals() 99
5.32.3.2 getDamage() 99
5.32.3.3 isBurn() 99
5.32.3.4 isReset() 99
5.32.3.5 isWeather() 100
5.32.4 Member Data Documentation 100
5.32.4.1 burn 100
5.32.4.2 damage 100
5.32.4.3 reset 100
5.32.4.4 weather 100
5.33 MenuUiStateObserver Class Reference 100
5.33.1 Detailed Description 102
5.33.2 Constructor & Destructor Documentation 102
5.33.2.1 MenuUiStateObserver() 102
5.33.3 Member Function Documentation 102
5.33.3.1 accept() 102
5.33.4 Member Data Documentation 102
5.33.4.1 burnButton 102

5.33.4.2 damageView 103
5.33.4.3 resetButton 103
5.33.4.4 weatherButton 103
5.34 MoralDamageCalculatorDecorator Class Reference 103
5.34.1 Detailed Description 104
5.34.2 Constructor & Destructor Documentation 105
5.34.2.1 MoralDamageCalculatorDecorator() 105
5.34.3 Member Function Documentation 105
5.34.3.1 calculateDamage() 105
5.34.3.2 isBuffed() 106
5.34.4 Member Data Documentation 106
5.34.4.1 unitIds 106
5.35 OnValueChangeListener< T extends Comparable Interface Template Reference 106
5.35.1 Detailed Description 107
5.36 OverlayDialog Class Reference 107
5.36.1 Detailed Description 108
5.36.2 Constructor & Destructor Documentation 108
5.36.2.1 OverlayDialog() [1/2] 108
5.36.2.2 OverlayDialog() [2/2] 108
5.36.3 Member Function Documentation 109
5.36.3.1 onCreate() 109
5.36.4 Member Data Documentation 109
5.36.4.1 BACKGROUND 109
5.36.4.2 dismissViewId 109
5.36.4.3 layout 109
5.36.4.4 NO_DISMISS_VIEW 109
5.37 RemoveUnitsUseCase Class Reference 110
5.37.1 Detailed Description 110
5.37.2 Member Function Documentation 110
5.37.2.1 getRevengeDialog() 110
5.37.2.2 remove() [1/2] 111
5.37.2.3 remove() [2/2] 112
5.38 ResetAlertDialogBuilderAdapter Class Reference 112
5.38.1 Detailed Description 113
5.38.2 Constructor & Destructor Documentation 113
5.38.2.1 ResetAlertDialogBuilderAdapter() 113
5.38.3 Member Function Documentation 113
5.38.3.1 create() 113
5.38.3.2 setMonsterDialog() 114
5.38.3.3 setTrigger() 114
5.38.4 Member Data Documentation 114
5.38.4.1 adapteeBuilder 114

5.38.4.2 dialogCallback 114
5.39 ResetDialogUseCase Class Reference 115
5.39.1 Detailed Description 115
5.39.2 Member Function Documentation 116
5.39.2.1 getDialogType() 116
5.39.2.2 reset() [1/2] 116
5.39.2.3 reset() [2/2] 117
5.40 ResetRepositoryUseCase Class Reference 117
5.40.1 Detailed Description 118
5.40.2 Member Function Documentation 118
5.40.2.1 getRandomUnit() 118
5.40.2.2 getRevengeDialog() 118
5.40.2.3 reset() [1/2] 119
5.40.2.4 reset() [2/2] 119
5.41 RevengeAlertDialogBuilderAdapter Class Reference 120
5.41.1 Detailed Description 121
5.41.2 Constructor & Destructor Documentation 121
5.41.2.1 RevengeAlertDialogBuilderAdapter() 121
5.41.3 Member Function Documentation 121
5.41.3.1 create() 121
5.41.3.2 insertAvengers() 121
5.41.3.3 setNegativeCallback() 122
5.41.3.4 setPositiveCallback() 122
5.41.4 Member Data Documentation 123
5.41.4.1 adapteeBuilder 123
5.41.4.2 AVENGER_ABILITY 123
5.41.4.3 AVENGER_DAMAGE 123
5.41.4.4 AVENGER_EPIC 123
5.41.4.5 AVENGER_ROW 123
5.41.4.6 AVENGER_SQUAD 123
5.42 RowDao Interface Reference 124
5.42.1 Detailed Description 124
5.42.2 Member Function Documentation 125
5.42.2.1 clearRows() 125
5.42.2.2 clearWeather() 125
5.42.2.3 insertRow() 125
5.42.2.4 isHorn() 125
5.42.2.5 isHornFlowable() 126
5.42.2.6 isWeather() 126
5.42.2.7 isWeatherFlowable() 126
5.42.2.8 updateHorn() 127
5.42.2.9 updateWeather() 127

5.43 RowEntity Class Reference 127
5.43.1 Detailed Description 128
5.43.2 Constructor & Destructor Documentation 128
5.43.2.1 RowEntity() 128
5.43.3 Member Function Documentation 130
5.43.3.1 getId() 130
5.43.3.2 isHorn() 130
5.43.3.3 isWeather() 130
5.43.3.4 setHorn() 130
5.43.3.5 setWeather() 130
5.43.4 Member Data Documentation 131
5.43.4.1 horn 131
5.43.4.2 id 131
5.43.4.3 weather 131
5.44 RowType Enum Reference 131
5.44.1 Detailed Description 132
5.44.2 Member Data Documentation 132
5.44.2.1 MELEE 132
5.44.2.2 RANGE 132
5.44.2.3 SIEGE 132
5.45 RowUiState Class Reference 132
5.45.1 Detailed Description 133
5.45.2 Constructor & Destructor Documentation 133
5.45.2.1 RowUiState() 133
5.45.3 Member Function Documentation 133
5.45.3.1 equals() 133
5.45.3.2 getDamage() 133
5.45.3.3 getUnits() 134
5.45.3.4 isHorn() 134
5.45.3.5 isWeather() 134
5.45.4 Member Data Documentation 134
5.45.4.1 damage 134
5.45.4.2 horn 134
5.45.4.3 units 135
5.45.4.4 weather 135
5.46 RowUiStateObserver Class Reference 135
5.46.1 Detailed Description 136
5.46.2 Constructor & Destructor Documentation 136
5.46.2.1 RowUiStateObserver() 136
5.46.3 Member Function Documentation 137
5.46.3.1 accept() 137
5.46.3.2 getObserver() 137

5.46.4 Member Data Documentation 138
5.46.4.1 damageView 138
5.46.4.2 hornView 138
5.46.4.3 unitView 138
5.46.4.4 weatherResource 138
5.46.4.5 weatherView 138
5.47 RuleActivity Class Reference 138
5.47.1 Detailed Description 139
5.47.2 Member Function Documentation 139
5.47.2.1 onCreate() 139
5.47.2.2 onOptionsItemSelected() 140
5.47.3 Member Data Documentation 140
5.47.3.1 INTENT_EXTRA_TAG 140
5.48 RuleSection Enum Reference 140
5.48.1 Detailed Description 141
5.48.2 Member Data Documentation 141
5.48.2.1 CARD_ABILITIES 141
5.48.2.2 CARDS 141
5.48.2.3 COMMANDER 141
5.48.2.4 COURSE 141
5.48.2.5 FACTIONS 141
5.48.2.6 GENERAL 141
5.48.2.7 SPECIAL_CARDS 142
5.49 SectionsPagerAdapter Class Reference 142
5.49.1 Detailed Description 143
5.49.2 Constructor & Destructor Documentation 143
5.49.2.1 SectionsPagerAdapter() 143
5.49.3 Member Function Documentation 143
5.49.3.1 createFragment() 143
5.49.3.2 getItemCount() 143
5.50 SettingsActivity Class Reference 143
5.50.1 Detailed Description 145
5.50.2 Member Function Documentation 145
5.50.2.1 onCreate() 145
5.50.2.2 onOptionsItemSelected() 145
5.50.2.3 onPreferenceStartFragment() 145
5.50.2.4 onSaveInstanceState() 146
5.50.3 Member Data Documentation 146
5.50.3.1 CURRENT_FRAGMENT_KEY 146
5.51 SettingsHeaderFragment Class Reference 146
5.51.1 Detailed Description 147
5.51.2 Member Function Documentation 147

5.51.2.1 onCreatePreferences() 147
5.52 SettingsRuleFragment Class Reference 147
5.52.1 Detailed Description 148
5.52.2 Member Function Documentation 148
5.52.2.1 getSectionClickListener() 148
5.52.2.2 onCreatePreferences() 149
5.53 SettingsSoundFragment Class Reference 149
5.53.1 Detailed Description 150
5.53.2 Member Function Documentation 150
5.53.2.1 onCreatePreferences() 150
5.54 ShowUnitsDialog Class Reference 151
5.54.1 Detailed Description 152
5.54.2 Constructor & Destructor Documentation 153
5.54.2.1 ShowUnitsDialog() 153
5.54.3 Member Function Documentation 153
5.54.3.1 getDialog() 153
5.54.3.2 getRow() 153
5.54.3.3 onCreate() 154
5.54.4 Member Data Documentation 154
5.54.4.1 cardListAdapter 154
5.54.4.2 disposables 154
5.54.4.3 row 154
5.54.4.4 soundManager 155
5.54.4.5 unitRecyclerView 155
5.55 Sound Class Reference 155
5.55.1 Detailed Description 156
5.55.2 Constructor & Destructor Documentation 156
5.55.2.1 Sound() 156
5.55.3 Member Function Documentation 156
5.55.3.1 createSound() 156
5.55.3.2 getPreferenceKey() 157
5.55.3.3 getSoundId() 157
5.55.3.4 isActivated() 157
5.55.3.5 setActivated() [1/2] 157
5.55.3.6 setActivated() [2/2] 158
5.55.4 Member Data Documentation 158
5.55.4.1 activated 158
5.55.4.2 preferenceKey 158
5.55.4.3 soundId 158
5.56 SoundManager Class Reference 158
5.56.1 Detailed Description 160
5.56.2 Constructor & Destructor Documentation 160

5.56.2.1 SoundManager() 160
5.56.3 Member Function Documentation 161
5.56.3.1 playBurnSound() 161
5.56.3.2 playCardAddSound() 161
5.56.3.3 playCardRemovedSound() 161
5.56.3.4 playClearWeatherSound() 161
5.56.3.5 playCoinSound() 162
5.56.3.6 playHornSound() 162
5.56.3.7 playResetSound() 162
5.56.3.8 playSound() 162
5.56.3.9 playWeatherSound() 163
5.56.4 Member Data Documentation 163
5.56.4.1 changeListener 163
5.56.4.2 SOUND_BURN 163
5.56.4.3 SOUND_CARDS_EPIC 163
5.56.4.4 SOUND_CARDS_MELEE 164
5.56.4.5 SOUND_CARDS_RANGE 164
5.56.4.6 SOUND_CARDS_SIEGE 164
5.56.4.7 SOUND_COIN 164
5.56.4.8 SOUND_HORN 164
5.56.4.9 SOUND_RESET 165
5.56.4.10 SOUND_WEATHER_FOG 165
5.56.4.11 SOUND_WEATHER_FROST 165
5.56.4.12 SOUND_WEATHER_GOOD 165
5.56.4.13 SOUND_WEATHER_RAIN 165
5.56.4.14 soundPool 166
5.56.4.15 sounds 166
5.57 SquadManager Class Reference 166
5.57.1 Detailed Description 167
5.57.2 Constructor & Destructor Documentation 167
5.57.2.1 SquadManager() 167
5.57.3 Member Function Documentation 167
5.57.3.1 getFirstSquadWithMembers() 167
5.57.3.2 onSquadChanged() 167
5.57.4 Member Data Documentation 168
5.57.4.1 MAX_NR_SQUADS 168
5.57.4.2 states 168
5.58 SquadState Class Reference 168
5.58.1 Detailed Description 169
5.58.2 Constructor & Destructor Documentation 169
5.58.2.1 SquadState() 169
5.58.3 Member Function Documentation 170

5.58.3.1 getMemberBaseDamage() . . . 170
5.58.3.2 getSquadMembers() . . . 170
5.58.3.3 getSquadNumber() . . . 170
5.58.3.4 getState() . . . 170
5.58.3.5 hasMembers() . . . 171
5.58.4 Member Data Documentation . . . 171
5.58.4.1 memberBaseDamage . . . 171
5.58.4.2 squadMembers . . . 171
5.58.4.3 squadNumber . . . 172
5.59 StringValuePicker< T extends Comparable Class Template Reference . . . 172
5.59.1 Detailed Description . . . 172
5.60 ResetDialogUseCase.Trigger Enum Reference . . . 172
5.60.1 Detailed Description . . . 173
5.60.2 Member Data Documentation . . . 173
5.60.2.1 BUTTON_CLICK . . . 173
5.60.2.2 FACTION_SWITCH . . . 173
5.61 UnitDao Interface Reference . . . 173
5.61.1 Detailed Description . . . 174
5.61.2 Member Function Documentation . . . 174
5.61.2.1 countUnits() [1/2] . . . 174
5.61.2.2 countUnits() [2/2] . . . 174
5.61.2.3 deleteUnit() . . . 174
5.61.2.4 deleteUnits() . . . 175
5.61.2.5 getUnit() . . . 175
5.61.2.6 getUnits() [1/2] . . . 175
5.61.2.7 getUnits() [2/2] . . . 175
5.61.2.8 getUnitsFlowable() . . . 176
5.61.2.9 hasNonEpicUnitsFlowable() . . . 176
5.61.2.10 insertUnit() [1/2] . . . 176
5.61.2.11 insertUnit() [2/2] . . . 176
5.62 UnitEntity Class Reference . . . 177
5.62.1 Detailed Description . . . 179
5.62.2 Constructor & Destructor Documentation . . . 179
5.62.2.1 UnitEntity() . . . 179
5.62.3 Member Function Documentation . . . 179
5.62.3.1 calculateDamage() . . . 179
5.62.3.2 collectionToString() . . . 180
5.62.3.3 getAbility() . . . 180
5.62.3.4 getDamage() . . . 180
5.62.3.5 getId() . . . 180
5.62.3.6 getRow() . . . 181
5.62.3.7 getSquad() . . . 181

5.62.3.8 isBuffed() 181
5.62.3.9 isEpic() 181
5.62.3.10 setAbility() 182
5.62.3.11 setDamage() 182
5.62.3.12 setEpic() 182
5.62.3.13 setId() 182
5.62.3.14 setRow() 183
5.62.3.15 setSquad() 183
5.62.3.16 toString() 183
5.62.4 Member Data Documentation 183
5.62.4.1 ability 183
5.62.4.2 damage 184
5.62.4.3 epic 184
5.62.4.4 EPIC_DAMAGE_VALUES 184
5.62.4.5 id 184
5.62.4.6 NON_EPIC_DAMAGE_VALUES_UPPER_BOUND 184
5.62.4.7 row 184
5.62.4.8 squad 184
5.63 UnitRepository Class Reference 184
5.63.1 Detailed Description 186
5.63.2 Constructor & Destructor Documentation 186
5.63.2.1 UnitRepository() 186
5.63.3 Member Function Documentation 187
5.63.3.1 clearWeather() 187
5.63.3.2 copy() 187
5.63.3.3 countUnits() [1/2] 187
5.63.3.4 countUnits() [2/2] 187
5.63.3.5 delete() [1/2] 188
5.63.3.6 delete() [2/2] 188
5.63.3.7 getRepository() 188
5.63.3.8 getUnit() 189
5.63.3.9 getUnits() [1/2] 189
5.63.3.10 getUnits() [2/2] 189
5.63.3.11 getUnitsFlowable() 190
5.63.3.12 hasNonEpicUnitsFlowable() 190
5.63.3.13 initializeRows() 190
5.63.3.14 insertUnit() [1/2] 190
5.63.3.15 insertUnit() [2/2] 191
5.63.3.16 isHorn() 192
5.63.3.17 isHornFlowable() 192
5.63.3.18 isWeather() 192
5.63.3.19 isWeatherFlowable() 193

5.63.3.20 reset() [1/2] 193
5.63.3.21 reset() [2/2] 193
5.63.3.22 switchHorn() 194
5.63.3.23 switchWeather() 194
5.63.4 Member Data Documentation 194
5.63.4.1 database 194
5.64 ValuePicker< T extends Comparable Class Template Reference 195
5.64.1 Detailed Description 195
5.65 WeatherDamageCalculator Class Reference 195
5.65.1 Detailed Description 196
5.65.2 Constructor & Destructor Documentation 197
5.65.2.1 WeatherDamageCalculator() 197
5.65.3 Member Function Documentation 197
5.65.3.1 calculateDamage() 197
5.65.3.2 isBuffed() 197
5.65.4 Member Data Documentation 198
5.65.4.1 weather 198

6 File Documentation 199

6.1 /home/runner/work/Gwent/Gwent/app/src/main/AndroidManifest.xml 199
6.2 Ability.java 199
6.3 AppDatabase.java 200
6.4 RowDao.java 200
6.5 RowEntity.java 200
6.6 RowType.java 201
6.7 UnitDao.java 201
6.8 UnitEntity.java 202
6.9 UnitRepository.java 204
6.10 BurnDialogUseCase.java 207
6.11 DamageCalculatorUseCase.java 208
6.12 RemoveUnitsUseCase.java 209
6.13 ResetAlertDialogBuilderAdapter.java 210
6.14 ResetDialogUseCase.java 210
6.15 ResetRepositoryUseCase.java 212
6.16 RevengeAlertDialogBuilderAdapter.java 213
6.17 BondDamageCalculatorDecorator.java 214
6.18 DamageCalculator.java 214
6.19 DamageCalculatorBuildDirector.java 214
6.20 DamageCalculatorBuilder.java 215
6.21 DamageCalculatorDecorator.java 216
6.22 HornDamageCalculatorDecorator.java 216
6.23 MoralDamageCalculatorDecorator.java 216

6.24 WeatherDamageCalculator.java 217
6.25 GwentApplication.java 217
6.26 AddCardDialog.java 218
6.27 CardNumberPickerAdapter.java 219
6.28 DamageValuePicker.java 220
6.29 SquadManager.java 221
6.30 SquadState.java 222
6.31 StringValuePicker.java 223
6.32 ValuePicker.java 224
6.33 CardListAdapter.java 225
6.34 CardUiState.java 227
6.35 CardUiStateFactory.java 228
6.36 ShowUnitsDialog.java 229
6.37 ChangeFactionDialog.java 231
6.38 CoinFlipDialog.java 232
6.39 OverlayDialog.java 232
6.40 IndicatorManager.java 233
6.41 IntroductionActivity.java 233
6.42 IntroductionFragment.java 234
6.43 SectionsPagerAdapter.java 235
6.44 FactionSwitchListener.java 235
6.45 GameBoardViewModel.java 237
6.46 ImageViewSwitchAnimator.java 239
6.47 MainActivity.java 240
6.48 MenuUiState.java 243
6.49 MenuUiStateObserver.java 243
6.50 RowUiState.java 244
6.51 RowUiStateObserver.java 245
6.52 RuleActivity.java 246
6.53 RuleSection.java 247
6.54 SettingsActivity.java 247
6.55 SettingsHeaderFragment.java 248
6.56 SettingsRuleFragment.java 248
6.57 SettingsSoundFragment.java 249
6.58 Sound.java 250
6.59 SoundManager.java 250
6.60 background_drawable.xml 252
6.61 icon_copy.xml 253
6.62 icon_delete.xml 253
6.63 icon_design.xml 253
6.64 icon_explore.xml 254
6.65 icon_factionreset.xml 254

6.66 icon_next.xml 254
6.67 icon_rules.xml 254
6.68 icon_sounds.xml 254
6.69 icon_warnings.xml 255
6.70 indicator_selected.xml 255
6.71 indicator_unselected.xml 255
6.72 rectangle.xml 255
6.73 rectangle_black_transparent.xml 255
6.74 rectangle_cancel_popup.xml 255
6.75 fragment_introduction.xml 256
6.76 fragment_introduction.xml 256
6.77 fragment_introduction.xml 257
6.78 fragment_introduction.xml 257
6.79 activity_rules.xml 258
6.80 activity_rules.xml 258
6.81 activity_introduction.xml 259
6.82 activity_main.xml 260
6.83 activity_settings.xml 262
6.84 alertdialog_checkbox.xml 263
6.85 card.xml 263
6.86 popup_add_card.xml 264
6.87 popup_cards.xml 266
6.88 popup_coin_normal.xml 267
6.89 popup_coin_stewie.xml 267
6.90 popup_coin_terry.xml 269
6.91 popup_coin_vin.xml 270
6.92 popup_faction.xml 270
6.93 row.xml 272
6.94 icon_launcher.xml 273
6.95 icon_launcher_round.xml 273
6.96 arrays.xml 273
6.97 attr.xml 274
6.98 colors.xml 274
6.99 dimens.xml 274
6.100 icon_launcher_background.xml 275
6.101 preferences.xml 275
6.102 strings.xml 275
6.103 strings.xml 279
6.104 styles.xml 283
6.105 header_preferences.xml 284
6.106 rule_preferences.xml 285
6.107 sound_preferences.xml 285

Index **287**

Chapter 1

Gwent - A Calculator

1. Overview
2. Printing analog cards
3. Documentation

1.1 Overview

Gwent - A Calculator is an Android app (available in Play Store) that enables the user to play the Gwent card game from The Witcher 3: Wild Hunt with analog cards. This is accomplished by providing an easy-to-use interface which allows adding cards with certain attributes that correspond to the cards played analogously. The app then calculates the power score of the player including weather effects and card abilities. To see what this looks like, please visit the Play Store entry!

1.2 Printing analog cards

To actually play Gwent using *Gwent - A Calculator*, you need to print out analog Gwent cards. The card images can be downloaded from the following links:

Language	**zip**	**tar**
English	gwent_cards_en.zip	gwent_cards_en.tar.gz
German	gwent_cards_de.zip	gwent_cards_de.tar.gz

Please be aware that the Skellige faction is only available in the german packages! This PDF summarizes in which quantity you need to print which card.
When you installed the app, printed out your deck and found a friend, you are ready to go and can start playing the original Gwent game from The Witcher 3: Wild Hunt.

1.3 Documentation

If you are a developer and interested in improving the app by adding new features or fixing bugs, you are always welcome to open pull requests. An extensive reference specification of the whole app can be found on this website or in this PDF-file.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- Ability 13
- AndroidViewModel
  - GameBoardViewModel 72
- AppCompatActivity
  - IntroductionActivity 90
  - MainActivity 94
  - RuleActivity 138
  - SettingsActivity 143
- Application
  - GwentApplication 80
- BurnDialogUseCase 22
- ResetAlertDialogBuilderAdapter.Callback 24
- ChangeFactionDialog.Callback 25
- CardNumberPickerAdapter 30
- CardUiState 33
- CardUiStateFactory 40
- DamageCalculator.Color 51
- Consumer
  - MenuUiStateObserver 100
  - RowUiStateObserver 135
- DamageCalculator 52
  - DamageCalculatorDecorator 59
    - BondDamageCalculatorDecorator 19
    - HornDamageCalculatorDecorator 82
    - MoralDamageCalculatorDecorator 103
  - WeatherDamageCalculator 195
- DamageCalculatorBuildDirector 54
- DamageCalculatorBuilder 57
- DamageCalculatorUseCase 61
- Dialog
  - OverlayDialog 107
    - ChangeFactionDialog 46
    - CoinFlipDialog 49
    - AddCardDialog 14
    - ShowUnitsDialog 151
- ResetDialogUseCase.DialogType 66
- Fragment
  - IntroductionFragment 91
- FragmentStateAdapter
  - SectionsPagerAdapter 142

ImageViewSwitchAnimator 86
IndicatorManager 88
ListAdapter
- CardListAdapter 26

MenuUiState 98
OnPreferenceStartFragmentCallback
- SettingsActivity 143

OnSharedPreferenceChangeListener
- FactionSwitchListener 67

OnValueChangeListener< T extends Comparable 106
PreferenceFragmentCompat
- SettingsHeaderFragment 146
- SettingsRuleFragment 147
- SettingsSoundFragment 149

RemoveUnitsUseCase 110
ResetAlertDialogBuilderAdapter 112
ResetDialogUseCase 115
ResetRepositoryUseCase 117
RevengeAlertDialogBuilderAdapter 120
RoomDatabase
- AppDatabase 18

RowDao 124
RowEntity 127
RowType 131
RowUiState 132
RuleSection 140
Sound 155
SoundManager 158
SquadManager 166
SquadState 168
StringValuePicker< T extends Comparable 172
ResetDialogUseCase.Trigger 172
UnitDao 173
UnitEntity 177
UnitRepository 184
ValuePicker
- DamageValuePicker 63

ValuePicker< T extends Comparable 195
ViewHolder
- CardListAdapter.CardViewHolder 42

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Ability
An enum listing the different card abilities possible 13

AddCardDialog
An OverlayDialog used to add new UnitEntity objects to a certain #row 14

AppDatabase
An abstract class extending RoomDatabase and representing the SQLite database of the application 18

BondDamageCalculatorDecorator
A DamageCalculator class responsible for calculating the tight bond buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::BINDING ability 19

BurnDialogUseCase
A use case class responsible for dispatching a remove call to RemoveUnitsUseCase 22

ResetAlertDialogBuilderAdapter.Callback
An interface defining functions to propagate the user's decision back to the creator 24

ChangeFactionDialog.Callback
An interface used for propagating a selected theme back to the creator of a ChangeFactionDialog 25

CardListAdapter
A ListAdapter used to provide CardListAdapter.CardViewHolder objects created from CardUiState objects to a RecyclerView 26

CardNumberPickerAdapter
A helper class responsible for initializing the NumberPicker views of an AddCardDialog (in #CardNumperPickerAdapter()) and adding UnitEntity objects with the selected attributes when #addSelectedUnits() is called 30

CardUiState
A data class encapsulating the visible state of a card in the card list shown by the ShowUnitsDialog 33

CardUiStateFactory
A factory class responsible for creating CardUiState objects from UnitEntity objects 40

CardListAdapter.CardViewHolder
A RecyclerView.ViewHolder class managing a card view that shows a representation of the UnitEntity referenced by #itemId 42

ChangeFactionDialog
An OverlayDialog class used to change the faction design 46

CoinFlipDialog
An OverlayDialog class showing the result of a coin-flip 49

DamageCalculator.Color
An enum containing values representing whether a unit is buffed, de-buffed or not affected by status effects 51

DamageCalculator
An interface for a class capable of calculating the (de-)buffed damage of a unit for a given Unit↩ Entity::id and UnitEntity::damage 52
DamageCalculatorBuildDirector
A build director responsible for the creation of DamageCalculator instances from sets of units using DamageCalculatorBuilder 54
DamageCalculatorBuilder
A builder class responsible for creating a decorator structure of DamageCalculator classes with the given parameters 57
DamageCalculatorDecorator
An abstract DamageCalculator that is used as a superclass for the decorator classes implementing the damage calculation 59
DamageCalculatorUseCase
A use case class responsible for creating DamageCalculator objects for rows using Damage↩ CalculatorBuildDirector 61
DamageValuePicker
A ValuePicker used for selection of damage values for UnitEntity::damage 63
ResetDialogUseCase.DialogType
An Enum defining which form of Dialog should be shown 66
FactionSwitchListener
A class implementing SharedPreference.OnSharedPreferenceChangeListener for the #↩ THEME_PREFERENCE_KEY SharedPreference 67
GameBoardViewModel
An AndroidViewModel class responsible for encapsulating and offering state of views in activity_main.xml, i.e., that show the overall game board 72
GwentApplication
An Application encapsulating the Gwent app 80
HornDamageCalculatorDecorator
A DamageCalculator class responsible for calculating the horn buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::HORN ability that are not the unit the damage is calculated for or if there is a commander's horn in this row 82
ImageViewSwitchAnimator
A class providing functionality for changing the resource shown by an ImageView using a fading animation 86
IndicatorManager
A class managing the indicator views shown in the bottom bar of the IntroductionActivity 88
IntroductionActivity
An AppCompatActivity that gives the user an introduction into the usage of the application 90
IntroductionFragment
A Fragment containing an ImageView and a TextView conveying introductory information 91
MainActivity
An AppCompatActivity that is called on startup and that encapsulates the main view onto the game board 94
MenuUiState
A data class encapsulating the visible state of the menu in the main view of the application . . 98
MenuUiStateObserver
An observer class responsible for updating the menu views when notified, i.e., when a new MenuUiState is produced by the subscribed Flowable 100
MoralDamageCalculatorDecorator
A DamageCalculator class responsible for calculating the moral buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability that are not the unit the damage is calculated for 103
OnValueChangeListener< T extends Comparable
An interface defining a callback for changes of the current value 106
OverlayDialog
A Dialog class which is used for popups that are shown on top of the calling Activity 107

RemoveUnitsUseCase
A use case class responsible for removing units from a UnitRepository 110

ResetAlertDialogBuilderAdapter
An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether he really wants to reset 112

ResetDialogUseCase
A use case class responsible for dispatching a reset call to ResetRepositoryUseCase, possibly after a confirmation by the user obtained from a Dialog 115

ResetRepositoryUseCase
A use case class responsible for resetting the UnitRepository 117

RevengeAlertDialogBuilderAdapter
An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether they want to invoke the Ability::REVENGE ability 120

RowDao
An interface defining data access operations related to `rows` table 124

RowEntity
A class representing the state of an attack row, i.e., weather and commanders horn 127

RowType
An enum listing the different types of attack rows 131

RowUiState
A data class encapsulating the visible state of a row in the main view of the application 132

RowUiStateObserver
An observer class responsible for updating the views of the row defined in #row when notified, i.e., when a new RowUiState is produced the observed Flowable 135

RuleActivity
An `AppCompatActivity` used to present the rules of the game 138

RuleSection
An `Enum` used to discern the different possible sections shown by `RuleActivity` 140

SectionsPagerAdapter
A `FragmentStateAdapter` that returns an `IntroductionFragment` corresponding to one of the sections/tabs/pages 142

SettingsActivity
An `AppCompatActivity` implementing `PreferenceFragmentCompat.OnPreferenceStartFragmentC` that is used by the user to manage the `SharedPreferences` of the application 143

SettingsHeaderFragment
A `PreferenceFragmentCompat` class encapsulating the main preference screen, i.e., the `Preference`s defined in `R.xml#header_preferences` 146

SettingsRuleFragment
A `PreferenceFragmentCompat` class encapsulating the rule preference screen, i.e., the rule sections defined in `R.xml#rule_preferences` 147

SettingsSoundFragment
A `PreferenceFragmentCompat` class encapsulating the sound preference screen, i.e., the `Preference`s defined in `R.xml#sound_preferences` 149

ShowUnitsDialog
An OverlayDialog used to list the units of a certain row and enabling the user to copy, add, or delete new units 151

Sound
A data class encapsulating information on a certain sound effect, e.g., its id in a SoundPool and whether it is activated in settings 155

SoundManager
A class responsible for initializing and playing sounds when they are enabled 158

SquadManager
A class used by the CardNumberPickerAdapter to determine the first squad with members or set the default damage according to the squads base damage 166

SquadState
A data class encapsulating information about the squad defined by #squadNumber, i.e., how many #squadMembers are there and what is the #memberBaseDamage of units in this squad 168

StringValuePicker< T extends Comparable
A ValuePicker displaying a resource String for each element in selectableValues 172
ResetDialogUseCase.Trigger
An Enum listing the possible triggers of a reset 172
UnitDao
An interface defining data access operations related to units table 173
UnitEntity
A class representing a card on the game board 177
UnitRepository
A facade class managing public access to the data layer 184
ValuePicker< T extends Comparable
A generic abstract wrapper class around NumberPicker used to select values of arbitrary types extending Comparable 195
WeatherDamageCalculator
A DamageCalculator class responsible for calculating the weather debuff if necessary, i.e., if #weather is true 195

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

/home/runner/work/Gwent/Gwent/app/src/main/AndroidManifest.xml 199
gwent/GwentApplication.java 217
gwent/data/Ability.java 199
gwent/data/AppDatabase.java 200
gwent/data/RowDao.java 200
gwent/data/RowEntity.java 200
gwent/data/RowType.java 201
gwent/data/UnitDao.java 201
gwent/data/UnitEntity.java 202
gwent/data/UnitRepository.java 204
gwent/domain/cases/BurnDialogUseCase.java 207
gwent/domain/cases/DamageCalculatorUseCase.java 208
gwent/domain/cases/RemoveUnitsUseCase.java 209
gwent/domain/cases/ResetAlertDialogBuilderAdapter.java 210
gwent/domain/cases/ResetDialogUseCase.java 210
gwent/domain/cases/ResetRepositoryUseCase.java 212
gwent/domain/cases/RevengeAlertDialogBuilderAdapter.java 213
gwent/domain/damage/BondDamageCalculatorDecorator.java 214
gwent/domain/damage/DamageCalculator.java 214
gwent/domain/damage/DamageCalculatorBuildDirector.java 214
gwent/domain/damage/DamageCalculatorBuilder.java 215
gwent/domain/damage/DamageCalculatorDecorator.java 216
gwent/domain/damage/HornDamageCalculatorDecorator.java 216
gwent/domain/damage/MoralDamageCalculatorDecorator.java 216
gwent/domain/damage/WeatherDamageCalculator.java 217
gwent/ui/dialogs/ChangeFactionDialog.java 231
gwent/ui/dialogs/CoinFlipDialog.java 232
gwent/ui/dialogs/OverlayDialog.java 232
gwent/ui/dialogs/addcard/AddCardDialog.java 218
gwent/ui/dialogs/addcard/CardNumberPickerAdapter.java 219
gwent/ui/dialogs/addcard/DamageValuePicker.java 220
gwent/ui/dialogs/addcard/SquadManager.java 221
gwent/ui/dialogs/addcard/SquadState.java 222
gwent/ui/dialogs/addcard/StringValuePicker.java 223
gwent/ui/dialogs/addcard/ValuePicker.java 224
gwent/ui/dialogs/cards/CardListAdapter.java 225
gwent/ui/dialogs/cards/CardUiState.java 227
gwent/ui/dialogs/cards/CardUiStateFactory.java 228
gwent/ui/dialogs/cards/ShowUnitsDialog.java 229
gwent/ui/introduction/IndicatorManager.java 233

gwent/ui/introduction/IntroductionActivity.java . . . 233
gwent/ui/introduction/IntroductionFragment.java . . . 234
gwent/ui/introduction/SectionsPagerAdapter.java . . . 235
gwent/ui/main/FactionSwitchListener.java . . . 235
gwent/ui/main/GameBoardViewModel.java . . . 237
gwent/ui/main/ImageViewSwitchAnimator.java . . . 239
gwent/ui/main/MainActivity.java . . . 240
gwent/ui/main/MenuUiState.java . . . 243
gwent/ui/main/MenuUiStateObserver.java . . . 243
gwent/ui/main/RowUiState.java . . . 244
gwent/ui/main/RowUiStateObserver.java . . . 245
gwent/ui/settings/RuleActivity.java . . . 246
gwent/ui/settings/RuleSection.java . . . 247
gwent/ui/settings/SettingsActivity.java . . . 247
gwent/ui/settings/SettingsHeaderFragment.java . . . 248
gwent/ui/settings/SettingsRuleFragment.java . . . 248
gwent/ui/settings/SettingsSoundFragment.java . . . 249
gwent/ui/sounds/Sound.java . . . 250
gwent/ui/sounds/SoundManager.java . . . 250
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/background_drawable.xml . . . 252
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_copy.xml . . . 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_delete.xml . . . 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_design.xml . . . 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_explore.xml . . . 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_factionreset.xml . . . 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_next.xml . . . 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_rules.xml . . . 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_sounds.xml . . . 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_warnings.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/indicator_selected.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/indicator_unselected.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle_black_transparent.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle_cancel_popup.xml . . . 255
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-land-v26/fragment_introduction.xml . . . 256
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-land/fragment_introduction.xml . . . 256
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-v26/activity_rules.xml . . . 258
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-v26/fragment_introduction.xml . . . 257
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_introduction.xml . . . 259
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_main.xml . . . 260
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_rules.xml . . . 258
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_settings.xml . . . 262
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/alertdialog_checkbox.xml . . . 263
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/card.xml . . . 263
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/fragment_introduction.xml . . . 257
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_add_card.xml . . . 264
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_cards.xml . . . 266
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_normal.xml . . . 267
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_stewie.xml . . . 267
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_terry.xml . . . 269
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_vin.xml . . . 270
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_faction.xml . . . 270
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/row.xml . . . 272
/home/runner/work/Gwent/Gwent/app/src/main/res/mipmap-anydpi-v26/icon_launcher.xml . . . 273
/home/runner/work/Gwent/Gwent/app/src/main/res/mipmap-anydpi-v26/icon_launcher_round.xml . . . 273
/home/runner/work/Gwent/Gwent/app/src/main/res/values-de-rDE/strings.xml . . . 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/arrays.xml . . . 273
/home/runner/work/Gwent/Gwent/app/src/main/res/values/attr.xml . . . 274

/home/runner/work/Gwent/Gwent/app/src/main/res/values/colors.xml 274
/home/runner/work/Gwent/Gwent/app/src/main/res/values/dimens.xml 274
/home/runner/work/Gwent/Gwent/app/src/main/res/values/icon_launcher_background.xml 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/preferences.xml 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/strings.xml 279
/home/runner/work/Gwent/Gwent/app/src/main/res/values/styles.xml 283
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/header_preferences.xml 284
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/rule_preferences.xml 285
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/sound_preferences.xml 285

Chapter 5

Class Documentation

5.1 Ability Enum Reference

An enum listing the different card abilities possible.
Collaboration diagram for Ability:

Public Attributes

- NONE

  Represents a card that has no ability.
- HORN

  Represents a card that has the Commanders Horn ability, e.g.
- REVENGE

  Represents a card that has the Summon Avenger ability.
- BINDING

  Represents a card that has the Tight Bond ability.
- MORAL_BOOST

  Represents a card that has the Morale Boost ability.

5.1.1 Detailed Description

An enum listing the different card abilities possible.
Definition at line 6 of file Ability.java.

5.1.2 Member Data Documentation

5.1.2.1 BINDING

```
BINDING
```

Represents a card that has the Tight Bond ability.
Definition at line 25 of file Ability.java.
Referenced by UnitEntity.UnitEntity(), UnitRepository.insertUnit(), and UnitEntity.setSquad().

5.1.2.2 HORN

HORN
Represents a card that has the Commanders Horn ability, e.g.
Dandelion.
Definition at line 15 of file Ability.java.

5.1.2.3 MORAL_BOOST

MORAL_BOOST
Represents a card that has the Morale Boost ability.
Definition at line 31 of file Ability.java.

5.1.2.4 NONE

NONE
Represents a card that has no ability.
Definition at line 10 of file Ability.java.

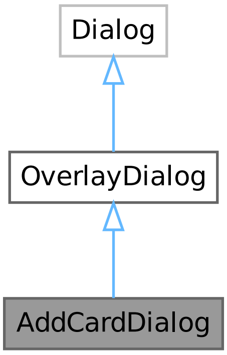
5.1.2.5 REVENGE

REVENGE
Represents a card that has the Summon Avenger ability.
Definition at line 20 of file Ability.java.
The documentation for this enum was generated from the following file:

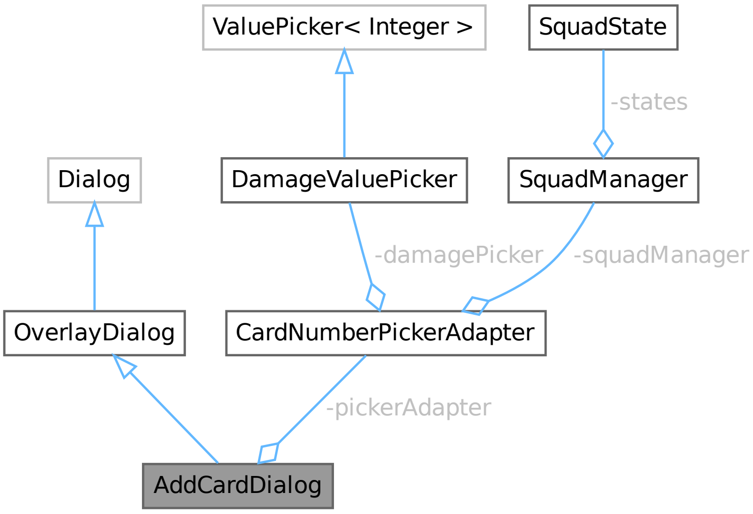
- gwent/data/Ability.java

5.2 AddCardDialog Class Reference

An OverlayDialog used to add new UnitEntity objects to a certain #row.
Inheritance diagram for AddCardDialog:



Collaboration diagram for AddCardDialog:



Public Member Functions

- AddCardDialog (@NonNull ShowUnitsDialog caller, @NonNull SoundManager soundManager)

  Constructor of an AddCardDialog called by the given ShowUnitsDialog using the given SoundManager.
- AddCardDialog (@NonNull Context context, @NonNull RowType row, @NonNull SoundManager soundManager)

  Constructor of an AddCardDialog in the given Context and for the given row.

Protected Member Functions

- void onCreate (Bundle savedInstanceState)

  Initializes the #pickerAdapter if not yet done so and sets View.OnClickListener for each button.

Protected Member Functions inherited from OverlayDialog

- OverlayDialog (@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)

  Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
- OverlayDialog (@NonNull Context context, @LayoutRes int layout)

  Constructor of an OverlayDialog in the given Context and with the given layout.

Private Attributes

- final RowType row

  RowType defining the row the new UnitEntity objects are added to.
- final SoundManager soundManager

  SoundManager used to play a Sound whenever a card is added.
- final CompositeDisposable disposables = new CompositeDisposable()

  CompositeDisposable keeping track of all subscriptions to observables made by this class.
- CardNumberPickerAdapter pickerAdapter = null

  CardNumberPickerAdapter managing the connection to the NumberPicker views of this Dialog.

Additional Inherited Members

Static Package Attributes inherited from OverlayDialog

- static final int NO_DISMISS_VIEW = -1

  Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

5.2.1 Detailed Description

An OverlayDialog used to add new UnitEntity objects to a certain #row.
Definition at line 22 of file AddCardDialog.java.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 AddCardDialog() [1/2]

```
AddCardDialog (
            @NonNull ShowUnitsDialog caller,
            @NonNull SoundManager soundManager )
```

Constructor of an AddCardDialog called by the given ShowUnitsDialog using the given SoundManager.
Sets a Dialog.OnDismissListener disposing #disposables and showing the caller again. Wrapper of #AddCardDialog(Context, RowType, SoundManager).

See also

#AddCardDialog(Context, RowType, SoundManager)

Parameters

caller	ShowUnitsDialog that called this Dialog.
soundManager	SoundManager used to play a Sound when cards are added.

Definition at line 67 of file AddCardDialog.java.
References AddCardDialog.disposables, and AddCardDialog.soundManager.

5.2.2.2 AddCardDialog() [2/2]

```
AddCardDialog (
            @NonNull Context context,
            @NonNull RowType row,
            @NonNull SoundManager soundManager )
```

Constructor of an AddCardDialog in the given Context and for the given row.
Sets a Dialog.OnDismissListener disposing #disposables and showing the caller again.

Parameters

context	Context this Dialog is shown in.
row	RowType defining which row the new UnitEntity objects are added to.
soundManager	SoundManager used to play a Sound when cards are added.

Definition at line 84 of file AddCardDialog.java.
References AddCardDialog.disposables, AddCardDialog.row, and AddCardDialog.soundManager.

5.2.3 Member Function Documentation

5.2.3.1 onCreate()

```
void onCreate (
            Bundle savedInstanceState )  [protected]
```

Initializes the #pickerAdapter if not yet done so and sets View.OnClickListener for each button.

Parameters

savedInstanceState	If this dialog is being reinitialized after the hosting activity was previously shut down, holds the result from the most recent call to `onSaveInstanceState`, or null if this is the first time.

Reimplemented from OverlayDialog.
Definition at line 102 of file AddCardDialog.java.
References CardNumberPickerAdapter.addSelectedUnits(), AddCardDialog.disposables, GwentApplication.getRepository(), AddCardDialog.pickerAdapter, and AddCardDialog.row.

5.2.4 Member Data Documentation

5.2.4.1 disposables

```
final CompositeDisposable disposables = new CompositeDisposable()  [private]
```

CompositeDisposable keeping track of all subscriptions to observables made by this class.
Is being disposed in an android.content.DialogInterface.OnDismissListener that is set in #AddCardDialog(Show↩ UnitsDialog) and #AddCardDialog(Context, RowType).

See also

android.content.DialogInterface.OnDismissListener

#AddCardDialog(ShowUnitsDialog, SoundManager)

#AddCardDialog(Context, RowType, SoundManager)

Definition at line 47 of file AddCardDialog.java.
Referenced by AddCardDialog.AddCardDialog(), AddCardDialog.AddCardDialog(), and AddCardDialog.onCreate().

5.2.4.2 pickerAdapter

```
CardNumberPickerAdapter pickerAdapter = null  [private]
```

CardNumberPickerAdapter managing the connection to the NumberPicker views of this Dialog.
Is lazily initialized in #onCreate().

See also

#onCreate(Bundle)

R.id::card_layout

Definition at line 57 of file AddCardDialog.java.
Referenced by AddCardDialog.onCreate().

5.2.4.3 row

```
final RowType row  [private]
```

RowType defining the row the new UnitEntity objects are added to.
Definition at line 27 of file AddCardDialog.java.
Referenced by AddCardDialog.AddCardDialog(), and AddCardDialog.onCreate().

5.2.4.4 soundManager

```
final SoundManager soundManager  [private]
```

SoundManager used to play a Sound whenever a card is added.

See also

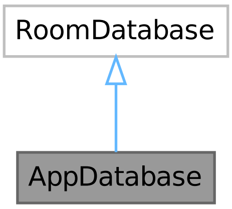
SoundManager::playCardAddSound(RowType, boolean)

Definition at line 35 of file AddCardDialog.java.
Referenced by AddCardDialog.AddCardDialog(), and AddCardDialog.AddCardDialog().
The documentation for this class was generated from the following file:

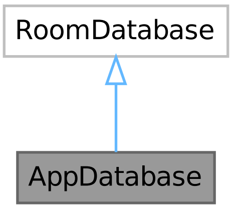
- gwent/ui/dialogs/addcard/AddCardDialog.java

5.3 AppDatabase Class Reference

An abstract class extending RoomDatabase and representing the SQLite database of the application.
Inheritance diagram for AppDatabase:



Collaboration diagram for AppDatabase:



Package Functions

- abstract UnitDao units ()

  Returns an object implementing the operations defined in UnitDao.
- abstract RowDao rows ()

  Returns an object implementing the operations defined in RowDao.

5.3.1 Detailed Description

An abstract class extending RoomDatabase and representing the SQLite database of the application.
The implementation is generated by the Room framework.
Definition at line 11 of file AppDatabase.java.

5.3.2 Member Function Documentation

5.3.2.1 rows()

`abstract RowDao rows ( ) [abstract], [package]`

Returns an object implementing the operations defined in RowDao.

Returns

RowDao implementing DAO for `rows` table.

Referenced by UnitRepository.clearWeather(), UnitRepository.initializeRows(), UnitRepository.isHorn(), UnitRepository.isHornFlowable
UnitRepository.isWeather(), UnitRepository.isWeatherFlowable(), UnitRepository.reset(), UnitRepository.switchHorn(), and UnitRepository.switchWeather().

5.3.2.2 units()

`abstract UnitDao units ( ) [abstract], [package]`

Returns an object implementing the operations defined in UnitDao.

Returns

UnitDao implementing DAO for `units` table.

Referenced by UnitRepository.countUnits(), UnitRepository.countUnits(), UnitRepository.delete(), UnitRepository.delete(), UnitRepository.getUnit(), UnitRepository.getUnits(), UnitRepository.getUnits(), UnitRepository.getUnitsFlowable(), UnitRepository.hasNonEpicUnitsFlowable(), UnitRepository.insertUnit(), and UnitRepository.reset().

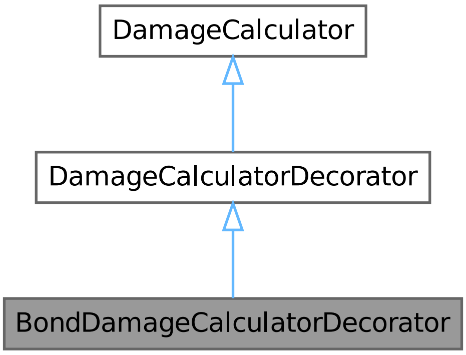
The documentation for this class was generated from the following file:

- gwent/data/AppDatabase.java

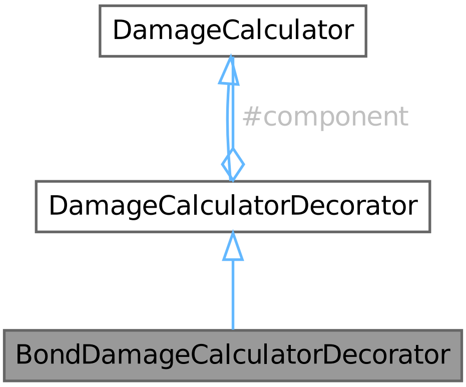
5.4 BondDamageCalculatorDecorator Class Reference

A DamageCalculator class responsible for calculating the tight bond buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::BINDING ability.

Inheritance diagram for BondDamageCalculatorDecorator:



Collaboration diagram for BondDamageCalculatorDecorator:



Public Member Functions

- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.

Package Functions

- BondDamageCalculatorDecorator (@NonNull DamageCalculator component, @NonNull Map< Integer, Integer > idToSquadSize)

  Constructor of a BondDamageCalculatorDecorator.

Package Functions inherited from DamageCalculatorDecorator

- DamageCalculatorDecorator (@NonNull DamageCalculator component)

  Constructor of a DamageCalculatorDecorator.

Private Attributes

- final Map< Integer, Integer > idToSquadSize

  A Map mapping the ids of all units with the com.peternaggschga.gwent.data.Ability::BINDING ability to the respective squad size.

Additional Inherited Members

Protected Attributes inherited from DamageCalculatorDecorator

- final DamageCalculator component

  A DamageCalculator which is decorated by this decorator.

5.4.1 Detailed Description

A DamageCalculator class responsible for calculating the tight bond buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::BINDING ability.

Acts as a ConcreteDecorator in the implemented decorator pattern and should decorate a WeatherDamageCalculator for correct calculation.
Definition at line 17 of file BondDamageCalculatorDecorator.java.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 BondDamageCalculatorDecorator()

```
BondDamageCalculatorDecorator (
             @NonNull DamageCalculator component,
             @NonNull Map< Integer, Integer > idToSquadSize )  [package]
```

Constructor of a BondDamageCalculatorDecorator.
Should only be called by DamageCalculatorBuilder. The given damage calculator should be of type WeatherDamageCalculator for correct damage calculation. The values of the given Map must contain non-`null` or positive Integers.

Parameters

component	DamageCalculator that is being decorated by this decorator.
idToSquadSize	Map mapping the ids of all units with the com.peternaggschga.gwent.data.Ability::BINDING ability to the respective squad size.

Exceptions

IllegalArgumentException	When idToSquadSize contains non-positive or null values.

See also

DamageCalculatorBuilder

Definition at line 35 of file BondDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, and BondDamageCalculatorDecorator.idToSquadSize.

5.4.3 Member Function Documentation

5.4.3.1 calculateDamage()

```
int calculateDamage (
             int id,
             @IntRange(from=0) int damage )
```

Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
Returns the given damage times the number of units with the same squad if the unit has the com.peternaggschga.gwent.data.Ability::B
ability.

Parameters

id	Integer representing the UnitEntity::id of the unit whose (de-)buff damage is calculated.
damage	Integer representing the base-damage of the unit whose (de-)buff damage is calculated.

Returns

Integer representing the (de-)buffed damage of the unit.

Exceptions

IllegalArgumentException	When damage is negative.

Implements DamageCalculator.
Definition at line 54 of file BondDamageCalculatorDecorator.java.
References DamageCalculator.calculateDamage(), DamageCalculatorDecorator.component, and BondDamageCalculatorDecorator.i

5.4.3.2 isBuffed()

```
Color isBuffed (
            int id )
```

Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT. Units are shown as Color::BUFFED when they are in a squad of two or more units, otherwise their Color is defined by #component.

Parameters

id	Integer representing the UnitEntity::id of the unit buff status is calculated.

Returns

Color representing whether the unit is buffed, de-buffed or not affected.

See also

Color

Implements DamageCalculator.
Definition at line 72 of file BondDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, BondDamageCalculatorDecorator.idToSquadSize, and DamageCalculator.isBuffed().

5.4.4 Member Data Documentation

5.4.4.1 idToSquadSize

```
final Map<Integer, Integer> idToSquadSize [private]
```

A Map mapping the ids of all units with the com.peternaggschga.gwent.data.Ability::BINDING ability to the respective squad size.
Definition at line 22 of file BondDamageCalculatorDecorator.java.
Referenced by BondDamageCalculatorDecorator.BondDamageCalculatorDecorator(), BondDamageCalculatorDecorator.calculateDa
and BondDamageCalculatorDecorator.isBuffed().
The documentation for this class was generated from the following file:

- gwent/domain/damage/BondDamageCalculatorDecorator.java

5.5 BurnDialogUseCase Class Reference

A use case class responsible for dispatching a remove call to RemoveUnitsUseCase.
Collaboration diagram for BurnDialogUseCase:

Static Public Member Functions

- static Single< Boolean > burn (@NonNull Context context, @NonNull SoundManager soundManager)

  Burns the strongest UnitEntity objects in UnitRepository.

Static Protected Member Functions

- static Single< Boolean > burn (@NonNull Context context, @NonNull UnitRepository repository, @NonNull SoundManager soundManager)

  Burns the strongest UnitEntity objects in UnitRepository.

Static Private Member Functions

- static Single< List< UnitEntity > > getBurnUnits (@NonNull UnitRepository repository)

  Returns the list of units that would be affected by a burn operation.

5.5.1 Detailed Description

A use case class responsible for dispatching a remove call to RemoveUnitsUseCase.
Definition at line 28 of file BurnDialogUseCase.java.

5.5.2 Member Function Documentation

5.5.2.1 burn() [1/2]

```
static Single< Boolean > burn (
              @NonNull Context context,
              @NonNull SoundManager soundManager )  [static]
```

Burns the strongest UnitEntity objects in UnitRepository.
Invokes a Dialog asking whether the user really wants to remove those units. ResetRepositoryUseCase is used for resetting. Wrapper for #burn(Context, UnitRepository).

Parameters

context	Context where a Dialog can be inflated.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Single emitting a Boolean defining whether the units really were burned.

See also

#burn(Context, UnitRepository, SoundManager)

RemoveUnitsUseCase::remove(Context, UnitRepository, Collection, SoundManager)

Definition at line 85 of file BurnDialogUseCase.java.
References BurnDialogUseCase.burn(), and GwentApplication.getRepository().
Referenced by BurnDialogUseCase.burn().

5.5.2.2 burn() [2/2]

```
static Single< Boolean > burn (
              @NonNull Context context,
              @NonNull UnitRepository repository,
              @NonNull SoundManager soundManager )  [static], [protected]
```

Burns the strongest UnitEntity objects in UnitRepository.
Invokes a Dialog asking whether the user really wants to remove those units. ResetRepositoryUseCase is used for resetting.

Parameters

context	Context where a Dialog can be inflated.
repository	UnitRepository where units are burned.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

> A Single emitting a Boolean defining whether the units really were burned.

See also

> RemoveUnitsUseCase::remove(Context, UnitRepository, Collection, SoundManager)

Definition at line 101 of file BurnDialogUseCase.java.
References BurnDialogUseCase.getBurnUnits().

5.5.2.3 getBurnUnits()

```
static Single< List< UnitEntity > > getBurnUnits (
            @NonNull UnitRepository repository )  [static], [private]
```

Returns the list of units that would be affected by a burn operation.
The returned list may be empty.

Parameters

repository	UnitRepository where units are fetched.

Returns

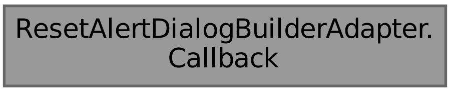
> A Single emitting the List of UnitEntity objects that would be affected by the operation.

Definition at line 37 of file BurnDialogUseCase.java.
References DamageCalculatorUseCase.getDamageCalculator().
Referenced by BurnDialogUseCase.burn().
The documentation for this class was generated from the following file:

- gwent/domain/cases/BurnDialogUseCase.java

5.6 ResetAlertDialogBuilderAdapter.Callback Interface Reference

An interface defining functions to propagate the user's decision back to the creator.
Collaboration diagram for ResetAlertDialogBuilderAdapter.Callback:



Public Member Functions

- void reset (boolean resetDecision, boolean keepUnit)

Called when the user makes a decision.

- default void reset (boolean resetDecision)

  Wrapper for #reset(boolean, boolean).

5.6.1 Detailed Description

An interface defining functions to propagate the user's decision back to the creator.
Definition at line 98 of file ResetAlertDialogBuilderAdapter.java.

5.6.2 Member Function Documentation

5.6.2.1 reset() [1/2]

```
default void reset (
              boolean resetDecision )
```

Wrapper for #reset(boolean, boolean).
Can be used when the AlertDialog is not a monster dialog since it calls #reset(boolean, boolean) without keeping a unit.

Parameters

resetDecision	Boolean defining whether the user has confirmed the reset.

See also

#reset(boolean, boolean)

Definition at line 117 of file ResetAlertDialogBuilderAdapter.java.
References ResetAlertDialogBuilderAdapter.Callback.reset().

5.6.2.2 reset() [2/2]

```
void reset (
              boolean resetDecision,
              boolean keepUnit )
```

Called when the user makes a decision.
When the AlertDialog is a monster dialog, #reset(boolean) can be used.

Parameters

resetDecision	Boolean defining whether the user has confirmed the reset.
keepUnit	Boolean defining whether a random unit should be kept when resetting.

See also

#reset(boolean)

Referenced by ResetAlertDialogBuilderAdapter.ResetAlertDialogBuilderAdapter(), ResetAlertDialogBuilderAdapter.Callback.reset(), and ResetAlertDialogBuilderAdapter.setMonsterDialog().
The documentation for this interface was generated from the following file:

- gwent/domain/cases/ResetAlertDialogBuilderAdapter.java

5.7 ChangeFactionDialog.Callback Interface Reference

An interface used for propagating a selected theme back to the creator of a ChangeFactionDialog.

Collaboration diagram for ChangeFactionDialog.Callback:

Public Member Functions

- void onThemeSelect (@IntRange(from=THEME_MONSTER, to=THEME_SCOIATAEL) int theme)

  Callback being called when a theme is selected in the respective ChangeFactionDialog.

5.7.1 Detailed Description

An interface used for propagating a selected theme back to the creator of a ChangeFactionDialog.
Definition at line 84 of file ChangeFactionDialog.java.

5.7.2 Member Function Documentation

5.7.2.1 onThemeSelect()

```
void onThemeSelect (
             @IntRange(from=THEME_MONSTER, to=THEME_SCOIATAEL) int theme )
```

Callback being called when a theme is selected in the respective ChangeFactionDialog.

Parameters

theme	Integer representing the selected theme.

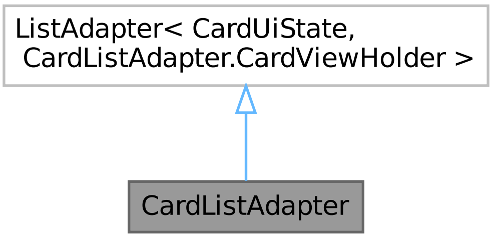
Referenced by ChangeFactionDialog.getOnThemeClickListener().
The documentation for this interface was generated from the following file:

- gwent/ui/dialogs/ChangeFactionDialog.java

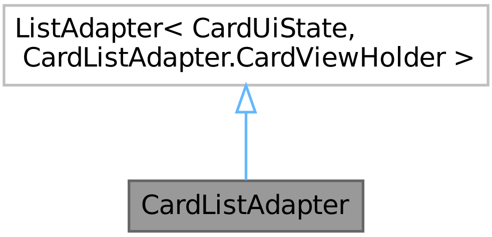
5.8 CardListAdapter Class Reference

A ListAdapter used to provide CardListAdapter.CardViewHolder objects created from CardUiState objects to a RecyclerView.

Inheritance diagram for CardListAdapter:



Collaboration diagram for CardListAdapter:



Classes

- class CardViewHolder

  A RecyclerView.ViewHolder class managing a card view that shows a representation of the UnitEntity referenced by #itemId.

Public Member Functions

- CardViewHolder onCreateViewHolder (@NonNull ViewGroup parent, int viewType)

  Creates a new CardViewHolder managing a new card view that is inflated from the given parent.
- void onBindViewHolder (@NonNull CardViewHolder holder, @IntRange(from=0) int position)

  Changes the views of CardViewHolder to represent the item at the given position.
- long getItemId (@IntRange(from=0) int position)

  Returns the UnitEntity::id of the UnitEntity represented at the given position.

Package Functions

- CardListAdapter (@NonNull Consumer< Integer > onCopy, @NonNull Consumer< Integer > onRemove)

  Constructor of a CardListAdapter with the given #onCopy and #onRemove callbacks.

Private Attributes

- final Consumer< Integer > onCopy

  Consumer that is used in #onCreateViewHolder(ViewGroup, int) as CardListAdapter.CardViewHolder::onCopy.
- final Consumer< Integer > onRemove

  Consumer that is used in #onCreateViewHolder(ViewGroup, int) as CardListAdapter.CardViewHolder::onRemove.

5.8.1 Detailed Description

A ListAdapter used to provide CardListAdapter.CardViewHolder objects created from CardUiState objects to a RecyclerView.

See also

CardUiState

CardListAdapter.CardViewHolder

ListAdapter

RecyclerView

Definition at line 28 of file CardListAdapter.java.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 CardListAdapter()

```
CardListAdapter (
             @NonNull Consumer< Integer > onCopy,
             @NonNull Consumer< Integer > onRemove )  [package]
```

Constructor of a CardListAdapter with the given #onCopy and #onRemove callbacks.
Calls super-constructor ListAdapter::ListAdapter(DiffUtil.ItemCallback) with CardUiState::DIFF_CALLBACK. Also calls #setHasStableIds() since #getItemId() returns the stable UnitEntity::id.

See also

ListAdapter::ListAdapter(DiffUtil.ItemCallback)

CardUiState::DIFF_CALLBACK

#setHasStableIds(boolean)

Parameters

onCopy	Consumer that is called with the UnitEntity::id, when the copy-button of the representing card is clicked.
onRemove	Consumer that is called with the UnitEntity::id, when the delete-button of the representing card is clicked.

Definition at line 65 of file CardListAdapter.java.
References CardUiState.DIFF_CALLBACK, CardListAdapter.onCopy, and CardListAdapter.onRemove.

5.8.3 Member Function Documentation

5.8.3.1 getItemId()

```
long getItemId (
             @IntRange(from=0) int position )
```

Returns the UnitEntity::id of the UnitEntity represented at the given position.

Parameters

position	Adapter position to query.

Returns

A Long referencing the UnitEntity::id of the UnitEntity represented at the given position.

Definition at line 122 of file CardListAdapter.java.

5.8.3.2 onBindViewHolder()

```
void onBindViewHolder (
            @NonNull CardViewHolder holder,
            @IntRange(from=0) int position )
```

Changes the views of CardViewHolder to represent the item at the given position.

Parameters

holder	The CardViewHolder which should be updated to represent the contents of the item at the given position in the data set.
position	The position of the item within the adapter's data set.

Definition at line 93 of file CardListAdapter.java.
References CardUiState.getAbilityImageId(), CardUiState.getDamageBackgroundImageId(), CardUiState.getDamageString(), CardUiState.getDamageTextColor(), CardUiState.getSquadString(), CardUiState.getUnitId(), CardUiState.showAbility(), and CardUiState.showSquad().

5.8.3.3 onCreateViewHolder()

```
CardViewHolder onCreateViewHolder (
            @NonNull ViewGroup parent,
            int viewType )
```

Creates a new CardViewHolder managing a new card view that is inflated from the given parent.

Parameters

parent	The ViewGroup into which the new View will be added after it is bound to an adapter position.
viewType	The view type of the new View.

Returns

A CardViewHolder object for a new card layout and with #onCopy and #onRemove callbacks.

Definition at line 81 of file CardListAdapter.java.
References CardListAdapter.onCopy, and CardListAdapter.onRemove.

5.8.4 Member Data Documentation

5.8.4.1 onCopy

```
final Consumer<Integer> onCopy [private]
```

Consumer that is used in #onCreateViewHolder(ViewGroup, int) as CardListAdapter.CardViewHolder::onCopy.
Is called when the user clicks on the copy-button of an element with the UnitEntity::id of the represented UnitEntity.

See also

#onCreateViewHolder(ViewGroup, int)

CardListAdapter.CardViewHolder::onCopy

Definition at line 39 of file CardListAdapter.java.
Referenced by CardListAdapter.CardListAdapter(), CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.onCreateViewHolder().

5.8.4.2 onRemove

`final Consumer<Integer> onRemove [private]`

Consumer that is used in #onCreateViewHolder(ViewGroup, int) as CardListAdapter.CardViewHolder::onRemove. Is called when the user clicks on the delete-button of an element with the UnitEntity::id of the represented UnitEntity.

See also

#onCreateViewHolder(ViewGroup, int)

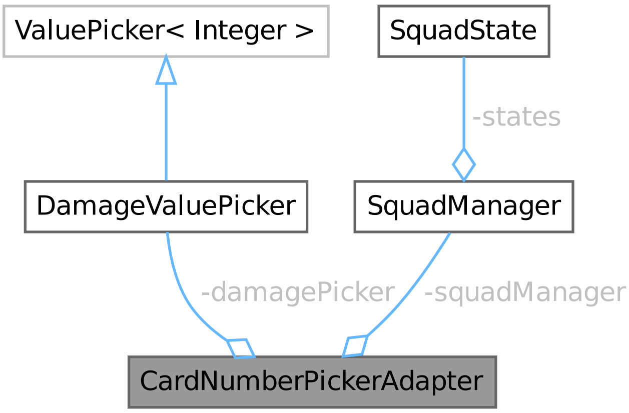
CardListAdapter.CardViewHolder::onRemove

Definition at line 50 of file CardListAdapter.java.
Referenced by CardListAdapter.CardListAdapter(), CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.onCreateViewHolder().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/cards/CardListAdapter.java

5.9 CardNumberPickerAdapter Class Reference

A helper class responsible for initializing the NumberPicker views of an AddCardDialog (in #CardNumperPickerAdapter()) and adding UnitEntity objects with the selected attributes when #addSelectedUnits() is called.
Collaboration diagram for CardNumberPickerAdapter:



Package Functions

- CardNumberPickerAdapter (@NonNull ViewGroup pickerGroup, @NonNull SquadManager squadManager)

    Constructor of a CardNumberPickerAdapter managing the NumberPicker views in the given ViewGroup using the given SquadManager.
- Single< Boolean > addSelectedUnits (@NonNull RowType row)

    Adds new UnitEntity objects the attributes selected by the managed pickers.

Static Package Functions

- static NumberPicker.OnValueChangeListener getDelayedOnValueChangeListener (@NonNull NumberPicker.OnValueChangeListener originalListener)

    Creates a NumberPicker.OnValueChangeListener that only executes the given NumberPicker.OnValueChangeListener after 500 ms if the value has not changed.

Private Attributes

- final ValuePicker< Boolean > epicPicker

  ValuePicker used to decide the value of UnitEntity::epic.
- final DamageValuePicker damagePicker

  DamageValuePicker used to decide the value of UnitEntity::damage.
- final ValuePicker< Ability > abilityPicker

  ValuePicker used to decide the value of UnitEntity::ability.
- final NumberPicker squadPicker

  NumberPicker used to decide the value of UnitEntity::squad.
- final SquadManager squadManager

  SquadManager containing current SquadState used to get squad information.
- final NumberPicker numberPicker

  NumberPicker used to decide the number of UnitEntity objects that are inserted.

Static Private Attributes

- static final List< Ability > EPIC_UNIT_ABILITIES = Arrays.asList(Ability.NONE, Ability.HORN, Ability.MORAL_BOOST)

  List of Ability values that epic units can have.

5.9.1 Detailed Description

A helper class responsible for initializing the NumberPicker views of an AddCardDialog (in #CardNumperPickerAdapter()) and adding UnitEntity objects with the selected attributes when #addSelectedUnits() is called.

See also

AddCardDialog

Definition at line 33 of file CardNumberPickerAdapter.java.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 CardNumberPickerAdapter()

```
CardNumberPickerAdapter (
            @NonNull ViewGroup pickerGroup,
            @NonNull SquadManager squadManager )  [package]
```

Constructor of a CardNumberPickerAdapter managing the NumberPicker views in the given ViewGroup using the given SquadManager.

Sets value bounds and NumberPicker.OnValueChangedListener for the pickers in the ViewGroup. The ViewGroup must be the ConstraintLayout with the id R.id::card_layout from popup_add_card.xml.

See also

R.id::card_layout

Parameters

pickerGroup	ViewGroup containing the managed NumberPicker views.
squadManager	SquadManager containing up-to-date SquadState.

Definition at line 89 of file CardNumberPickerAdapter.java.

References CardNumberPickerAdapter.abilityPicker, CardNumberPickerAdapter.damagePicker, CardNumberPickerAdapter.EPIC_UN
CardNumberPickerAdapter.epicPicker, CardNumberPickerAdapter.getDelayedOnValueChangeListener(), SquadManager.getFirstSqu
SquadManager.MAX_NR_SQUADS, CardNumberPickerAdapter.numberPicker, SquadManager.onSquadChanged(),
DamageValuePicker.setEpicValues(), CardNumberPickerAdapter.squadManager, and CardNumberPickerAdapter.squadPicker.

5.9.3 Member Function Documentation

5.9.3.1 addSelectedUnits()

```
Single< Boolean > addSelectedUnits (
            @NonNull RowType row )  [package]
```

Adds new UnitEntity objects the attributes selected by the managed pickers.

Parameters

row	RowType defining to which row the units are added.

Returns

A Single tracking operation status and containing whether the added unit is epic or not.

Definition at line 166 of file CardNumberPickerAdapter.java.
References CardNumberPickerAdapter.abilityPicker, CardNumberPickerAdapter.damagePicker, CardNumberPickerAdapter.epicPicke SquadManager.getFirstSquadWithMembers(), GwentApplication.getRepository(), DamageValuePicker.getValue(), CardNumberPickerAdapter.numberPicker, CardNumberPickerAdapter.squadManager, and CardNumberPickerAdapter.squadPicker.
Referenced by AddCardDialog.onCreate().

5.9.3.2 getDelayedOnValueChangeListener()

```
static NumberPicker.OnValueChangeListener getDelayedOnValueChangeListener (
            @NonNull NumberPicker.OnValueChangeListener originalListener )  [static], [package]
```

Creates a NumberPicker.OnValueChangeListener that only executes the given NumberPicker.OnValueChange↩ Listener after 500 ms if the value has not changed.

See also

NumberPicker.OnValueChangeListener

Parameters

originalListener	NumberPicker.OnValueChangeListener that is called when the value does not change.

Returns

A NumberPicker.OnValueChangeListener with delayed execution.

Definition at line 152 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter().

5.9.4 Member Data Documentation

5.9.4.1 abilityPicker

```
final ValuePicker<Ability> abilityPicker  [private]
```

ValuePicker used to decide the value of UnitEntity::ability.

See also

#squadPicker

Definition at line 58 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.9.4.2 damagePicker

```
final DamageValuePicker damagePicker  [private]
```

DamageValuePicker used to decide the value of UnitEntity::damage.
Definition at line 50 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.9.4.3 EPIC_UNIT_ABILITIES

```
final List<Ability> EPIC_UNIT_ABILITIES = Arrays.asList(Ability.NONE, Ability.HORN, Ability.↩
MORAL_BOOST)  [static], [private]
```

List of Ability values that epic units can have.
Definition at line 38 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter().

5.9.4.4 epicPicker

```
final ValuePicker<Boolean> epicPicker  [private]
```

ValuePicker used to decide the value of UnitEntity::epic.
Definition at line 44 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.9.4.5 numberPicker

```
final NumberPicker numberPicker  [private]
```

NumberPicker used to decide the number of UnitEntity objects that are inserted.
Definition at line 78 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.9.4.6 squadManager

```
final SquadManager squadManager  [private]
```

SquadManager containing current SquadState used to get squad information.
Definition at line 72 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.9.4.7 squadPicker

```
final NumberPicker squadPicker  [private]
```

NumberPicker used to decide the value of UnitEntity::squad.
Only visible if the value of #abilityPicker is set to Ability::BINDING.

See also

#abilityPicker

Definition at line 66 of file CardNumberPickerAdapter.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/CardNumberPickerAdapter.java

5.10 CardUiState Class Reference

A data class encapsulating the visible state of a card in the card list shown by the ShowUnitsDialog.

Collaboration diagram for CardUiState:

Public Member Functions

- CardUiState (int unitId, @DrawableRes int damageBackgroundImageId, @IntRange(from=UNUSED) int damage, @ColorInt int damageTextColor, @DrawableRes int abilityImageId, @Nullable @IntRange(from=1) Integer squad)

    Constructor of a CardUiState encapsulating the given data.
- boolean showAbility ()

    Returns whether the ability view is shown, i.e., if the unit has an ability other than Ability::NONE.
- boolean showSquad ()

    Returns whether the squad view is shown, i.e., if the unit has the Ability::BINDING ability.
- int getUnitId ()

    Returns the #unitId of the represented UnitEntity.
- int getDamageBackgroundImageId ()

    Returns the drawable resource shown as the background of the damage view.
- String getDamageString ()

    Returns the String shown in the damage view.
- int getDamageTextColor ()

    Returns the text color of the damage view.
- int getAbilityImageId ()

    Returns the drawable resource shown in the ability view.
- String getSquadString ()

    Returns the String shown in the squad view.
- boolean equals (Object o)

    Checks whether the given Object looks the same as this CardUiState.

Static Public Attributes

- static final DiffUtil.ItemCallback< CardUiState > DIFF_CALLBACK

    DiffUtil.ItemCallback used to compare different CardUiState objects in a ListAdapter, e.g., CardListAdapter.
- static final int UNUSED = -1

    Integer that is used instead of a drawable resource to indicate the absence of a meaningful #abilityImage.

Private Attributes

- final int unitId

    Integer referencing the UnitEntity::id of the represented UnitEntity.
- final int damageBackgroundImageId

    Integer referencing the drawable resource shown as the background of the damage view.
- final String damageString

    String containing the number shown in the damage view or nothing if the represented UnitEntity is epic.
- final int damageTextColor

    Integer representing the text color of the damage view.

- final int abilityImageId

  Integer referencing the drawable resource shown in the ability view.
- final String squadString

  String containing the number shown in the squad view.

5.10.1 Detailed Description

A data class encapsulating the visible state of a card in the card list shown by the ShowUnitsDialog.
Can be created from the represented UnitEntity using CardUiStateFactory.

See also

CardUiStateFactory

ShowUnitsDialog

Definition at line 24 of file CardUiState.java.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 CardUiState()

```
CardUiState (
                int unitId,
                @DrawableRes int damageBackgroundImageId,
                @IntRange(from=UNUSED) int damage,
                @ColorInt int damageTextColor,
                @DrawableRes int abilityImageId,
                @Nullable @IntRange(from=1) Integer squad )
```

Constructor of a CardUiState encapsulating the given data.

Parameters

unitId	Integer representing the UnitEntity::id of the represented UnitEntity.
damageBackground←↩ ImageId	Integer referencing the drawable resource shown by the damage view.
damage	Integer representing the damage of the represented UnitEntity.
damageTextColor	Integer representing the text color of the damage view.
abilityImageId	Integer referencing the drawable resource shown by the ability image view or #UNUSED.
squad	Integer representing the UnitEntity::squad of the represented UnitEntity.

Exceptions

IllegalArgumentException	When damage is less than #UNUSED or squad is neither null nor greater than zero.

See also

CardUiStateFactory::createCardUiState(UnitEntity, DamageCalculator)

Definition at line 108 of file CardUiState.java.
References CardUiState.abilityImageId, CardUiState.damageBackgroundImageId, CardUiState.damageTextColor, CardUiState.unitId, and CardUiState.UNUSED.

5.10.3 Member Function Documentation

5.10.3.1 equals()

```
boolean equals (
Object o )
```

Checks whether the given Object looks the same as this CardUiState.

Does not compare #unitId since that field does not influence the visual representation of CardUiState objects.

Parameters

o	Object that is being compared with this CardUiState.

Returns

A Boolean defining whether the objects look the same.

Definition at line 212 of file CardUiState.java.

References CardUiState.abilityImageId, CardUiState.damageBackgroundImageId, CardUiState.damageString, CardUiState.damageTextColor, CardUiState.equals(), and CardUiState.squadString.

Referenced by CardUiState.equals().

5.10.3.2 getAbilityImageId()

```
int getAbilityImageId ( )
```

Returns the drawable resource shown in the ability view.

Returns

An Integer referencing a drawable resource.

See also

#abilityImageId

#showAbility()

Definition at line 189 of file CardUiState.java.

References CardUiState.abilityImageId.

Referenced by CardListAdapter.onBindViewHolder().

5.10.3.3 getDamageBackgroundImageId()

```
int getDamageBackgroundImageId ( )
```

Returns the drawable resource shown as the background of the damage view.

Returns

An Integer referencing a drawable resource.

See also

#damageBackgroundImageId

Definition at line 158 of file CardUiState.java.

References CardUiState.damageBackgroundImageId.

Referenced by CardListAdapter.onBindViewHolder().

5.10.3.4 getDamageString()

```
String getDamageString ( )
```

Returns the String shown in the damage view.

Returns

A String containing the damage of the represented UnitEntity or nothing if the unit is empty.

See also

#damageString

Definition at line 168 of file CardUiState.java.

References CardUiState.damageString.

Referenced by CardListAdapter.onBindViewHolder().

5.10.3.5 getDamageTextColor()

```
int getDamageTextColor ( )
```

Returns the text color of the damage view.

Returns

An Integer representing a color.

See also

#damageTextColor

Definition at line 178 of file CardUiState.java.
References CardUiState.damageTextColor.
Referenced by CardListAdapter.onBindViewHolder().

5.10.3.6 getSquadString()

```
String getSquadString ( )
```

Returns the String shown in the squad view.

Returns

A String containing the squad of the represented UnitEntity or nothing.

See also

#squadString

#showSquad()

Definition at line 200 of file CardUiState.java.
References CardUiState.squadString.
Referenced by CardListAdapter.onBindViewHolder().

5.10.3.7 getUnitId()

```
int getUnitId ( )
```

Returns the #unitId of the represented UnitEntity.

See also

#unitId

Returns

An Integer referencing the UnitEntity::id of the represented UnitEntity.

Definition at line 148 of file CardUiState.java.
References CardUiState.unitId.
Referenced by CardListAdapter.onBindViewHolder().

5.10.3.8 showAbility()

```
boolean showAbility ( )
```

Returns whether the ability view is shown, i.e., if the unit has an ability other than Ability::NONE.

Returns

A Boolean defining whether the ability view is shown.

Definition at line 130 of file CardUiState.java.
References CardUiState.abilityImageId, and CardUiState.UNUSED.
Referenced by CardListAdapter.onBindViewHolder(), and CardUiState.showSquad().

5.10.3.9 showSquad()

```
boolean showSquad ( )
```

Returns whether the squad view is shown, i.e., if the unit has the Ability::BINDING ability.

Returns

A Boolean defining whether the squad view is shown.

Definition at line 139 of file CardUiState.java.
References CardUiState.showAbility(), and CardUiState.squadString.
Referenced by CardListAdapter.onBindViewHolder().

5.10.4 Member Data Documentation

5.10.4.1 abilityImageId

```
final int abilityImageId [private]
```

Integer referencing the drawable resource shown in the ability view.
May be #UNUSED if the view is not visible.

See also

#UNUSED

#getAbilityImageId()

Definition at line 88 of file CardUiState.java.
Referenced by CardUiState.CardUiState(), CardUiState.equals(), CardUiState.getAbilityImageId(), and CardUiState.showAbility().

5.10.4.2 damageBackgroundImageId

```
final int damageBackgroundImageId [private]
```

Integer referencing the drawable resource shown as the background of the damage view.

See also

#getDamageBackgroundImageId()

Definition at line 64 of file CardUiState.java.
Referenced by CardUiState.CardUiState(), CardUiState.equals(), and CardUiState.getDamageBackgroundImageId().

5.10.4.3 damageString

```
final String damageString [private]
```

String containing the number shown in the damage view or nothing if the represented UnitEntity is epic.

See also

#getDamageString()

Definition at line 72 of file CardUiState.java.
Referenced by CardUiState.equals(), and CardUiState.getDamageString().

5.10.4.4 damageTextColor

```
final int damageTextColor [private]
```

Integer representing the text color of the damage view.

See also

#getDamageTextColor()

Definition at line 79 of file CardUiState.java.
Referenced by CardUiState.CardUiState(), CardUiState.equals(), and CardUiState.getDamageTextColor().

5.10.4.5 DIFF_CALLBACK

```
final DiffUtil.ItemCallback<CardUiState> DIFF_CALLBACK  [static]
```

Initial value:

```
= new DiffUtil.ItemCallback<CardUiState>() {
        @Override
        public boolean areItemsTheSame(@NonNull CardUiState oldItem, @NonNull CardUiState newItem) {
            return oldItem.unitId == newItem.unitId;
        }

        @Override
        public boolean areContentsTheSame(@NonNull CardUiState oldItem, @NonNull CardUiState newItem) {
            return Objects.equals(oldItem, newItem);
        }
    }
```

DiffUtil.ItemCallback used to compare different CardUiState objects in a ListAdapter, e.g., CardListAdapter.

See also

#unitId

#equals(Object)

Definition at line 33 of file CardUiState.java.
Referenced by CardListAdapter.CardListAdapter().

5.10.4.6 squadString

```
final String squadString  [private]
```

String containing the number shown in the squad view.

See also

#getSquadString()

Definition at line 95 of file CardUiState.java.
Referenced by CardUiState.equals(), CardUiState.getSquadString(), and CardUiState.showSquad().

5.10.4.7 unitId

```
final int unitId  [private]
```

Integer referencing the UnitEntity::id of the represented UnitEntity.
The only member that is not shown in the UI.

See also

#getUnitId()

Definition at line 57 of file CardUiState.java.
Referenced by CardUiState.CardUiState(), and CardUiState.getUnitId().

5.10.4.8 UNUSED

```
final int UNUSED = -1  [static]
```

Integer that is used instead of a drawable resource to indicate the absence of a meaningful #abilityImage.

See also

#abilityImageId

Definition at line 50 of file CardUiState.java.
Referenced by CardUiState.CardUiState(), CardUiStateFactory.createCardUiState(), and CardUiState.showAbility().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/cards/CardUiState.java

5.11 CardUiStateFactory Class Reference

A factory class responsible for creating CardUiState objects from UnitEntity objects.
Collaboration diagram for CardUiStateFactory:

Public Member Functions

- CardUiStateFactory (@NonNull Context context, boolean weather, boolean horn)

  Constructor of a CardUiStateFactory for a row with the given weather and horn (de-)buff values.
- List< CardUiState > createCardUiState (@NonNull Collection< UnitEntity > units)

  Creates a List of CardUiState objects from the given List of UnitEntity objects.
- CardUiState createCardUiState (@NonNull UnitEntity unit, @NonNull DamageCalculator calculator)

  Creates a CardUiState from the given UnitEntity.

Private Attributes

- final boolean weather

  Boolean defining the status of the weather debuff in the row for which CardUiState objects are created.
- final boolean horn

  Boolean defining the status of the horn buff in the row for which CardUiState objects are created.
- final Map< Color, Integer > damageTextColors = new HashMap<>(Color.values().length)

  Map containing a color integer for each Color value possible, i.e., Color::DEFAULT, Color::BUFFED, and Color::↩ DEBUFFED.

5.11.1 Detailed Description

A factory class responsible for creating CardUiState objects from UnitEntity objects.

See also

CardUiState

Definition at line 28 of file CardUiStateFactory.java.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 CardUiStateFactory()

```
CardUiStateFactory (
             @NonNull Context context,
             boolean weather,
             boolean horn )
```

Constructor of a CardUiStateFactory for a row with the given weather and horn (de-)buff values.
The Context parameter is used to retrieve the color values saved in #damageTextColors.

Parameters

context	Context object used to get colors.
weather	Boolean defining whether the weather debuff is active.
horn	Boolean defining whether the horn buff is active.

See also

Context::getColor(int)

Definition at line 58 of file CardUiStateFactory.java.
References CardUiStateFactory.damageTextColors, CardUiStateFactory.horn, and CardUiStateFactory.weather.

5.11.3 Member Function Documentation

5.11.3.1 createCardUiState() [1/2]

```
List< CardUiState > createCardUiState (
            @NonNull Collection< UnitEntity > units )
```

Creates a List of CardUiState objects from the given List of UnitEntity objects.
Basically calls #createCardUiState(UnitEntity, DamageCalculator) for each given UnitEntity.

See also

#createCardUiState(UnitEntity, DamageCalculator)

Parameters

units	List of UnitEntity objects that are converted to CardUiState objects.

Returns

A List of CardUiState objects from the given UnitEntity objects.

Definition at line 74 of file CardUiStateFactory.java.
References CardUiStateFactory.createCardUiState(), CardUiStateFactory.horn, and CardUiStateFactory.weather.
Referenced by CardUiStateFactory.createCardUiState().

5.11.3.2 createCardUiState() [2/2]

```
CardUiState createCardUiState (
            @NonNull UnitEntity unit,
            @NonNull DamageCalculator calculator )
```

Creates a CardUiState from the given UnitEntity.

See also

#createCardUiState(Collection)

Parameters

unit	UnitEntity that is converted to a CardUiState.
calculator	DamageCalculator used to calculate damage and Color of the given UnitEntity.

Returns

A CardUiState obtained from the given UnitEntity.

Definition at line 89 of file CardUiStateFactory.java.
References CardUiStateFactory.damageTextColors, and CardUiState.UNUSED.

5.11.4 Member Data Documentation

5.11.4.1 damageTextColors

```
final Map<Color, Integer> damageTextColors = new HashMap<>(Color.values().length)  [private]
```

Map containing a color integer for each Color value possible, i.e., Color::DEFAULT, Color::BUFFED, and Color::↩ DEBUFFED.
Definition at line 46 of file CardUiStateFactory.java.
Referenced by CardUiStateFactory.CardUiStateFactory(), and CardUiStateFactory.createCardUiState().

5.11.4.2 horn

`final boolean horn [private]`
Boolean defining the status of the horn buff in the row for which CardUiState objects are created.
Definition at line 39 of file CardUiStateFactory.java.
Referenced by CardUiStateFactory.CardUiStateFactory(), and CardUiStateFactory.createCardUiState().

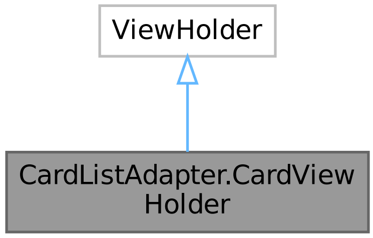
5.11.4.3 weather

`final boolean weather [private]`
Boolean defining the status of the weather debuff in the row for which CardUiState objects are created.
Definition at line 33 of file CardUiStateFactory.java.
Referenced by CardUiStateFactory.CardUiStateFactory(), and CardUiStateFactory.createCardUiState().
The documentation for this class was generated from the following file:

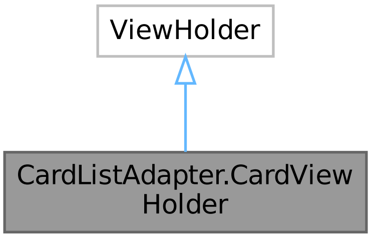
- gwent/ui/dialogs/cards/CardUiStateFactory.java

5.12 CardListAdapter.CardViewHolder Class Reference

A RecyclerView.ViewHolder class managing a card view that shows a representation of the UnitEntity referenced by #itemId.
Inheritance diagram for CardListAdapter.CardViewHolder:



Collaboration diagram for CardListAdapter.CardViewHolder:



Package Functions

- CardViewHolder (@NonNull View itemView, @NonNull Consumer< Integer > onCopy, @NonNull Consumer< Integer > onRemove)

  Constructor of a CardViewHolder for the given View initializing #damageView, #abilityView, and #bindingView.
- TextView getDamageView ()

  Returns the #damageView of the managed View.
- ImageView getAbilityView ()

  Returns the #abilityView of the managed View.
- TextView getBindingView ()

  Returns the #bindingView of the managed View.
- void setItemId (int itemId)

  Sets the #unitId of this ViewHolder.

Private Attributes

- final TextView damageView

  TextView showing the (de-)buffed damage of the represented UnitEntity.
- final ImageView abilityView

  ImageView showing an image of the UnitEntity::ability of the represented UnitEntity.
- final TextView bindingView

  TextView showing the UnitEntity::squad of the represented UnitEntity if UnitEntity::squad is Ability::BINDING.
- int itemId = (int) NO_ID

  Integer referencing the UnitEntity::id of the represented UnitEntity.

5.12.1 Detailed Description

A RecyclerView.ViewHolder class managing a card view that shows a representation of the UnitEntity referenced by #itemId.

See also

RecyclerView.ViewHolder

Definition at line 131 of file CardListAdapter.java.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 CardViewHolder()

```
CardViewHolder (
             @NonNull View itemView,
             @NonNull Consumer< Integer > onCopy,
             @NonNull Consumer< Integer > onRemove ) [package]
```

Constructor of a CardViewHolder for the given View initializing #damageView, #abilityView, and #bindingView. Sets View.OnClickListener on copy-button and delete-button calling the given onCopy and onRemove callbacks with #unitId.

Parameters

itemView	View with the card layout.
onCopy	Consumer that is called with #unitId, when the copy-button of the given View is clicked.
onRemove	Consumer that is called with #unitId, when the delete-button of the given View is clicked.

Definition at line 173 of file CardListAdapter.java.
References CardListAdapter.CardViewHolder.abilityView, CardListAdapter.CardViewHolder.bindingView, CardListAdapter.CardViewH
CardListAdapter.CardViewHolder.itemId, CardListAdapter.onCopy, and CardListAdapter.onRemove.

5.12.3 Member Function Documentation

5.12.3.1 getAbilityView()

```
ImageView getAbilityView ( ) [package]
```

Returns the #abilityView of the managed View.

Returns

An ImageView showing the UnitEntity::ability of the represented UnitEntity.

See also

#abilityView

Definition at line 211 of file CardListAdapter.java.
References CardListAdapter.CardViewHolder.abilityView.

5.12.3.2 getBindingView()

```
TextView getBindingView ( ) [package]
```

Returns the #bindingView of the managed View.

Returns

A TextView showing the UnitEntity::squad of the represented UnitEntity.

See also

#bindingView

Definition at line 221 of file CardListAdapter.java.
References CardListAdapter.CardViewHolder.bindingView.

5.12.3.3 getDamageView()

```
TextView getDamageView ( ) [package]
```

Returns the #damageView of the managed View.

Returns

A TextView showing the damage of the represented UnitEntity.

See also

#damageView

Definition at line 201 of file CardListAdapter.java.
References CardListAdapter.CardViewHolder.damageView.

5.12.3.4 setItemId()

```
void setItemId (
            int itemId )  [package]
```

Sets the #unitId of this ViewHolder.

Parameters

item↩ Id	Integer referencing the UnitEntity::id of the represented UnitEntity.

Exceptions

IllegalArgumentException	When itemId is negative.

See also

#itemId

Definition at line 232 of file CardListAdapter.java.
References CardListAdapter.CardViewHolder.itemId.

5.12.4 Member Data Documentation

5.12.4.1 abilityView

```
final ImageView abilityView  [private]
```

ImageView showing an image of the UnitEntity::ability of the represented UnitEntity.
If UnitEntity::ability is Ability::NONE, this view's visibility should be View::GONE.

See also

#getAbilityView()

Definition at line 144 of file CardListAdapter.java.
Referenced by CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.CardViewHolder.getAbilityView().

5.12.4.2 bindingView

```
final TextView bindingView  [private]
```

TextView showing the UnitEntity::squad of the represented UnitEntity if UnitEntity::squad is Ability::BINDING.
If UnitEntity::ability is not Ability::BINDING, this view's visibility should be View::GONE.

See also

#getBindingView()

Definition at line 152 of file CardListAdapter.java.
Referenced by CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.CardViewHolder.getBindingView().

5.12.4.3 damageView

```
final TextView damageView [private]
```

TextView showing the (de-)buffed damage of the represented UnitEntity.

See also

#getDamageView()

Definition at line 137 of file CardListAdapter.java.
Referenced by CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.CardViewHolder.getDamageView().

5.12.4.4 itemId

```
int itemId = (int) NO_ID [private]
```

Integer referencing the UnitEntity::id of the represented UnitEntity.
Is initialized with #NO_ID and therefore must be set using #setItemId() in #onBindViewHolder(CardViewHolder, int).

See also

#setItemId(int)

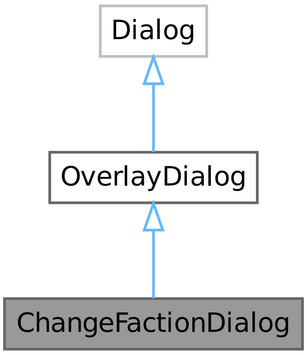
#onBindViewHolder(CardViewHolder, int)

Definition at line 161 of file CardListAdapter.java.
Referenced by CardListAdapter.CardViewHolder.CardViewHolder(), and CardListAdapter.CardViewHolder.setItemId().
The documentation for this class was generated from the following file:

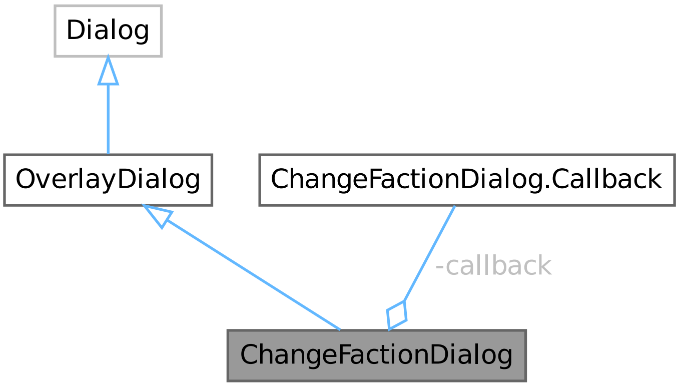
- gwent/ui/dialogs/cards/CardListAdapter.java

5.13 ChangeFactionDialog Class Reference

An OverlayDialog class used to change the faction design.
Inheritance diagram for ChangeFactionDialog:



Collaboration diagram for ChangeFactionDialog:



Classes

- interface Callback

    An interface used for propagating a selected theme back to the creator of a ChangeFactionDialog.

Public Member Functions

- ChangeFactionDialog (@NonNull Context context, @NonNull Callback callback)

    Constructor of a ChangeFactionDialog that calls the given Callback when one theme is selected.

Protected Member Functions

- void onCreate (Bundle savedInstanceState)

    Initializes layout and sets listeners for each view.

Protected Member Functions inherited from OverlayDialog

- OverlayDialog (@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)

    Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
- OverlayDialog (@NonNull Context context, @LayoutRes int layout)

    Constructor of an OverlayDialog in the given Context and with the given layout.

Private Member Functions

- View.OnClickListener getOnThemeClickListener (@IntRange(from=THEME_MONSTER, to=THEME_↩ SCOIATAEL) int theme)

    Returns a View.OnclickListener instance that calls #cancel() and uses #callback to propagate the selected theme.

Private Attributes

- final Callback callback

    Callback that is called when a theme is selected.

Additional Inherited Members

Static Package Attributes inherited from OverlayDialog

- static final int NO_DISMISS_VIEW = -1

  Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

5.13.1 Detailed Description

An OverlayDialog class used to change the faction design.
Definition at line 20 of file ChangeFactionDialog.java.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 ChangeFactionDialog()

```
ChangeFactionDialog (
            @NonNull Context context,
            @NonNull Callback callback )
```

Constructor of a ChangeFactionDialog that calls the given Callback when one theme is selected.

Parameters

context	Context in which this Dialog is run.
callback	Callback that is called when a theme is selected.

Definition at line 33 of file ChangeFactionDialog.java.
References ChangeFactionDialog.callback.

5.13.3 Member Function Documentation

5.13.3.1 getOnThemeClickListener()

```
View.OnClickListener getOnThemeClickListener (
            @IntRange(from=THEME_MONSTER, to=THEME_SCOIATAEL) int theme )  [private]
```

Returns a View.OnclickListener instance that calls #cancel() and uses #callback to propagate the selected theme.

Parameters

theme	Integer representing the selected theme.

Returns

A View.OnClickListener handling theme input.

See also

#cancel()

Callback::onThemeSelect(int)

Definition at line 73 of file ChangeFactionDialog.java.
References ChangeFactionDialog.callback, and ChangeFactionDialog.Callback.onThemeSelect().
Referenced by ChangeFactionDialog.onCreate().

5.13.3.2 onCreate()

```
void onCreate (
            Bundle savedInstanceState )  [protected]
```

Initializes layout and sets listeners for each view.

Parameters

savedInstanceState	If this dialog is being reinitialized after the hosting activity was previously shut down, holds the result from the most recent call to `onSaveInstanceState`, or null if this is the first time.

Reimplemented from OverlayDialog.
Definition at line 47 of file ChangeFactionDialog.java.
References ChangeFactionDialog.getOnThemeClickListener().

5.13.4 Member Data Documentation

5.13.4.1 callback

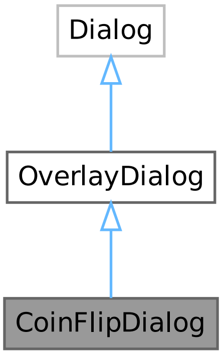
```
final Callback callback  [private]
```

Callback that is called when a theme is selected.
Definition at line 24 of file ChangeFactionDialog.java.
Referenced by ChangeFactionDialog.ChangeFactionDialog(), and ChangeFactionDialog.getOnThemeClickListener().
The documentation for this class was generated from the following file:

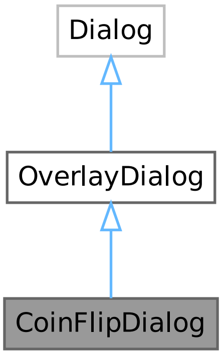
- gwent/ui/dialogs/ChangeFactionDialog.java

5.14 CoinFlipDialog Class Reference

An OverlayDialog class showing the result of a coin-flip.
Inheritance diagram for CoinFlipDialog:



Collaboration diagram for CoinFlipDialog:



Public Member Functions

- CoinFlipDialog (@NonNull Context context)

  Constructor of a CoinFlipDialog in the given Context.

Protected Member Functions

- void onCreate (Bundle savedInstanceState)

  Initializes layout and sets listeners for each view.

Protected Member Functions inherited from OverlayDialog

- OverlayDialog (@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)

  Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
- OverlayDialog (@NonNull Context context, @LayoutRes int layout)

  Constructor of an OverlayDialog in the given Context and with the given layout.

Static Private Attributes

- static final Random RANDOM = new Random()

  Random used to decide the result of the coin-flip.

Additional Inherited Members

Static Package Attributes inherited from OverlayDialog

- static final int NO_DISMISS_VIEW = -1

  Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

5.14.1 Detailed Description

An OverlayDialog class showing the result of a coin-flip.
Shows heads or tails with 48.5% probability respectively. The remaining 3% are divided equally between three tie situations.

See also

R.layout::popup_coin_normal

R.layout::popup_coin_stewie

R.layout::popup_coin_terry

R.layout::popup_coin_vin

Definition at line 24 of file CoinFlipDialog.java.

5.14.2 Constructor & Destructor Documentation

5.14.2.1 CoinFlipDialog()

```
CoinFlipDialog (
            @NonNull Context context )
```

Constructor of a CoinFlipDialog in the given Context.

Parameters

context	Context in which the Dialog is run.

Definition at line 34 of file CoinFlipDialog.java.

5.14.3 Member Function Documentation

5.14.3.1 onCreate()

```
void onCreate (
            Bundle savedInstanceState ) [protected]
```

Initializes layout and sets listeners for each view.
The result of the coin-flip is decided here.

Parameters

savedInstanceState	If this dialog is being reinitialized after the hosting activity was previously shut down, holds the result from the most recent call to `onSaveInstanceState`, or null if this is the first time.

Reimplemented from OverlayDialog.
Definition at line 48 of file CoinFlipDialog.java.
References CoinFlipDialog.RANDOM.

5.14.4 Member Data Documentation

5.14.4.1 RANDOM

```
final Random RANDOM = new Random() [static], [private]
```

Random used to decide the result of the coin-flip.
Definition at line 28 of file CoinFlipDialog.java.
Referenced by CoinFlipDialog.onCreate().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/CoinFlipDialog.java

5.15 DamageCalculator.Color Enum Reference

An enum containing values representing whether a unit is buffed, de-buffed or not affected by status effects.

Collaboration diagram for DamageCalculator.Color:

Public Attributes

- DEFAULT

  The unit is not affected by status effects.
- BUFFED

  The unit is buffed by status effects.
- DEBUFFED

  The unit is de-buffed by weather effects.

5.15.1 Detailed Description

An enum containing values representing whether a unit is buffed, de-buffed or not affected by status effects.
Definition at line 49 of file DamageCalculator.java.

5.15.2 Member Data Documentation

5.15.2.1 BUFFED

`BUFFED`
The unit is buffed by status effects.
Definition at line 57 of file DamageCalculator.java.

5.15.2.2 DEBUFFED

`DEBUFFED`
The unit is de-buffed by weather effects.
Definition at line 62 of file DamageCalculator.java.

5.15.2.3 DEFAULT

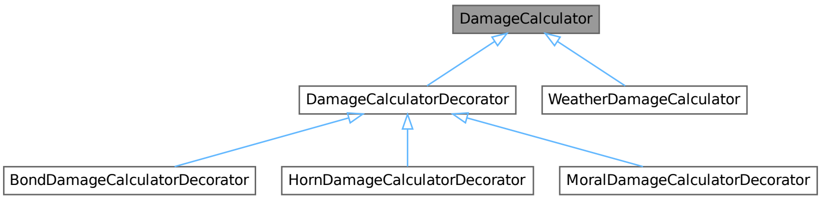
`DEFAULT`
The unit is not affected by status effects.
Definition at line 53 of file DamageCalculator.java.
The documentation for this enum was generated from the following file:

- gwent/domain/damage/DamageCalculator.java

5.16 DamageCalculator Interface Reference

An interface for a class capable of calculating the (de-)buffed damage of a unit for a given UnitEntity::id and UnitEntity::damage.

Inheritance diagram for DamageCalculator:


DamageCalculator
DamageCalculatorDecorator
WeatherDamageCalculator
BondDamageCalculatorDecorator
HornDamageCalculatorDecorator
MoralDamageCalculatorDecorator

Collaboration diagram for DamageCalculator:

Classes

- enum Color

  An enum containing values representing whether a unit is buffed, de-buffed or not affected by status effects.

Public Member Functions

- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.
- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.

5.16.1 Detailed Description

An interface for a class capable of calculating the (de-)buffed damage of a unit for a given UnitEntity::id and UnitEntity::damage.
Is used to implement the visitor design pattern, see com.peternaggschga.gwent.data.UnitEntity::calculateDamage()
The respective visitors are created as a decorator hierarchy, see DamageCalculatorBuildDirector::getCalculator().

See also

com.peternaggschga.gwent.data.UnitEntity::calculateDamage(DamageCalculator)

DamageCalculatorBuildDirector::getCalculator(boolean, boolean, Collection)

Definition at line 17 of file DamageCalculator.java.

5.16.2 Member Function Documentation

5.16.2.1 calculateDamage()

```
int calculateDamage (
            int id,
            @IntRange(from=0) int damage )
```

Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
Calculation is defined by the underlying decorator structure implemented via DamageCalculatorDecorator and WeatherDamageCalculator.

Parameters

id	Integer representing the UnitEntity::id of the unit whose (de-)buff damage is calculated.
damage	Integer representing the base-damage of the unit whose (de-)buff damage is calculated.

Returns

Integer representing the (de-)buffed damage of the unit.

See also

DamageCalculatorDecorator

WeatherDamageCalculator

Implemented in BondDamageCalculatorDecorator, HornDamageCalculatorDecorator, MoralDamageCalculatorDecorator, and WeatherDamageCalculator.
Referenced by BondDamageCalculatorDecorator.calculateDamage(), HornDamageCalculatorDecorator.calculateDamage(), and MoralDamageCalculatorDecorator.calculateDamage().

5.16.2.2 isBuffed()

```
Color isBuffed (
            int id )
```

Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.
Units are shown as Color::DEFAULT when they are not affected by any damage changing buffs or de-buffs. Units are shown as Color::DEBUFFED when they are only affected by the weather de-buff. Units are shown as Color::BUFFED when they are affected by any damage-increasing buff.

Parameters

id	Integer representing the UnitEntity::id of the unit buff status is calculated.

Returns

Color representing whether the unit is buffed, de-buffed or not affected.

See also

Color

Implemented in BondDamageCalculatorDecorator, HornDamageCalculatorDecorator, MoralDamageCalculatorDecorator, and WeatherDamageCalculator.
Referenced by BondDamageCalculatorDecorator.isBuffed(), HornDamageCalculatorDecorator.isBuffed(), and MoralDamageCalculatorDecorator.isBuffed().
The documentation for this interface was generated from the following file:

- gwent/domain/damage/DamageCalculator.java

5.17 DamageCalculatorBuildDirector Class Reference

A build director responsible for the creation of DamageCalculator instances from sets of units using DamageCalculatorBuilder.

Collaboration diagram for DamageCalculatorBuildDirector:

Static Public Member Functions

- static DamageCalculator getCalculator (boolean weather, boolean horn, @NonNull Collection< UnitEntity > units)

  Creates a DamageCalculator for a row with the given weather status, horn status and units.

Static Private Member Functions

- static void setSquads (@NonNull Collection< UnitEntity > units, @NonNull DamageCalculatorBuilder builder)

  Uses the given units to add a BondDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setBond().
- static void setMoralBoosts (@NonNull Collection< UnitEntity > units, @NonNull DamageCalculatorBuilder builder)

  Uses the given units to add a MoralDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setMoral().
- static void setHorns (boolean horn, @NonNull Collection< UnitEntity > units, @NonNull DamageCalculatorBuilder builder)

  Uses the given commander's horn status and the given units to add a HornDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setHorn().

5.17.1 Detailed Description

A build director responsible for the creation of DamageCalculator instances from sets of units using DamageCalculatorBuilder.
Encapsulates the logic for correct order of decorators.

See also

DamageCalculatorBuilder

Definition at line 21 of file DamageCalculatorBuildDirector.java.

5.17.2 Member Function Documentation

5.17.2.1 getCalculator()

```
static DamageCalculator getCalculator (
            boolean weather,
            boolean horn,
            @NonNull Collection< UnitEntity > units ) [static]
```

Creates a DamageCalculator for a row with the given weather status, horn status and units.

Parameters

weather	Boolean defining whether the calculation encompasses the weather debuff.
horn	Boolean defining whether a commander's horn is in the row.
units	Collection of UnitEntity objects that are in the row.

Returns

A DamageCalculator object capable of calculating damage for the given units.

Definition at line 31 of file DamageCalculatorBuildDirector.java.
References DamageCalculatorBuilder.getResult(), DamageCalculatorBuildDirector.setHorns(), DamageCalculatorBuildDirector.setMo
DamageCalculatorBuildDirector.setSquads(), and DamageCalculatorBuilder.setWeather().

5.17.2.2 setHorns()

```
static void setHorns (
            boolean horn,
            @NonNull Collection< UnitEntity > units,
            @NonNull DamageCalculatorBuilder builder )  [static], [private]
```

Uses the given commander's horn status and the given units to add a HornDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setHorn().
To accomplish this, the given unit list is converted to a List of the ids of the units with the Ability::HORN Ability (including `null` if horn is `true`).

Parameters

horn	Boolean defining whether a commander's horn is in the row.
units	Collection of UnitEntity objects that are in the row.
builder	DamageCalculatorBuilder where the horn buff is added.

See also

DamageCalculatorBuilder::setHorn(List)

Definition at line 106 of file DamageCalculatorBuildDirector.java.
Referenced by DamageCalculatorBuildDirector.getCalculator().

5.17.2.3 setMoralBoosts()

```
static void setMoralBoosts (
            @NonNull Collection< UnitEntity > units,
            @NonNull DamageCalculatorBuilder builder )  [static], [private]
```

Uses the given units to add a MoralDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setMoral().
To accomplish this, the given unit list is converted to a List of the ids of the units with the Ability::MORAL_BOOST Ability.

Parameters

units	Collection of UnitEntity objects that are in the row.
builder	DamageCalculatorBuilder where the horn buff is added.

See also

DamageCalculatorBuilder::setMoral(List)

Definition at line 85 of file DamageCalculatorBuildDirector.java.
Referenced by DamageCalculatorBuildDirector.getCalculator().

5.17.2.4 setSquads()

```
static void setSquads (
            @NonNull Collection< UnitEntity > units,
            @NonNull DamageCalculatorBuilder builder )  [static], [private]
```

Uses the given units to add a BondDamageCalculatorDecorator to the builder using DamageCalculatorBuilder::setBond().
To accomplish this, the given unit list is converted to a Map from ids of units with the Ability::BINDING Ability to the respective squad size.

Parameters

units	Collection of UnitEntity objects that are in the row.
builder	DamageCalculatorBuilder where the horn buff is added.

See also

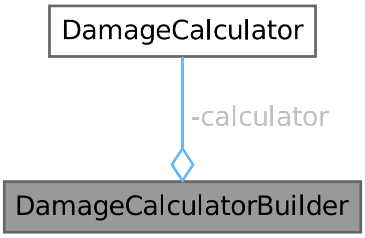
DamageCalculatorBuilder::setBond(Map)

Definition at line 61 of file DamageCalculatorBuildDirector.java.
Referenced by DamageCalculatorBuildDirector.getCalculator().
The documentation for this class was generated from the following file:

- gwent/domain/damage/DamageCalculatorBuildDirector.java

5.18 DamageCalculatorBuilder Class Reference

A builder class responsible for creating a decorator structure of DamageCalculator classes with the given parameters.
Collaboration diagram for DamageCalculatorBuilder:



Package Functions

- void setWeather (boolean weather)

  Sets the weather debuff for the calculation.
- void setBond (@NonNull Map< Integer, Integer > idToSquad)

  Sets the tight bond buff for the calculation.
- void setMoral (@NonNull List< Integer > unitIds)

  Sets the moral boost buff for the calculation.
- void setHorn (@NonNull List< Integer > unitIds)

  Sets the commander's horn buff for the calculation.
- DamageCalculator getResult ()

  Returns the current state of the built DamageCalculator, i.e.

Private Attributes

- DamageCalculator calculator = new WeatherDamageCalculator(false)

  The DamageCalculator instance that is being built by this DamageCalculatorBuilder.

5.18.1 Detailed Description

A builder class responsible for creating a decorator structure of DamageCalculator classes with the given parameters.
Should only be used by DamageCalculatorBuildDirector. Calls to #setWeather(), #setBond(), #setMoral() or #set↩
Horn() should be in that exact order to create a correct DamageCalculator. Every function call is optional, i.e. every function should be called once or not at all.

See also

DamageCalculatorBuildDirector

DamageCalculator

DamageCalculatorDecorator

Definition at line 19 of file DamageCalculatorBuilder.java.

5.18.2 Member Function Documentation

5.18.2.1 getResult()

```
DamageCalculator getResult ( )  [package]
```

Returns the current state of the built DamageCalculator, i.e.
#calculator.

Returns

A DamageCalculator that has been built by this builder.

Definition at line 72 of file DamageCalculatorBuilder.java.
References DamageCalculatorBuilder.calculator.
Referenced by DamageCalculatorBuildDirector.getCalculator().

5.18.2.2 setBond()

```
void setBond (
              @NonNull Map< Integer, Integer > idToSquad )  [package]
```

Sets the tight bond buff for the calculation.
Should be called after #setWeather() or never.

Parameters

idToSquad	Map mapping the ids of all units with the com.peternaggschga.gwent.data.Ability::BINDING ability to the respective squad size.

Definition at line 41 of file DamageCalculatorBuilder.java.
References DamageCalculatorBuilder.calculator.

5.18.2.3 setHorn()

```
void setHorn (
              @NonNull List< Integer > unitIds )  [package]
```

Sets the commander's horn buff for the calculation.
Should be called after #setMoral() or never.

Parameters

unitIds	List of Integers representing ids of units with the com.peternaggschga.gwent.data.Ability::HORN ability.

Definition at line 62 of file DamageCalculatorBuilder.java.
References DamageCalculatorBuilder.calculator.

5.18.2.4 setMoral()

```
void setMoral (
             @NonNull List< Integer > unitIds )  [package]
```

Sets the moral boost buff for the calculation.
Should be called after #setBond() or never.

Parameters

unitIds	List of Integers representing ids of units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability.

Definition at line 51 of file DamageCalculatorBuilder.java.
References DamageCalculatorBuilder.calculator.

5.18.2.5 setWeather()

```
void setWeather (
             boolean weather )  [package]
```

Sets the weather debuff for the calculation.
Should be called first or never.

Parameters

weather	Boolean defining whether the weather debuff is active.

Definition at line 32 of file DamageCalculatorBuilder.java.
References DamageCalculatorBuilder.calculator.
Referenced by DamageCalculatorBuildDirector.getCalculator().

5.18.3 Member Data Documentation

5.18.3.1 calculator

```
DamageCalculator calculator = new WeatherDamageCalculator(false)  [private]
```

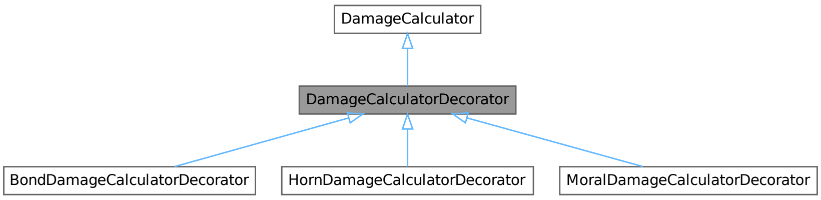
The DamageCalculator instance that is being built by this DamageCalculatorBuilder.
Starts as a WeatherDamageCalculator, since this is always at the end of the decorator chain.
Definition at line 25 of file DamageCalculatorBuilder.java.
Referenced by DamageCalculatorBuilder.getResult(), DamageCalculatorBuilder.setBond(), DamageCalculatorBuilder.setHorn(), DamageCalculatorBuilder.setMoral(), and DamageCalculatorBuilder.setWeather().
The documentation for this class was generated from the following file:

- gwent/domain/damage/DamageCalculatorBuilder.java

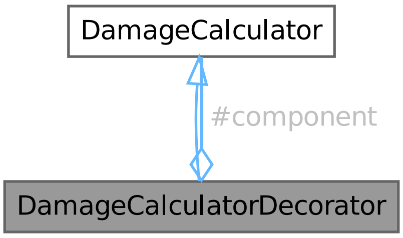
5.19 DamageCalculatorDecorator Class Reference

An abstract DamageCalculator that is used as a superclass for the decorator classes implementing the damage calculation.

Inheritance diagram for DamageCalculatorDecorator:


DamageCalculator
DamageCalculatorDecorator
BondDamageCalculatorDecorator
HornDamageCalculatorDecorator
MoralDamageCalculatorDecorator

Collaboration diagram for DamageCalculatorDecorator:



Protected Attributes

- final DamageCalculator component

  A DamageCalculator which is decorated by this decorator.

Package Functions

- DamageCalculatorDecorator (@NonNull DamageCalculator component)

  Constructor of a DamageCalculatorDecorator.

Additional Inherited Members

Public Member Functions inherited from DamageCalculator

- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.
- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.

5.19.1 Detailed Description

An abstract DamageCalculator that is used as a superclass for the decorator classes implementing the damage calculation.
Definition at line 9 of file DamageCalculatorDecorator.java.

5.19.2 Constructor & Destructor Documentation

5.19.2.1 DamageCalculatorDecorator()

```
DamageCalculatorDecorator (
            @NonNull DamageCalculator component )  [package]
```

Constructor of a DamageCalculatorDecorator.

Parameters

component	DamageCalculator that is being decorated by this decorator.

Definition at line 20 of file DamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component.

5.19.3 Member Data Documentation

5.19.3.1 component

```
final DamageCalculator component  [protected]
```

A DamageCalculator which is decorated by this decorator.
Definition at line 13 of file DamageCalculatorDecorator.java.
Referenced by BondDamageCalculatorDecorator.BondDamageCalculatorDecorator(), DamageCalculatorDecorator.DamageCalculat
HornDamageCalculatorDecorator.HornDamageCalculatorDecorator(), MoralDamageCalculatorDecorator.MoralDamageCalculatorDe
BondDamageCalculatorDecorator.calculateDamage(), HornDamageCalculatorDecorator.calculateDamage(),
MoralDamageCalculatorDecorator.calculateDamage(), BondDamageCalculatorDecorator.isBuffed(), HornDamageCalculatorDecorato
and MoralDamageCalculatorDecorator.isBuffed().
The documentation for this class was generated from the following file:

- gwent/domain/damage/DamageCalculatorDecorator.java

5.20 DamageCalculatorUseCase Class Reference

A use case class responsible for creating DamageCalculator objects for rows using DamageCalculatorBuildDirector.
Collaboration diagram for DamageCalculatorUseCase:

Static Public Member Functions

- static Single< DamageCalculator > getDamageCalculator (@NonNull UnitRepository repository, @NonNull RowType row)

  Returns a Single emitting a DamageCalculator object for the given row with properties retrieved from the given Unit↩ Repository.
- static DamageCalculator getDamageCalculator (boolean weather, boolean horn, @NonNull Collection< UnitEntity > units)

  Creates a DamageCalculator for a row with the given weather status, horn status and units.

5.20.1 Detailed Description

A use case class responsible for creating DamageCalculator objects for rows using DamageCalculatorBuildDirector.

See also

DamageCalculatorBuildDirector

DamageCalculator

Definition at line 23 of file DamageCalculatorUseCase.java.

5.20.2 Member Function Documentation

5.20.2.1 getDamageCalculator() [1/2]

```
static Single< DamageCalculator > getDamageCalculator (
            @NonNull UnitRepository repository,
            @NonNull RowType row )  [static]
```

Returns a Single emitting a DamageCalculator object for the given row with properties retrieved from the given UnitRepository.

Parameters

repository	UnitRepository used for data collection.
row	RowType defining which row is queried for DamageCalculator generation.

Returns

A Single emitting a DamageCalculator object for the given row.

Definition at line 32 of file DamageCalculatorUseCase.java.
References DamageCalculatorUseCase.getDamageCalculator().
Referenced by BurnDialogUseCase.getBurnUnits(), and DamageCalculatorUseCase.getDamageCalculator().

5.20.2.2 getDamageCalculator() [2/2]

```
static DamageCalculator getDamageCalculator (
            boolean weather,
            boolean horn,
            @NonNull Collection< UnitEntity > units )  [static]
```

Creates a DamageCalculator for a row with the given weather status, horn status and units.
Basically calls DamageCalculatorBuildDirector::getCalculator with the given parameters.

Parameters

weather	Boolean defining whether the calculation encompasses the weather debuff.
horn	Boolean defining whether a commander's horn is in the row.
units	Collection of UnitEntity objects that are in the row.

Returns

A DamageCalculator object capable of calculating damage for the given units.

See also

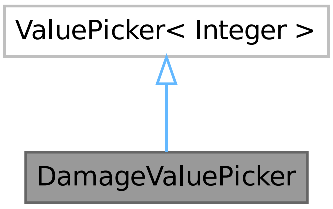
DamageCalculatorBuildDirector::getCalculator(boolean, boolean, Collection)

Definition at line 50 of file DamageCalculatorUseCase.java.
The documentation for this class was generated from the following file:

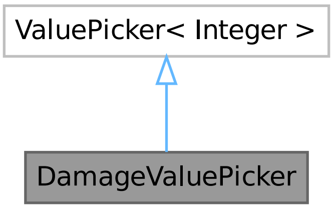
- gwent/domain/cases/DamageCalculatorUseCase.java

5.21 DamageValuePicker Class Reference

A ValuePicker used for selection of damage values for UnitEntity::damage.
Inheritance diagram for DamageValuePicker:



Collaboration diagram for DamageValuePicker:



Protected Member Functions

- String getDisplayString (@NonNull Integer value)

  Returns a localized String representing the given value when #epicValues is false or otherwise the corresponding epic damage value.

Package Functions

- DamageValuePicker (@NonNull NumberPicker picker)

  Constructor of a DamageValuePicker wrapping the given NumberPicker.
- void setValue (@NonNull @IntRange(from=0, to=NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) Integer value)

  Sets the picker to the given value.
- Integer getValue ()

  Returns the currently selected value.
- void setEpicValues (boolean epicValues)

  Switches the selectableValues between non-epic and epic damage values.
- void setSelectableValues (@NonNull Collection< Integer > values, @Nullable Integer defaultValue)

  Does nothing.

Private Attributes

- boolean epicValues = false

  Boolean defining whether or not this DamageValuePicker shows epic damage values.

5.21.1 Detailed Description

A ValuePicker used for selection of damage values for UnitEntity::damage.
Allows for switching between epic and normal damage values using #setEpicValues().
Definition at line 25 of file DamageValuePicker.java.

5.21.2 Constructor & Destructor Documentation

5.21.2.1 DamageValuePicker()

```
DamageValuePicker (
            @NonNull NumberPicker picker )  [package]
```

Constructor of a DamageValuePicker wrapping the given NumberPicker.
Calls #setEpicValues() to initialize the NumberPicker with non-epic damage values.

See also

ValuePicker::ValuePicker(NumberPicker, SortedMap)

#setEpicValues(boolean)

Parameters

picker	NumberPicker wrapped by the created DamageValuePicker.

Definition at line 39 of file DamageValuePicker.java.
References DamageValuePicker.setEpicValues().

5.21.3 Member Function Documentation

5.21.3.1 getDisplayString()

```
String getDisplayString (
            @NonNull Integer value )  [protected]
```

Returns a localized String representing the given value when #epicValues is false or otherwise the corresponding epic damage value.

Parameters

value	Value that should be represented as a String.

Returns

A localized String representing the value.

Exceptions

IllegalStateException	When #epicValues is true and displayIntegers does not contain a mapping for the given value.

Definition at line 57 of file DamageValuePicker.java.
References DamageValuePicker.epicValues.

5.21.3.2 getValue()

```
Integer getValue ( )  [package]
```

Returns the currently selected value.

Returns

A value that is selected in picker.

Definition at line 90 of file DamageValuePicker.java.
References DamageValuePicker.epicValues.
Referenced by CardNumberPickerAdapter.addSelectedUnits().

5.21.3.3 setEpicValues()

```
void setEpicValues (
            boolean epicValues )  [package]
```

Switches the selectableValues between non-epic and epic damage values.
Sets #epicValues.

See also

#epicValues

Parameters

epicValues	Boolean defining whether or not epic values are shown.

Definition at line 100 of file DamageValuePicker.java.
References DamageValuePicker.epicValues.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and DamageValuePicker.DamageValuePicker().

5.21.3.4 setSelectableValues()

```
void setSelectableValues (
            @NonNull Collection< Integer > values,
            @Nullable Integer defaultValue )  [package]
```

Does nothing.
Overrides parent since selectableValues should only be modified by #setEpicValues().

Parameters

values	Collection of the new selectable values.
defaultValue	Value that is shown in the beginning. If null, then the first value defined by the Comparable interface is used.

Definition at line 121 of file DamageValuePicker.java.

5.21.3.5 setValue()

```
void setValue (
            @NonNull @IntRange(from=0, to=NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) Integer value )
[package]
```

Sets the picker to the given value.
If #epicValues is true, ValuePicker::setValue() is called. Otherwise the picker is simply set to the given value.

Parameters

value	Value that the picker is set to.

Exceptions

IllegalStateException	When #epicValues is false and the given value is not in [0,20].

See also

ValuePicker::setValue(Comparable)

Definition at line 74 of file DamageValuePicker.java.
References DamageValuePicker.epicValues.

5.21.4 Member Data Documentation

5.21.4.1 epicValues

```
boolean epicValues = false  [private]
```

Boolean defining whether or not this DamageValuePicker shows epic damage values.

See also

#setEpicValues(boolean)

Definition at line 30 of file DamageValuePicker.java.
Referenced by DamageValuePicker.getDisplayString(), DamageValuePicker.getValue(), DamageValuePicker.setEpicValues(), and DamageValuePicker.setValue().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/DamageValuePicker.java

5.22 ResetDialogUseCase.DialogType Enum Reference

An `Enum` defining which form of Dialog should be shown.
Collaboration diagram for ResetDialogUseCase.DialogType:

Public Attributes

- NONE

  No Dialog must be invoked.
- DEFAULT

  A default Dialog asking whether to reset should be invoked.
- MONSTER

  A monster Dialog asking whether to reset and whether to invoke the monster perk, should be invoked.

5.22.1 Detailed Description

An `Enum` defining which form of Dialog should be shown.

See also

#getDialogType(Context, UnitRepository, Trigger)

Definition at line 148 of file ResetDialogUseCase.java.

5.22.2 Member Data Documentation

5.22.2.1 DEFAULT

`DEFAULT`

A default Dialog asking whether to reset should be invoked.
Definition at line 156 of file ResetDialogUseCase.java.
Referenced by ResetDialogUseCase.getDialogType().

5.22.2.2 MONSTER

`MONSTER`

A monster Dialog asking whether to reset and whether to invoke the monster perk, should be invoked.
Definition at line 162 of file ResetDialogUseCase.java.
Referenced by ResetDialogUseCase.reset().

5.22.2.3 NONE

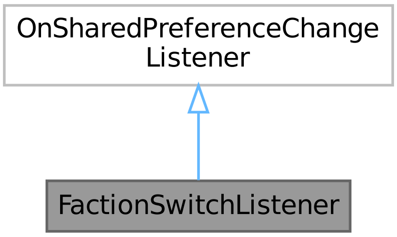
`NONE`

No Dialog must be invoked.
Definition at line 152 of file ResetDialogUseCase.java.
Referenced by ResetDialogUseCase.getDialogType(), and ResetDialogUseCase.reset().
The documentation for this enum was generated from the following file:

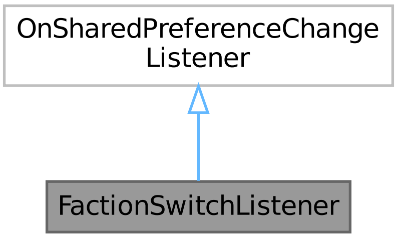
- gwent/domain/cases/ResetDialogUseCase.java

5.23 FactionSwitchListener Class Reference

A class implementing SharedPreference.OnSharedPreferenceChangeListener for the #THEME_PREFERENCE↩ _KEY SharedPreference.
Inheritance diagram for FactionSwitchListener:



Collaboration diagram for FactionSwitchListener:



Public Member Functions

- void onSharedPreferenceChanged (SharedPreferences sharedPreferences, @Nullable String key)

  Called when a shared preference is changed, added, or removed.

Static Public Member Functions

- static FactionSwitchListener getListener (@NonNull Window mainWindow)

  Returns a new FactionSwitchListener instance for the given Window.
- static void setTheme (@NonNull Context context)

  Changes the theme of the content referenced by the given `Context` depending on the preferences obtainable by said `Context`.

Static Public Attributes

- static final String THEME_PREFERENCE_KEY = "theme"

  String constant defining the key of the theme SharedPreference.
- static final int THEME_MONSTER = 0

  Integer constant representing the Monster theme.
- static final int THEME_NILFGAARD = 1

  Integer constant representing the Nilfgaard theme.
- static final int THEME_NORTHERN_KINGDOMS = 2

  Integer constant representing the Northern Kingdoms theme.
- static final int THEME_SCOIATAEL = 3

  Integer constant representing the Scoia'tael theme.

Private Member Functions

- FactionSwitchListener (@NonNull List< ImageView > ballViews, @NonNull List< ImageView > cardViews, @NonNull List< TextView > unitViews, @NonNull ImageButton factionButton)

  Constructor of a FactionSwitchListener updating the given View objects when #onSharedPreferenceChanged() is called.

Private Attributes

- final List< ImageView > ballViews

  List of ImageView objects that show a colored ball.
- final List< ImageView > cardViews

List of ImageView objects that show a card back.

- final List< TextView > unitViews

  List of TextView objects that show the number of units in a certain color.

- final ImageButton factionButton

  ImageButton showing the logo of the current faction.

5.23.1 Detailed Description

A class implementing SharedPreference.OnSharedPreferenceChangeListener for the #THEME_PREFERENCE↩ _KEY SharedPreference.
When this Preference is updated, the theme is switched in an animated way using ImageViewSwitchAnimator.

See also

ImageViewSwitchAnimator

Definition at line 31 of file FactionSwitchListener.java.

5.23.2 Constructor & Destructor Documentation

5.23.2.1 FactionSwitchListener()

```
FactionSwitchListener (
            @NonNull List< ImageView > ballViews,
            @NonNull List< ImageView > cardViews,
            @NonNull List< TextView > unitViews,
            @NonNull ImageButton factionButton )  [private]
```

Constructor of a FactionSwitchListener updating the given View objects when #onSharedPreferenceChanged() is called.
Should only be used by factory method #getListener().

See also

#getListener(Window)

Parameters

ballViews	List of ImageView objects showing a colored ball.
cardViews	List of ImageView objects showing the backside of a card.
unitViews	List of TextView objects showing the number of units.
factionButton	ImageButton that shows the current faction logo.

Definition at line 108 of file FactionSwitchListener.java.
References FactionSwitchListener.ballViews, FactionSwitchListener.cardViews, FactionSwitchListener.factionButton, and FactionSwitchListener.unitViews.

5.23.3 Member Function Documentation

5.23.3.1 getListener()

```
static FactionSwitchListener getListener (
            @NonNull Window mainWindow )  [static]
```

Returns a new FactionSwitchListener instance for the given Window.
Factory method of FactionSwitchListener.

Parameters

mainWindow	Window, that is updated by the new FactionSwitchListener.

Returns

A FactionSwitchListener instance for the given Window.

Definition at line 124 of file FactionSwitchListener.java.
References FactionSwitchListener.ballViews, FactionSwitchListener.cardViews, and FactionSwitchListener.unitViews.
Referenced by MainActivity.onCreate().

5.23.3.2 onSharedPreferenceChanged()

```
void onSharedPreferenceChanged (
            SharedPreferences sharedPreferences,
            @Nullable String key )
```

Called when a shared preference is changed, added, or removed.
Only handles changes when the given key is equal to #THEME_PREFERENCE_KEY. Updates the View objects in #ballViews, #cardViews, #unitViews, and #factionButton using ImageViewSwitchAnimator.

See also

ImageViewSwitchAnimator

Parameters

sharedPreferences	SharedPreferences that received the change.
key	String representing the key of the preference that was changed, added, or removed.

Definition at line 176 of file FactionSwitchListener.java.
References ImageViewSwitchAnimator.animatedSwitch(), FactionSwitchListener.ballViews, FactionSwitchListener.cardViews, FactionSwitchListener.factionButton, FactionSwitchListener.THEME_MONSTER, FactionSwitchListener.THEME_NILFGAARD, FactionSwitchListener.THEME_NORTHERN_KINGDOMS, FactionSwitchListener.THEME_PREFERENCE_KEY, FactionSwitchListener.THEME_SCOIATAEL, and FactionSwitchListener.unitViews.

5.23.3.3 setTheme()

```
static void setTheme (
            @NonNull Context context ) [static]
```

Changes the theme of the content referenced by the given `Context` depending on the preferences obtainable by said `Context`.

Parameters

context	`Context` that is used to query the current theme preference and that gets it theme updated accordingly.

Definition at line 148 of file FactionSwitchListener.java.
References FactionSwitchListener.THEME_MONSTER, FactionSwitchListener.THEME_NILFGAARD, FactionSwitchListener.THEME
FactionSwitchListener.THEME_PREFERENCE_KEY, and FactionSwitchListener.THEME_SCOIATAEL.
Referenced by MainActivity.onCreate().

5.23.4 Member Data Documentation

5.23.4.1 ballViews

```
final List<ImageView> ballViews [private]
```

List of ImageView objects that show a colored ball.

See also

R.drawable::ball_red

R.drawable::ball_grey

R.drawable::ball_blue

R.drawable::ball_green

Definition at line 66 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.FactionSwitchListener(), FactionSwitchListener.getListener(), and FactionSwitchListener.onShar

5.23.4.2 cardViews

```
final List<ImageView> cardViews  [private]
```

List of ImageView objects that show a card back.

See also

R.drawable::card_monster_landscape_free

R.drawable::card_nilfgaard_landscape_free

R.drawable::card_northern_kingdoms_landscape_free

R.drawable::card_scoiatael_landscape_free

Definition at line 76 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.FactionSwitchListener(), FactionSwitchListener.getListener(), and FactionSwitchListener.onShar

5.23.4.3 factionButton

```
final ImageButton factionButton  [private]
```

ImageButton showing the logo of the current faction.

See also

R.drawable::icon_round_monster

R.drawable::icon_round_nilfgaard

R.drawable::icon_round_northern_kingdoms

R.drawable::icon_round_scoiatael

Definition at line 96 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.FactionSwitchListener(), and FactionSwitchListener.onSharedPreferenceChanged().

5.23.4.4 THEME_MONSTER

```
final int THEME_MONSTER = 0  [static]
```

Integer constant representing the Monster theme.
Definition at line 41 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.onSharedPreferenceChanged(), and FactionSwitchListener.setTheme().

5.23.4.5 THEME_NILFGAARD

```
final int THEME_NILFGAARD = 1  [static]
```

Integer constant representing the Nilfgaard theme.
Definition at line 46 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.onSharedPreferenceChanged(), and FactionSwitchListener.setTheme().

5.23.4.6 THEME_NORTHERN_KINGDOMS

```
final int THEME_NORTHERN_KINGDOMS = 2  [static]
```

Integer constant representing the Northern Kingdoms theme.
Definition at line 51 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.onSharedPreferenceChanged(), and FactionSwitchListener.setTheme().

5.23.4.7 THEME_PREFERENCE_KEY

```
final String THEME_PREFERENCE_KEY = "theme"  [static]
```

String constant defining the key of the theme SharedPreference.
Definition at line 36 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.onSharedPreferenceChanged(), and FactionSwitchListener.setTheme().

5.23.4.8 THEME_SCOIATAEL

```
final int THEME_SCOIATAEL = 3  [static]
```

Integer constant representing the Scoia'tael theme.
Definition at line 56 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.onSharedPreferenceChanged(), and FactionSwitchListener.setTheme().

5.23.4.9 unitViews

```
final List<TextView> unitViews  [private]
```

List of TextView objects that show the number of units in a certain color.

See also

R.color::color_text_monster

R.color::color_text_nilfgaard

R.color::color_text_northern_kingdoms

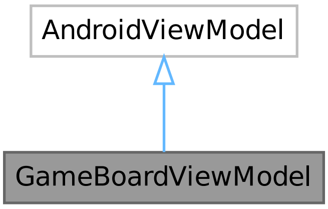
R.color::color_text_scoiatael

Definition at line 86 of file FactionSwitchListener.java.
Referenced by FactionSwitchListener.FactionSwitchListener(), FactionSwitchListener.getListener(), and FactionSwitchListener.onShar
The documentation for this class was generated from the following file:

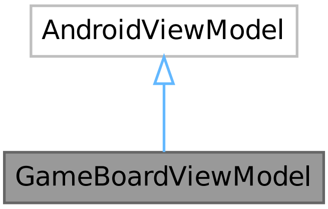
- gwent/ui/main/FactionSwitchListener.java

5.24 GameBoardViewModel Class Reference

An AndroidViewModel class responsible for encapsulating and offering state of views in activity_main.xml, i.e., that show the overall game board.
Inheritance diagram for GameBoardViewModel:



Collaboration diagram for GameBoardViewModel:



Public Member Functions

- Flowable< RowUiState > getRowUiState (@NonNull RowType row)

  Returns a Flowable object emitting RowUiState for the given row.
- Flowable< MenuUiState > getMenuUiState ()

  Returns a Flowable object emitting MenuUiState.
- SoundManager getSoundManager ()

  Returns the SoundManager managed by this GameBoardViewModel.
- Completable onWeatherViewPressed (@NonNull RowType row)

  Updates the weather debuff of the given row.
- Completable onHornViewPressed (@NonNull RowType row)

  Updates the horn buff of the given row.
- Completable onResetButtonPressed (@NonNull Context context)

  Triggers a reset and possibly an alert dialog, depending on preferences.
- Completable onFactionSwitchReset (@NonNull Context context)

  Triggers a reset and possibly an alert dialog, depending on preferences.
- Completable onWeatherButtonPressed ()

  Clears all weather effects.
- Completable onBurnButtonPressed (@NonNull Context context)

  Clears units with the most damage.

Static Public Member Functions

- static GameBoardViewModel getModel (@NonNull ViewModelStoreOwner owner, @NonNull UnitRepository repository, @NonNull SoundManager soundManager)

  Factory method of a GameBoardViewModel.

Private Member Functions

- GameBoardViewModel (@NonNull GwentApplication application)

  Constructor of a GameBoardViewModel object.
- Single< UnitRepository > getRepository ()

  Returns the UnitRepository used by the parent GwentApplication.
- Completable reset (@NonNull Context context, @NonNull ResetDialogUseCase.Trigger trigger)

  Triggers a reset and possibly an alert dialog, depending on preferences.

Private Attributes

- final Map< RowType, Flowable< RowUiState > > rowUiStates = new HashMap<>(RowType.↩ values().length)

  A map structure containing the Flowable objects emitting the RowUiState for each row.
- Flowable< MenuUiState > menuUiState = Flowable.empty()

  Flowable emitting the MenuUiState for the right-hand side menu.
- SoundManager soundManager

  SoundManager used to play Sound effects on click events.

Static Private Attributes

- static final ViewModelInitializer< GameBoardViewModel > INITIALIZER

  ViewModelInitializer used by androidx.lifecycle.ViewModelProvider.Factory to instantiate the class.

5.24.1 Detailed Description

An AndroidViewModel class responsible for encapsulating and offering state of views in activity_main.xml, i.e., that show the overall game board.
Click events on the rows and the menu are handled also.
Definition at line 38 of file GameBoardViewModel.java.

5.24.2 Constructor & Destructor Documentation

5.24.2.1 GameBoardViewModel()

```
GameBoardViewModel (
            @NonNull GwentApplication application )  [private]
```

Constructor of a GameBoardViewModel object.
Should only be called in #initializer.

Parameters

application	GwentApplication that uses this AndroidViewModel.

See also

#INITIALIZER

Definition at line 60 of file GameBoardViewModel.java.

5.24.3 Member Function Documentation

5.24.3.1 getMenuUiState()

```
Flowable< MenuUiState > getMenuUiState ( )
```

Returns a Flowable object emitting MenuUiState.

Returns

A Flowable object for the state of the menu.

See also

MenuUiState

#menuUiState

Definition at line 185 of file GameBoardViewModel.java.
References GameBoardViewModel.menuUiState.
Referenced by MainActivity.initializeViewModel().

5.24.3.2 getModel()

```
static GameBoardViewModel getModel (
            @NonNull ViewModelStoreOwner owner,
            @NonNull UnitRepository repository,
            @NonNull SoundManager soundManager )  [static]
```

Factory method of a GameBoardViewModel.
Creates a new GameBoardViewModel for the given owner and initializes #rowUiStates and #menuUiState.

Parameters

owner	ViewModelStoreOwner instantiating the GameBoardViewModel.
repository	UnitRepository where Flowables are retrieved.
soundManager	SoundManager used to play Sound effects on click events.

Returns

A new GameBoardViewModel instance.

See also

ViewModelProvider::ViewModelProvider(ViewModelStoreOwner, ViewModelProvider.Factory)

Definition at line 98 of file GameBoardViewModel.java.
References RowUiState.getDamage(), RowUiState.getUnits(), GameBoardViewModel.INITIALIZER, RowUiState.isHorn(), RowUiState.isWeather(), GameBoardViewModel.menuUiState, GameBoardViewModel.reset(), GameBoardViewModel.rowUiStates, and GameBoardViewModel.soundManager.
Referenced by MainActivity.onCreate().

5.24.3.3 getRepository()

```
Single< UnitRepository > getRepository ( )  [private]
```

Returns the UnitRepository used by the parent GwentApplication.
Basically a wrapper for GwentApplication::getRepository(Context).

Returns

A Single emitting the UnitRepository instance.

See also

GwentApplication::getRepository(Context)

Definition at line 161 of file GameBoardViewModel.java.
References GwentApplication.getRepository().
Referenced by GameBoardViewModel.onHornViewPressed(), GameBoardViewModel.onWeatherButtonPressed(), and GameBoardViewModel.onWeatherViewPressed().

5.24.3.4 getRowUiState()

```
Flowable< RowUiState > getRowUiState (
            @NonNull RowType row )
```

Returns a Flowable object emitting RowUiState for the given row.

Parameters

row	RowType defining the row for which the state is queried.

Returns

A Flowable object for the state of the given row.

See also

RowUiState

#rowUiStates

Definition at line 173 of file GameBoardViewModel.java.
References GameBoardViewModel.rowUiStates.
Referenced by MainActivity.initializeViewModel().

5.24.3.5 getSoundManager()

```
SoundManager getSoundManager ( )
```

Returns the SoundManager managed by this GameBoardViewModel.

Returns

A SoundManager used by this GameBoardViewModel.

Definition at line 195 of file GameBoardViewModel.java.
References GameBoardViewModel.soundManager.
Referenced by MainActivity.inflateCoinFlipPopup(), and MainActivity.initializeViewModel().

5.24.3.6 onBurnButtonPressed()

```
Completable onBurnButtonPressed (
             @NonNull Context context )
```

Clears units with the most damage.
May inflate a warning dialog depending on the user's preferences and a Toast informing the user about the burned units. Should only be called by the View.OnClickListener of the burn button. Plays a matching sound, if units were removed.

Parameters

context	Context

Returns

A Completable tracking operation status.

See also

BurnDialogUseCase::burn(Context, SoundManager)

Definition at line 310 of file GameBoardViewModel.java.
References GameBoardViewModel.soundManager.
Referenced by MainActivity.initializeViewModel().

5.24.3.7 onFactionSwitchReset()

```
Completable onFactionSwitchReset (
             @NonNull Context context )
```

Triggers a reset and possibly an alert dialog, depending on preferences.
Should only be called when the faction has been switched. Wrapper for #reset(Context, ResetDialogUseCase.↩ Trigger).

Parameters

context	Context object used to acquire SharedPreferences and inflate Dialog views.

Returns

A Completable tracking operation status.

See also

#reset(Context, ResetDialogUseCase.Trigger)

Definition at line 263 of file GameBoardViewModel.java.
References GameBoardViewModel.reset().
Referenced by MainActivity.inflateFactionPopup().

5.24.3.8 onHornViewPressed()

```
Completable onHornViewPressed (
            @NonNull RowType row )
```

Updates the horn buff of the given row.
Flips between on and off. Plays a matching sound, if the horn is switched on.

Parameters

row	RowType defining the affected row.

Returns

A Completable tracking operation status.

See also

UnitRepository::switchHorn(RowType)

Definition at line 227 of file GameBoardViewModel.java.
References GameBoardViewModel.getRepository(), and GameBoardViewModel.soundManager.
Referenced by MainActivity.initializeViewModel().

5.24.3.9 onResetButtonPressed()

```
Completable onResetButtonPressed (
            @NonNull Context context )
```

Triggers a reset and possibly an alert dialog, depending on preferences.
Should only be called by the button's View.OnClickListener. Wrapper for #reset(Context, ResetDialogUseCase.↩ Trigger).

Parameters

context	Context object used to acquire SharedPreferences and inflate Dialog views.

Returns

A Completable tracking operation status.

See also

#reset(Context, ResetDialogUseCase.Trigger)

ResetDialogUseCase.Trigger::BUTTON_CLICK

Definition at line 250 of file GameBoardViewModel.java.
References GameBoardViewModel.reset().
Referenced by MainActivity.initializeViewModel().

5.24.3.10 onWeatherButtonPressed()

```
Completable onWeatherButtonPressed ( )
```

Clears all weather effects.
Should only be called by the View.OnClickListener of the weather button. Plays a matching sound on completion.

Returns

A Completable tracking operation status.

See also

UnitRepository::clearWeather()

Definition at line 295 of file GameBoardViewModel.java.
References GameBoardViewModel.getRepository().
Referenced by MainActivity.initializeViewModel().

5.24.3.11 onWeatherViewPressed()

```
Completable onWeatherViewPressed (
             @NonNull RowType row )
```

Updates the weather debuff of the given row.
Flips between good and bad weather. Plays a matching sound, if the weather is switched on.

Parameters

row	RowType defining the affected row.

Returns

A Completable tracking operation status.

See also

UnitRepository::switchWeather(RowType)

Definition at line 207 of file GameBoardViewModel.java.
References GameBoardViewModel.getRepository(), and GameBoardViewModel.soundManager.
Referenced by MainActivity.initializeViewModel().

5.24.3.12 reset()

```
Completable reset (
             @NonNull Context context,
             @NonNull ResetDialogUseCase.Trigger trigger )  [private]
```

Triggers a reset and possibly an alert dialog, depending on preferences.
Plays a matching sound, if units were removed.

Parameters

context	Context object used to acquire SharedPreferences and inflate Dialog views.

Parameters

trigger	`com.peternaggschga.gwent.domain.cases.ResetDialogUseCase.Trigger` defining which action triggered the reset.

Returns

A Completable tracking operation status.

See also

#reset(Context, ResetDialogUseCase.Trigger)

Definition at line 276 of file GameBoardViewModel.java.
References GameBoardViewModel.soundManager.
Referenced by GameBoardViewModel.getModel(), GameBoardViewModel.onFactionSwitchReset(), and GameBoardViewModel.onResetButtonPressed().

5.24.4 Member Data Documentation

5.24.4.1 INITIALIZER

`final ViewModelInitializer<GameBoardViewModel> INITIALIZER [static], [private]`

Initial value:

```
= new ViewModelInitializer<>(
            GameBoardViewModel.class,
            creationExtras -> {
                GwentApplication app = (GwentApplication) creationExtras.get(APPLICATION_KEY);
                assert app != null;
                return new GameBoardViewModel(app);
            })
```

ViewModelInitializer used by androidx.lifecycle.ViewModelProvider.Factory to instantiate the class.

See also

androidx.lifecycle.ViewModelProvider.Factory::from(ViewModelInitializer[ ])

Definition at line 45 of file GameBoardViewModel.java.
Referenced by GameBoardViewModel.getModel().

5.24.4.2 menuUiState

`Flowable<MenuUiState> menuUiState = Flowable.empty() [private]`

Flowable emitting the MenuUiState for the right-hand side menu.
Initialized in #getModel().

See also

#getMenuUiState()

MenuUiState

Definition at line 80 of file GameBoardViewModel.java.
Referenced by GameBoardViewModel.getMenuUiState(), and GameBoardViewModel.getModel().

5.24.4.3 rowUiStates

`final Map<RowType, Flowable<RowUiState> > rowUiStates = new HashMap<>(RowType.values().length) [private]`

A map structure containing the Flowable objects emitting the RowUiState for each row.
Initialized in #getModel().

See also

#getRowUiState(RowType)

RowUiState

Definition at line 71 of file GameBoardViewModel.java.
Referenced by GameBoardViewModel.getModel(), and GameBoardViewModel.getRowUiState().

5.24.4.4 soundManager

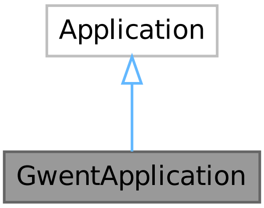
`SoundManager soundManager [private]`

SoundManager used to play Sound effects on click events.
Definition at line 85 of file GameBoardViewModel.java.
Referenced by GameBoardViewModel.getModel(), GameBoardViewModel.getSoundManager(), GameBoardViewModel.onBurnButto
GameBoardViewModel.onHornViewPressed(), GameBoardViewModel.onWeatherViewPressed(), and GameBoardViewModel.reset()
The documentation for this class was generated from the following file:

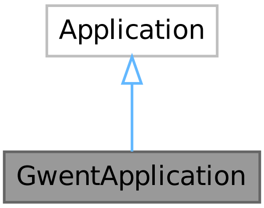
- gwent/ui/main/GameBoardViewModel.java

5.25 GwentApplication Class Reference

An `Application` encapsulating the Gwent app.
Inheritance diagram for GwentApplication:



Collaboration diagram for GwentApplication:



Public Member Functions

- void onCreate ()

  Called when the application is starting, before any activity, service, or receiver objects (excluding content providers) have been created.
- Single< UnitRepository > getRepository ()

  Returns the `repository` used to create, read, update and delete game state.

Static Public Member Functions

- static Single< UnitRepository > getRepository (@NonNull Context context)

Returns a `UnitRepository` used by the `GwentApplication` referenced by the given `Context`.

Private Attributes

- AppDatabase database

  `AppDatabase` used by this app.

Static Private Attributes

- static UnitRepository repository = null

  `UnitRepository` used for communication to `database`.

5.25.1 Detailed Description

An `Application` encapsulating the Gwent app.
This class is responsible for maintaining a Singleton `UnitRepository` connected to the `AppDatabase` of this app.
Definition at line 18 of file GwentApplication.java.

5.25.2 Member Function Documentation

5.25.2.1 getRepository() [1/2]

```
Single< UnitRepository > getRepository ( )
```

Returns the `repository` used to create, read, update and delete game state.
Lazily initializes `repository`, if not yet done so.

See also

#getRepository(Context)

Returns

A `UnitRepository` used to access game state.

Definition at line 62 of file GwentApplication.java.
References GwentApplication.database, and GwentApplication.repository.

5.25.2.2 getRepository() [2/2]

```
static Single< UnitRepository > getRepository (
            @NonNull Context context )  [static]
```

Returns a `UnitRepository` used by the `GwentApplication` referenced by the given `Context`.
Wrapper of `getRepository()`.

See also

#getRepository()

Parameters

context	`Context` of a `GwentApplication`.

Returns

A `UnitRepository` used by the given `GwentApplication` to access game state.

Definition at line 41 of file GwentApplication.java.
Referenced by CardNumberPickerAdapter.addSelectedUnits(), BurnDialogUseCase.burn(), ShowUnitsDialog.getDialog(), GameBoardViewModel.getRepository(), MainActivity.initializeViewModel(), MainActivity.onCreate(), AddCardDialog.onCreate(), and ResetDialogUseCase.reset().

5.25.2.3 onCreate()

`void onCreate ( )`

Called when the application is starting, before any activity, service, or receiver objects (excluding content providers) have been created.
Initializes `database`.
Definition at line 51 of file GwentApplication.java.
References GwentApplication.database.

5.25.3 Member Data Documentation

5.25.3.1 database

`AppDatabase database [private]`

`AppDatabase` used by this app.
Is initialized in `onCreate()`.

See also

#onCreate()

Definition at line 32 of file GwentApplication.java.
Referenced by GwentApplication.getRepository(), and GwentApplication.onCreate().

5.25.3.2 repository

`UnitRepository repository = null [static], [private]`

`UnitRepository` used for communication to `database`.
Is lazily initialized in `getRepository()` and provided as a Singleton.

See also

#getRepository()

#getRepository(Context)

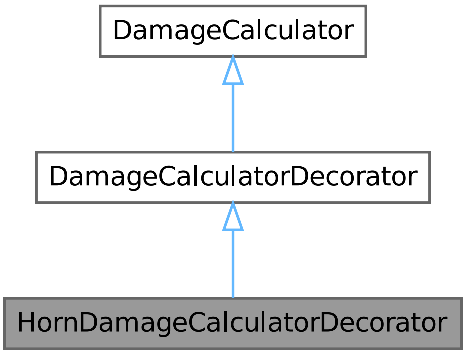
Definition at line 25 of file GwentApplication.java.
Referenced by GwentApplication.getRepository().
The documentation for this class was generated from the following file:

- gwent/GwentApplication.java

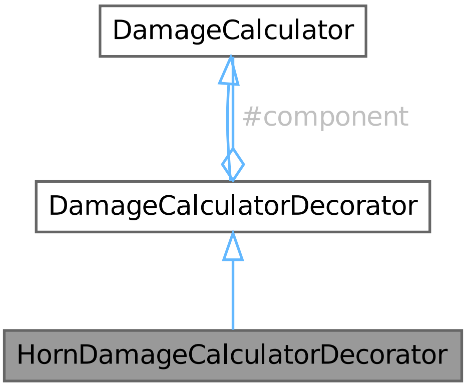
5.26 HornDamageCalculatorDecorator Class Reference

A DamageCalculator class responsible for calculating the horn buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::HORN ability that are not the unit the damage is calculated for or if there is a commander's horn in this row.

Inheritance diagram for HornDamageCalculatorDecorator:



Collaboration diagram for HornDamageCalculatorDecorator:



Public Member Functions

- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.

Package Functions

- HornDamageCalculatorDecorator (@NonNull DamageCalculator component, @NonNull List< Integer > unitIds)

  Constructor of a HornDamageCalculatorDecorator.

Package Functions inherited from DamageCalculatorDecorator

- DamageCalculatorDecorator (@NonNull DamageCalculator component)

  Constructor of a DamageCalculatorDecorator.

Private Member Functions

- boolean doubleDamage (int id)

  Decides whether the unit with the given id is affected by the horn buff.

Private Attributes

- final List< Integer > unitIds

  A List of Integers containing the ids of all units with the com.peternaggschga.gwent.data.Ability::HORN ability.

Additional Inherited Members

Protected Attributes inherited from DamageCalculatorDecorator

- final DamageCalculator component

  A DamageCalculator which is decorated by this decorator.

5.26.1 Detailed Description

A DamageCalculator class responsible for calculating the horn buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::HORN ability that are not the unit the damage is calculated for or if there is a commander's horn in this row.

Acts as a ConcreteDecorator in the implemented decorator pattern and should decorate a WeatherDamageCalculator, a BondDamageCalculatorDecorator or a MoralDamageCalculatorDecorator.

Definition at line 18 of file HornDamageCalculatorDecorator.java.

5.26.2 Constructor & Destructor Documentation

5.26.2.1 HornDamageCalculatorDecorator()

```
HornDamageCalculatorDecorator (
            @NonNull DamageCalculator component,
            @NonNull List< Integer > unitIds )  [package]
```

Constructor of a HornDamageCalculatorDecorator.

Should only be called by DamageCalculatorBuilder. The given damage calculator should be of type WeatherDamageCalculator, BondDamageCalculatorDecorator or MoralDamageCalculatorDecorator for correct damage calculation. If the respective row has a commander's horn, `null` must be an element of the given List.

Parameters

component	DamageCalculator that is being decorated by this decorator.
unitIds	List of Integers representing ids of units with the com.peternaggschga.gwent.data.Ability::HORN ability.

See also

DamageCalculatorBuilder

Definition at line 38 of file HornDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, and HornDamageCalculatorDecorator.unitIds.

5.26.3 Member Function Documentation

5.26.3.1 calculateDamage()

```
int calculateDamage (
             int id,
             @IntRange(from=0) int damage )
```

Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
Returns given damage times two if the unit is buffed by a commander's horn.

See also

#doubleDamage(int)

Parameters

id	Integer representing the UnitEntity::id of the unit whose (de-)buff damage is calculated.
damage	Integer representing the base-damage of the unit whose (de-)buff damage is calculated.

Returns

Integer representing the (de-)buffed damage of the unit.

Exceptions

IllegalArgumentException	When damage is negative.

Implements DamageCalculator.
Definition at line 63 of file HornDamageCalculatorDecorator.java.
References DamageCalculator.calculateDamage(), DamageCalculatorDecorator.component, and HornDamageCalculatorDecorator.d

5.26.3.2 doubleDamage()

```
boolean doubleDamage (
             int id ) [private]
```

Decides whether the unit with the given id is affected by the horn buff.

Parameters

id	Integer representing the UnitEntity::id of the unit that is being evaluated.

Returns

A Boolean value defining whether the horn buff affects the given unit.

Definition at line 49 of file HornDamageCalculatorDecorator.java.
References HornDamageCalculatorDecorator.unitIds.
Referenced by HornDamageCalculatorDecorator.calculateDamage(), and HornDamageCalculatorDecorator.isBuffed().

5.26.3.3 isBuffed()

```
Color isBuffed (
int id )
```

Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT. Units are shown as Color::BUFFED when they are affected by a horn buff, otherwise their Color is defined by #component.

Parameters

id	Integer representing the UnitEntity::id of the unit buff status is calculated.

Returns

Color representing whether the unit is buffed, de-buffed or not affected.

See also

Color

#doubleDamage(int)

Implements DamageCalculator.
Definition at line 82 of file HornDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, HornDamageCalculatorDecorator.doubleDamage(), and DamageCalculator.isBuffed().

5.26.4 Member Data Documentation

5.26.4.1 unitIds

```
final List<Integer> unitIds [private]
```

A List of Integers containing the ids of all units with the com.peternaggschga.gwent.data.Ability::HORN ability.
If a commander's horn is in the respective row, #unitIds contains `null`.
Definition at line 24 of file HornDamageCalculatorDecorator.java.
Referenced by HornDamageCalculatorDecorator.HornDamageCalculatorDecorator(), and HornDamageCalculatorDecorator.doubleDa
The documentation for this class was generated from the following file:

- gwent/domain/damage/HornDamageCalculatorDecorator.java

5.27 ImageViewSwitchAnimator Class Reference

A class providing functionality for changing the resource shown by an ImageView using a fading animation.
Collaboration diagram for ImageViewSwitchAnimator:

Static Public Member Functions

- static Completable animatedSwitch (@NonNull ImageView view, @DrawableRes int resId)

  Changes the resource of the given ImageView in an animation to the given resource id.

Static Private Member Functions

- static ImageView getOverlayView (@NonNull ImageView template)

  Returns a copy of the given template which is one layer above it.

5.27.1 Detailed Description

A class providing functionality for changing the resource shown by an ImageView using a fading animation.

See also

#animatedSwitch(ImageView, int)

Definition at line 20 of file ImageViewSwitchAnimator.java.

5.27.2 Member Function Documentation

5.27.2.1 animatedSwitch()

```
static Completable animatedSwitch (
            @NonNull ImageView view,
            @DrawableRes int resId )  [static]
```

Changes the resource of the given ImageView in an animation to the given resource id.
Animation is realized by creating a new ImageView on top of the given ImageView which is then faded out using the `fade_out` animation provided by Android. The operation is complete when the animation ends.

See also

android.R.anim::fade_out

Parameters

view	ImageView, whose resource is to be changed.
res← Id	Integer representing the drawable resource that the given view should show.

Returns

A Completable tracking operation status.

Definition at line 53 of file ImageViewSwitchAnimator.java.
References ImageViewSwitchAnimator.getOverlayView().
Referenced by MenuUiStateObserver.accept(), RowUiStateObserver.accept(), and FactionSwitchListener.onSharedPreferenceChang

5.27.2.2 getOverlayView()

```
static ImageView getOverlayView (
            @NonNull ImageView template )  [static], [private]
```

Returns a copy of the given template which is one layer above it.

Parameters

template	ImageView that is copied.

Returns

An ImageView similar to template but one layer in front of it.

Definition at line 27 of file ImageViewSwitchAnimator.java.
Referenced by ImageViewSwitchAnimator.animatedSwitch().
The documentation for this class was generated from the following file:

- gwent/ui/main/ImageViewSwitchAnimator.java

5.28 IndicatorManager Class Reference

A class managing the indicator views shown in the bottom bar of the `IntroductionActivity`.
Collaboration diagram for IndicatorManager:

Package Functions

- IndicatorManager (@NonNull Window window)

  Constructor of an `IndicatorManager` for the given `Window`.
- void updateIndicators (@IntRange(from=0, to=4) int position)

  Updates the indicator views according to the given new position.

Private Attributes

- final ImageView[ ] indicators = new ImageView[5]

  Array of `ImageView`s that contains the indicator views.
- int currentView = 0

  `Integer` referencing the position, i.e.

5.28.1 Detailed Description

A class managing the indicator views shown in the bottom bar of the `IntroductionActivity`.
The managed views show the user how many pages there are and which one they are on.
Definition at line 15 of file IndicatorManager.java.

5.28.2 Constructor & Destructor Documentation

5.28.2.1 IndicatorManager()

```
IndicatorManager (
            @NonNull Window window )  [package]
```

Constructor of an `IndicatorManager` for the given `Window`.
Initializes values in `indicators`.

Parameters

window	`Window` containing the indicator views.

See also

Window::findViewById(int)

R.id::introduction_indicator_0

R.id::introduction_indicator_1

R.id::introduction_indicator_2

R.id::introduction_indicator_3

R.id::introduction_indicator_4

Definition at line 43 of file IndicatorManager.java.
References IndicatorManager.indicators.

5.28.3 Member Function Documentation

5.28.3.1 updateIndicators()

```
void updateIndicators (
            @IntRange(from=0, to=4) int position )  [package]
```

Updates the indicator views according to the given new position.
Sets the `currentView` to `R.drawable#indicator_unselected` and the view at the new position to `R.drawable#indicator_selected`. Saves the given position in `currentView`.

Parameters

position	`Integer` referencing the page represented by the managed indicators.

Exceptions

ArrayIndexOutOfBoundsException	When position is not in [0, 4].

Definition at line 59 of file IndicatorManager.java.
References IndicatorManager.currentView, and IndicatorManager.indicators.

5.28.4 Member Data Documentation

5.28.4.1 currentView

```
int currentView = 0  [private]
```

`Integer` referencing the position, i.e.
the index in `indicators`, of the view that is currently shown as selected. Is set in `updateIndicators(int)`.

See also

#updateIndicators(int)

Definition at line 30 of file IndicatorManager.java.
Referenced by IndicatorManager.updateIndicators().

5.28.4.2 indicators

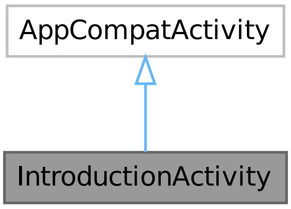
```
final ImageView [] indicators = new ImageView[5]  [private]
```

Array of `ImageView`s that contains the indicator views.
Definition at line 20 of file IndicatorManager.java.
Referenced by IndicatorManager.IndicatorManager(), and IndicatorManager.updateIndicators().
The documentation for this class was generated from the following file:

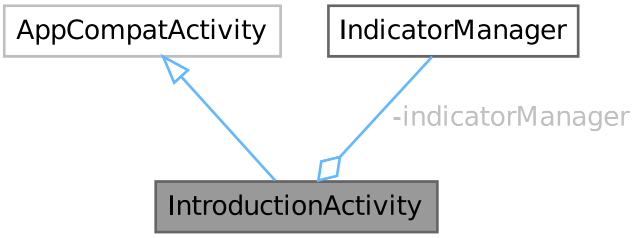
- gwent/ui/introduction/IndicatorManager.java

5.29 IntroductionActivity Class Reference

An `AppCompatActivity` that gives the user an introduction into the usage of the application.
Inheritance diagram for IntroductionActivity:



Collaboration diagram for IntroductionActivity:



Protected Member Functions

- void onCreate (Bundle savedInstanceState)

  Sets layout to `R.layout#activity_introduction`, initializes `indicatorManager`, sets listeners on the buttons, and creates a new `OnBackPressedCallback` that switches to the previous page, if it is not the first one.

Private Attributes

- IndicatorManager indicatorManager

  `IndicatorManager` used to update the progress indicators according to the currently shown page.

5.29.1 Detailed Description

An `AppCompatActivity` that gives the user an introduction into the usage of the application.
Is called when the app is first started (as tracked by the `androidx.preference.Preference` at key `R.string#preference_first_use_key`.
Definition at line 20 of file IntroductionActivity.java.

5.29.2 Member Function Documentation

5.29.2.1 onCreate()

```
void onCreate (
             Bundle savedInstanceState ) [protected]
```

Sets layout to `R.layout#activity_introduction`, initializes `indicatorManager`, sets listeners on the buttons, and creates a new `OnBackPressedCallback` that switches to the previous page, if it is not the first one.

Parameters

savedInstanceState	If the activity is being re-initialized after previously being shut down then this Bundle contains the data it most recently supplied in `onSaveInstanceState`. ***Note: Otherwise it is null.***

Called when enabled and the user clicks on the back-button. Switches the `ViewPager2` containing the `IntroductionFragment`s to the last position.

Updates the `IntroductionActivity#indicatorManager`, changes button-visibility and whether the callback for onBackPressed is active.

See also

IndicatorManager::updateIndicators(int)

Parameters

position	Position index of the new selected page.

Definition at line 37 of file IntroductionActivity.java.
References IntroductionActivity.indicatorManager.

5.29.3 Member Data Documentation

5.29.3.1 indicatorManager

```
IndicatorManager indicatorManager [private]
```

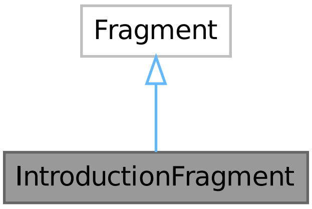
`IndicatorManager` used to update the progress indicators according to the currently shown page.
Is initialized in `onCreate(Bundle)`.
Definition at line 25 of file IntroductionActivity.java.
Referenced by IntroductionActivity.onCreate().
The documentation for this class was generated from the following file:

- gwent/ui/introduction/IntroductionActivity.java

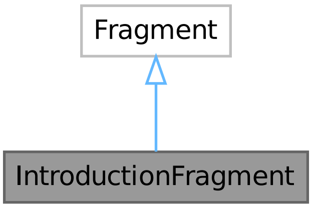
5.30 IntroductionFragment Class Reference

A `Fragment` containing an `ImageView` and a `TextView` conveying introductory information.

Inheritance diagram for IntroductionFragment:



Collaboration diagram for IntroductionFragment:



Public Member Functions

- View onCreateView (@NonNull LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)

  Inflates the `R.layout#fragment_introduction` layout and sets text and image.

Static Public Member Functions

- static IntroductionFragment newInstance (@IntRange(from=0, to=PAGES_COUNT - 1) int index)

  Creates a new `IntroductionFragment` showing the page defined by the given index.

Static Public Attributes

- static final int PAGES_COUNT = 5

  `Integer` constant defining the number of possible pages represented by this fragment.

Static Private Attributes

- static final String ARG_SECTION_NUMBER = "section_number"

  `String` constant defining the argument key used to communicate the requested page to the fragment.
- static final int[ ] imageIds

  `Integer` array constant containing the resource ids for all introductory images.
- static final int[ ] stringIds

  `Integer` array constant containing the resource ids for all introductory texts.

5.30.1 Detailed Description

A `Fragment` containing an `ImageView` and a `TextView` conveying introductory information.
Definition at line 21 of file IntroductionFragment.java.

5.30.2 Member Function Documentation

5.30.2.1 newInstance()

```
static IntroductionFragment newInstance (
            @IntRange(from=0, to=PAGES_COUNT - 1) int index )  [static]
```

Creates a new `IntroductionFragment` showing the page defined by the given index.
Factory method of `IntroductionFragment`.

Parameters

index	`Integer` defining the page that the created fragment shows.

Returns

An `IntroductionFragment` showing the page with the given index.

Exceptions

IllegalArgumentException	When the given index is not in [0, `PAGES_COUNT` - 1].

Definition at line 50 of file IntroductionFragment.java.
References IntroductionFragment.ARG_SECTION_NUMBER, and IntroductionFragment.PAGES_COUNT.
Referenced by SectionsPagerAdapter.createFragment().

5.30.2.2 onCreateView()

```
View onCreateView (
            @NonNull LayoutInflater inflater,
            ViewGroup container,
            Bundle savedInstanceState )
```

Inflates the `R.layout#fragment_introduction` layout and sets text and image.

Parameters

inflater	The LayoutInflater object that can be used to inflate any views in the fragment,
container	If non-null, this is the parent view that the fragment's UI should be attached to. The fragment should not add the view itself, but this can be used to generate the LayoutParams of the view.
savedInstanceState	If non-null, this fragment is being re-constructed from a previous saved state as given here.

Returns

A `View` that is the root of the newly inflated layout.

Definition at line 75 of file IntroductionFragment.java.
References IntroductionFragment.ARG_SECTION_NUMBER, IntroductionFragment.imageIds, and IntroductionFragment.stringIds.

5.30.3 Member Data Documentation

5.30.3.1 ARG_SECTION_NUMBER

```
final String ARG_SECTION_NUMBER = "section_number"  [static], [private]
```

`String` constant defining the argument key used to communicate the requested page to the fragment.
Definition at line 30 of file IntroductionFragment.java.
Referenced by IntroductionFragment.newInstance(), and IntroductionFragment.onCreateView().

5.30.3.2 imageIds

```
final int [] imageIds  [static], [private]
```

Initial value:

```
= {R.drawable.introduction_1, R.drawable.introduction_2,
            R.drawable.introduction_3, R.drawable.introduction_4, R.drawable.introduction_5}
```

`Integer` array constant containing the resource ids for all introductory images.
Definition at line 34 of file IntroductionFragment.java.
Referenced by IntroductionFragment.onCreateView().

5.30.3.3 PAGES_COUNT

```
final int PAGES_COUNT = 5  [static]
```

`Integer` constant defining the number of possible pages represented by this fragment.
Is equivalent to the length of `stringIds` and `imageIds`.
Definition at line 26 of file IntroductionFragment.java.
Referenced by SectionsPagerAdapter.getItemCount(), and IntroductionFragment.newInstance().

5.30.3.4 stringIds

```
final int [] stringIds  [static], [private]
```

Initial value:

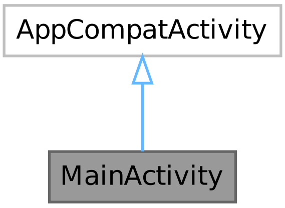
```
= {R.string.introduction_page1, R.string.introduction_page2,
            R.string.introduction_page3, R.string.introduction_page4, R.string.introduction_page5}
```

`Integer` array constant containing the resource ids for all introductory texts.
Definition at line 39 of file IntroductionFragment.java.
Referenced by IntroductionFragment.onCreateView().
The documentation for this class was generated from the following file:

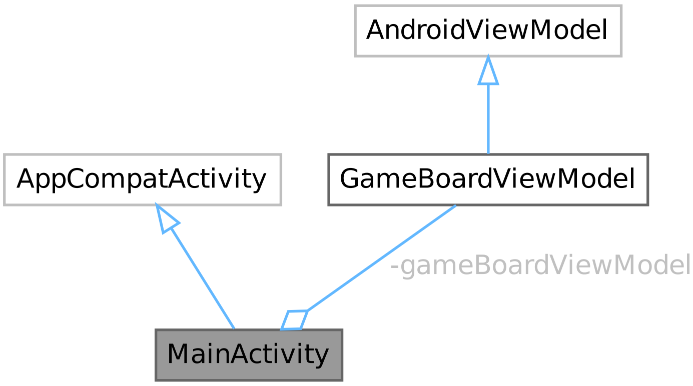
- gwent/ui/introduction/IntroductionFragment.java

5.31 MainActivity Class Reference

An `AppCompatActivity` that is called on startup and that encapsulates the main view onto the game board.
Inheritance diagram for MainActivity:



Collaboration diagram for MainActivity:



Public Member Functions

- void onWindowFocusChanged (boolean hasFocus)

  Called when the focus of this activities `android.view.Window` changes.

Protected Member Functions

- void onCreate (@Nullable Bundle savedInstanceState)

  Sets the theme and layout, initializes `gameBoardViewModel` and `factionSwitchListener` and sets listeners for some menu buttons.
- void onResume ()

  Called when the application is resumed after a pause or on startup.
- void onDestroy ()

  Called when the activity is destroyed.

Private Member Functions

- void initializeViewModel ()

  Initializes the `View`s in this activity to communicate with the `gameBoardViewModel`.
- void inflateFactionPopup ()

  Shows a new `ChangeFactionDialog` enabling the user to choose the preferred theme.
- void inflateCoinFlipPopup ()

  Shows a new `CoinFlipDialog` and plays a coin-flip sound using `SoundManager#playCoinSound()`.

Private Attributes

- final CompositeDisposable disposables = new CompositeDisposable()

  `CompositeDisposable` used to store all `io.reactivex.rxjava3.disposables.Disposable`s, this activity might create to allow for their disposal in `onDestroy()`.
- SharedPreferences.OnSharedPreferenceChangeListener factionSwitchListener

  `SharedPreferences.OnSharedPreferenceChangeListener` that is called when faction-layout, i.e., the `androidx.preference.Preference` at the key defined by `com.peternaggschga.gwent.ui.main.FactionSwit` is changed.
- GameBoardViewModel gameBoardViewModel

  `GameBoardViewModel` holding the ui state of this activity.

5.31.1 Detailed Description

An AppCompatActivity that is called on startup and that encapsulates the main view onto the game board.
Definition at line 46 of file MainActivity.java.

5.31.2 Member Function Documentation

5.31.2.1 inflateCoinFlipPopup()

```
void inflateCoinFlipPopup ( )  [private]
```

Shows a new CoinFlipDialog and plays a coin-flip sound using SoundManager#playCoinSound().
Definition at line 274 of file MainActivity.java.
References MainActivity.gameBoardViewModel, and GameBoardViewModel.getSoundManager().
Referenced by MainActivity.onCreate().

5.31.2.2 inflateFactionPopup()

```
void inflateFactionPopup ( )  [private]
```

Shows a new ChangeFactionDialog enabling the user to choose the preferred theme.
The decision is saved in the preference at the key FactionSwitchListener#THEME_PREFERENCE_KEY and if the user opted to reset on faction switch (i.e., the preference at R.string#preference_key_faction_reset is true), GameBoardViewModel#onFactionSwitchReset(Context) is called.
Definition at line 250 of file MainActivity.java.
References MainActivity.disposables, MainActivity.gameBoardViewModel, and GameBoardViewModel.onFactionSwitchReset().
Referenced by MainActivity.onCreate().

5.31.2.3 initializeViewModel()

```
void initializeViewModel ( )  [private]
```

Initializes the Views in this activity to communicate with the gameBoardViewModel.
Sets android.widget.Button.OnClickListener for the weather-, horn-, and card-views of each row as well as listeners for the reset-, weather- and burn-buttons.
Definition at line 191 of file MainActivity.java.
References MainActivity.disposables, MainActivity.gameBoardViewModel, GameBoardViewModel.getMenuUiState(), RowUiStateObserver.getObserver(), GwentApplication.getRepository(), GameBoardViewModel.getRowUiState(), GameBoardViewModel.getSoundManager(), GameBoardViewModel.onBurnButtonPressed(), GameBoardViewModel.onHornViewPr GameBoardViewModel.onResetButtonPressed(), GameBoardViewModel.onWeatherButtonPressed(), and GameBoardViewModel.onWeatherViewPressed().
Referenced by MainActivity.onCreate().

5.31.2.4 onCreate()

```
void onCreate (
            @Nullable Bundle savedInstanceState )  [protected]
```

Sets the theme and layout, initializes gameBoardViewModel and factionSwitchListener and sets listeners for some menu buttons.
If the application is started for the first time (as tracked by the preference at key R.string#preference_first_use_key) the IntroductionActivity is called first. The theme is set according to the preference at the key specified by FactionSwitchListener#THEME_PREFERENCE_KEY. The layout is set to R.layout#activity_main. android.widget.Button.OnClickListeners are set for the buttons referenced by R.id#factionButton, R.id#coinButton, and R.id#settingsButton.

Parameters

savedInstanceState	If the activity is being re-initialized after previously being shut down then this Bundle contains the data it most recently supplied in onSaveInstanceState. ***Note: Otherwise it is null.***

Definition at line 83 of file MainActivity.java.
References MainActivity.disposables, MainActivity.factionSwitchListener, MainActivity.gameBoardViewModel,

FactionSwitchListener.getListener(), GameBoardViewModel.getModel(), GwentApplication.getRepository(), MainActivity.inflateCoinFlipPopup(), MainActivity.inflateFactionPopup(), MainActivity.initializeViewModel(), and FactionSwitchListener.setTheme().

5.31.2.5 onDestroy()

```
void onDestroy ( )  [protected]
```

Called when the activity is destroyed.
Disposes and clears all `io.reactivex.rxjava3.disposables.Disposable`s in `disposables`.
Definition at line 180 of file MainActivity.java.
References MainActivity.disposables.

5.31.2.6 onResume()

```
void onResume ( )  [protected]
```

Called when the application is resumed after a pause or on startup.
Sets the background image according to the preference at the key referenced by `R.string#preference_key_design`.
Definition at line 124 of file MainActivity.java.

5.31.2.7 onWindowFocusChanged()

```
void onWindowFocusChanged (
            boolean hasFocus )
```

Called when the focus of this activities `android.view.Window` changes.
Hides system ui and sets flags to keep the screen on when the window is in focus for more than 250 ms.

Parameters

hasFocus	Whether the window of this activity has focus.

Definition at line 154 of file MainActivity.java.

5.31.3 Member Data Documentation

5.31.3.1 disposables

```
final CompositeDisposable disposables = new CompositeDisposable()  [private]
```

`CompositeDisposable` used to store all `io.reactivex.rxjava3.disposables.Disposable`s, this activity might create to allow for their disposal in `onDestroy()`.
Definition at line 52 of file MainActivity.java.
Referenced by MainActivity.inflateFactionPopup(), MainActivity.initializeViewModel(), MainActivity.onCreate(), and MainActivity.onDestroy().

5.31.3.2 factionSwitchListener

```
SharedPreferences.OnSharedPreferenceChangeListener factionSwitchListener  [private]
```

`SharedPreferences.OnSharedPreferenceChangeListener` that is called when faction-layout, i.e., the `androidx.preference.Preference` at the key defined by `com.peternaggschga.gwent.ui.main.FactionSw` is changed.
Reference must be kept (even if not used) to avoid garbage collection of the registered listener (see `here` for more information).
Definition at line 61 of file MainActivity.java.
Referenced by MainActivity.onCreate().

5.31.3.3 gameBoardViewModel

```
GameBoardViewModel gameBoardViewModel  [private]
```

`GameBoardViewModel` holding the ui state of this activity.
Definition at line 66 of file MainActivity.java.

Referenced by MainActivity.inflateCoinFlipPopup(), MainActivity.inflateFactionPopup(), MainActivity.initializeViewModel(), and MainActivity.onCreate().

The documentation for this class was generated from the following file:

- gwent/ui/main/MainActivity.java

5.32 MenuUiState Class Reference

A data class encapsulating the visible state of the menu in the main view of the application.

Collaboration diagram for MenuUiState:

Public Member Functions

- MenuUiState (@IntRange(from=0) int damage, boolean reset, boolean weather, boolean burn)

  Constructor of a MenuUiState encapsulating the given data.
- int getDamage ()

  Returns the summed-up damage of all units on the game board.
- boolean isReset ()

  Returns whether the reset button is clickable.
- boolean isWeather ()

  Returns whether the weather button is clickable.
- boolean isBurn ()

  Returns whether the burn button is clickable.
- boolean equals (Object o)

Private Attributes

- final int damage

  Defines the summed-up damage of all units on the game board.
- final boolean reset

  Defines whether the reset button is clickable.
- final boolean weather

  Defines whether the weather button is clickable.
- final boolean burn

  Defines whether the burn button is clickable.

5.32.1 Detailed Description

A data class encapsulating the visible state of the menu in the main view of the application.

Definition at line 10 of file MenuUiState.java.

5.32.2 Constructor & Destructor Documentation

5.32.2.1 MenuUiState()

```
MenuUiState (
            @IntRange(from=0) int damage,
            boolean reset,
            boolean weather,
            boolean burn )
```

Constructor of a MenuUiState encapsulating the given data.

Parameters

damage	Integer representing the summed-up damage of all units.
reset	Boolean defining whether the reset button is clickable.
weather	Boolean defining whether the weather button is clickable.
burn	Boolean defining whether the burn button is clickable.

Exceptions

IllegalArgumentException	When damage is negative.

Definition at line 41 of file MenuUiState.java.
References MenuUiState.burn, MenuUiState.damage, MenuUiState.reset, and MenuUiState.weather.

5.32.3 Member Function Documentation

5.32.3.1 equals()

```
boolean equals (
            Object o )
```

Definition at line 89 of file MenuUiState.java.

5.32.3.2 getDamage()

```
int getDamage ( )
```

Returns the summed-up damage of all units on the game board.

Returns

An Integer representing the summed-up damage of all units.

Definition at line 57 of file MenuUiState.java.
References MenuUiState.damage.

5.32.3.3 isBurn()

```
boolean isBurn ( )
```

Returns whether the burn button is clickable.

Returns

A Boolean defining whether the burn button is clickable.

Definition at line 84 of file MenuUiState.java.
References MenuUiState.burn.

5.32.3.4 isReset()

```
boolean isReset ( )
```

Returns whether the reset button is clickable.

Returns

A Boolean defining whether the reset button is clickable.

Definition at line 66 of file MenuUiState.java.
References MenuUiState.reset.

5.32.3.5 isWeather()

```
boolean isWeather ( )
```

Returns whether the weather button is clickable.

Returns

A Boolean defining whether the weather button is clickable.

Definition at line 75 of file MenuUiState.java.
References MenuUiState.weather.

5.32.4 Member Data Documentation

5.32.4.1 burn

```
final boolean burn [private]
```

Defines whether the burn button is clickable.
Definition at line 30 of file MenuUiState.java.
Referenced by MenuUiState.MenuUiState(), and MenuUiState.isBurn().

5.32.4.2 damage

```
final int damage [private]
```

Defines the summed-up damage of all units on the game board.
Definition at line 15 of file MenuUiState.java.
Referenced by MenuUiState.MenuUiState(), and MenuUiState.getDamage().

5.32.4.3 reset

```
final boolean reset [private]
```

Defines whether the reset button is clickable.
Definition at line 20 of file MenuUiState.java.
Referenced by MenuUiState.MenuUiState(), and MenuUiState.isReset().

5.32.4.4 weather

```
final boolean weather [private]
```

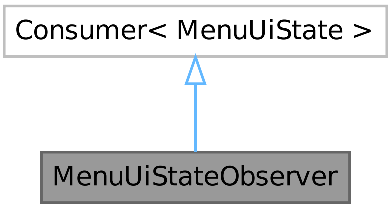
Defines whether the weather button is clickable.
Definition at line 25 of file MenuUiState.java.
Referenced by MenuUiState.MenuUiState(), and MenuUiState.isWeather().
The documentation for this class was generated from the following file:

- gwent/ui/main/MenuUiState.java

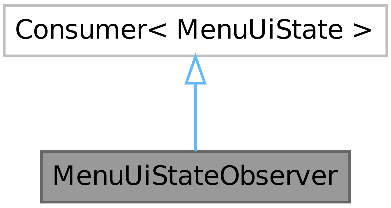
5.33 MenuUiStateObserver Class Reference

An observer class responsible for updating the menu views when notified, i.e., when a new MenuUiState is produced by the subscribed Flowable.

Inheritance diagram for MenuUiStateObserver:



Collaboration diagram for MenuUiStateObserver:



Public Member Functions

- MenuUiStateObserver (@NonNull TextView damageView, @NonNull ImageButton resetButton, @NonNull ImageButton weatherButton, @NonNull ImageButton burnButton)

  Constructor of a MenuUiStateObserver updating the given views when #onChanged() is called.
- void accept (@NonNull MenuUiState menuUiState)

  Changes value in #damageView as well as appearance and clickable status of #resetButton, #weatherButton, and #burnButton.

Private Attributes

- final TextView damageView

  A TextView showing the user the summed-up damage of all units, i.e., MenuUiState::damage.
- final ImageButton resetButton

  An ImageButton responsible for resetting the whole board, i.e., clearing weather and commander's horn and deleting all units.
- final ImageButton weatherButton

  An ImageButton responsible for clearing all weather effects.
- final ImageButton burnButton

  An ImageButton responsible for deleting the units with the highest damage.

5.33.1 Detailed Description

An observer class responsible for updating the menu views when notified, i.e., when a new MenuUiState is produced by the subscribed Flowable.

See also

MenuUiState

Definition at line 18 of file MenuUiStateObserver.java.

5.33.2 Constructor & Destructor Documentation

5.33.2.1 MenuUiStateObserver()

```
MenuUiStateObserver (
            @NonNull TextView damageView,
            @NonNull ImageButton resetButton,
            @NonNull ImageButton weatherButton,
            @NonNull ImageButton burnButton )
```

Constructor of a MenuUiStateObserver updating the given views when #onChanged() is called.

Parameters

damageView	TextView showing the summed-up damage of all units.
resetButton	ImageButton used to reset the whole game board.
weatherButton	ImageButton used to reset the weather debuff in all rows.
burnButton	ImageButton used to remove the strongest units from the game board.

Definition at line 55 of file MenuUiStateObserver.java.
References MenuUiStateObserver.burnButton, MenuUiStateObserver.damageView, MenuUiStateObserver.resetButton, and MenuUiStateObserver.weatherButton.

5.33.3 Member Function Documentation

5.33.3.1 accept()

```
void accept (
            @NonNull MenuUiState menuUiState )
```

Changes value in #damageView as well as appearance and clickable status of #resetButton, #weatherButton, and #burnButton.
Appearance changes are animated using ImageViewSwitchAnimator.

Parameters

menuUiState	MenuUiState representing the updated state of the menu.

See also

ImageViewSwitchAnimator

Definition at line 72 of file MenuUiStateObserver.java.
References ImageViewSwitchAnimator.animatedSwitch(), MenuUiStateObserver.burnButton, MenuUiStateObserver.damageView, MenuUiStateObserver.resetButton, and MenuUiStateObserver.weatherButton.

5.33.4 Member Data Documentation

5.33.4.1 burnButton

```
final ImageButton burnButton  [private]
```

An ImageButton responsible for deleting the units with the highest damage.
Is not clickable and gray when MenuUiState::burn is `false`.
Definition at line 45 of file MenuUiStateObserver.java.
Referenced by MenuUiStateObserver.MenuUiStateObserver(), and MenuUiStateObserver.accept().

5.33.4.2 damageView

`final TextView damageView [private]`
A TextView showing the user the summed-up damage of all units, i.e., MenuUiState::damage.
Definition at line 23 of file MenuUiStateObserver.java.
Referenced by MenuUiStateObserver.MenuUiStateObserver(), and MenuUiStateObserver.accept().

5.33.4.3 resetButton

`final ImageButton resetButton [private]`
An ImageButton responsible for resetting the whole board, i.e., clearing weather and commander's horn and deleting all units.
Is not clickable and gray when MenuUiState::reset is `false`.
Definition at line 31 of file MenuUiStateObserver.java.
Referenced by MenuUiStateObserver.MenuUiStateObserver(), and MenuUiStateObserver.accept().

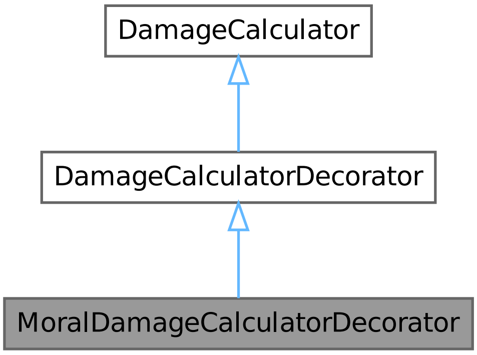
5.33.4.4 weatherButton

`final ImageButton weatherButton [private]`
An ImageButton responsible for clearing all weather effects.
Is not clickable and gray when MenuUiState::weather is `false`.
Definition at line 38 of file MenuUiStateObserver.java.
Referenced by MenuUiStateObserver.MenuUiStateObserver(), and MenuUiStateObserver.accept().
The documentation for this class was generated from the following file:

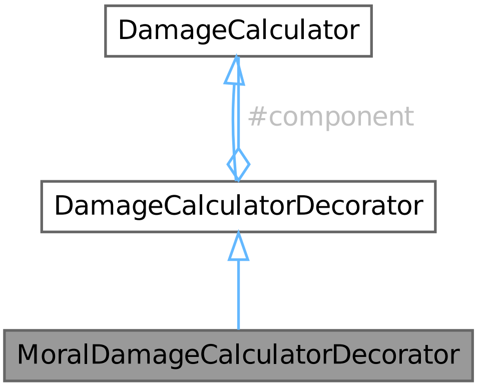
- gwent/ui/main/MenuUiStateObserver.java

5.34 MoralDamageCalculatorDecorator Class Reference

A DamageCalculator class responsible for calculating the moral buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability that are not the unit the damage is calculated for.
Inheritance diagram for MoralDamageCalculatorDecorator:



Collaboration diagram for MoralDamageCalculatorDecorator:



Public Member Functions

- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.

Package Functions

- MoralDamageCalculatorDecorator (@NonNull DamageCalculator component, @NonNull List< Integer > unitIds)

  Constructor of a MoralDamageCalculatorDecorator.

Package Functions inherited from DamageCalculatorDecorator

- DamageCalculatorDecorator (@NonNull DamageCalculator component)

  Constructor of a DamageCalculatorDecorator.

Private Attributes

- final List< Integer > unitIds

  A List of Integers containing the ids of all units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability.

Additional Inherited Members

Protected Attributes inherited from DamageCalculatorDecorator

- final DamageCalculator component

  A DamageCalculator which is decorated by this decorator.

5.34.1 Detailed Description

A DamageCalculator class responsible for calculating the moral buff if necessary, i.e., if there are units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability that are not the unit the damage is calculated for.

Acts as a ConcreteDecorator in the implemented decorator pattern and should decorate a WeatherDamage↩ Calculator or a BondDamageCalculatorDecorator for correct damage calculation.
Definition at line 17 of file MoralDamageCalculatorDecorator.java.

5.34.2 Constructor & Destructor Documentation

5.34.2.1 MoralDamageCalculatorDecorator()

```
MoralDamageCalculatorDecorator (
            @NonNull DamageCalculator component,
            @NonNull List< Integer > unitIds )  [package]
```

Constructor of a MoralDamageCalculatorDecorator.
Should only be called by DamageCalculatorBuilder. The given damage calculator should be of type Weather↩ DamageCalculator or BondDamageCalculatorDecorator for correct damage calculation. The given List must not contain `null`.

Parameters

component	DamageCalculator that is being decorated by this decorator.
unitIds	List of Integers representing ids of units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability.

Exceptions

IllegalArgumentException	When unitIds contains null values.

See also

DamageCalculatorBuilder

Definition at line 36 of file MoralDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, and MoralDamageCalculatorDecorator.unitIds.

5.34.3 Member Function Documentation

5.34.3.1 calculateDamage()

```
int calculateDamage (
            int id,
            @IntRange(from=0) int damage )
```

Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
Returns the given damage plus the number of moral boosts. If the unit itself has the com.peternaggschga.gwent.data.Ability::MORAL_I ability, it is boosted one time less.

Parameters

id	Integer representing the UnitEntity::id of the unit whose (de-)buff damage is calculated.
damage	Integer representing the base-damage of the unit whose (de-)buff damage is calculated.

Returns

Integer representing the (de-)buffed damage of the unit.

Exceptions

IllegalArgumentException	When damage is negative.

Implements DamageCalculator.
Definition at line 56 of file MoralDamageCalculatorDecorator.java.
References DamageCalculator.calculateDamage(), DamageCalculatorDecorator.component, and MoralDamageCalculatorDecorator.u

5.34.3.2 isBuffed()

```
Color isBuffed (
            int id )
```

Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT. Units are shown as Color::BUFFED when they are affected by a moral boost buff, otherwise their Color is defined by #component.

Parameters

id	Integer representing the UnitEntity::id of the unit buff status is calculated.

Returns

Color representing whether the unit is buffed, de-buffed or not affected.

See also

Color

Implements DamageCalculator.
Definition at line 75 of file MoralDamageCalculatorDecorator.java.
References DamageCalculatorDecorator.component, DamageCalculator.isBuffed(), and MoralDamageCalculatorDecorator.unitIds.

5.34.4 Member Data Documentation

5.34.4.1 unitIds

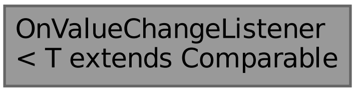
```
final List<Integer> unitIds  [private]
```

A List of Integers containing the ids of all units with the com.peternaggschga.gwent.data.Ability::MORAL_BOOST ability.
Definition at line 22 of file MoralDamageCalculatorDecorator.java.
Referenced by MoralDamageCalculatorDecorator.MoralDamageCalculatorDecorator(), MoralDamageCalculatorDecorator.calculateDa
and MoralDamageCalculatorDecorator.isBuffed().
The documentation for this class was generated from the following file:

- gwent/domain/damage/MoralDamageCalculatorDecorator.java

5.35 OnValueChangeListener< T extends Comparable Interface Template Reference

An interface defining a callback for changes of the current value.
Collaboration diagram for OnValueChangeListener< T extends Comparable:



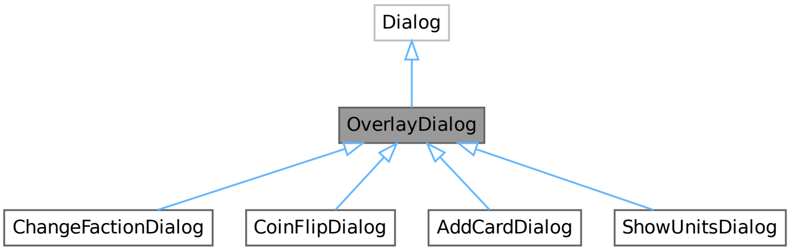
5.35.1 Detailed Description

An interface defining a callback for changes of the current value.
Definition at line 211 of file ValuePicker.java.
The documentation for this interface was generated from the following file:

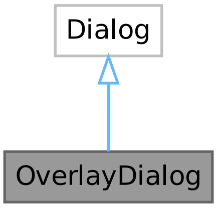
- gwent/ui/dialogs/addcard/ValuePicker.java

5.36 OverlayDialog Class Reference

A Dialog class which is used for popups that are shown on top of the calling Activity.
Inheritance diagram for OverlayDialog:



Collaboration diagram for OverlayDialog:



Protected Member Functions

- OverlayDialog (@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)

  Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
- OverlayDialog (@NonNull Context context, @LayoutRes int layout)

  Constructor of an OverlayDialog in the given Context and with the given layout.
- void onCreate (Bundle savedInstanceState)

  Initializes #layout using #setContentView(int).

Static Package Attributes

- static final int NO_DISMISS_VIEW = -1

  Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

Private Attributes

- final int layout

  Integer referencing the layout shown by the Dialog.
- final int dismissViewId

  Integer referencing a view that can be clicked to dismiss the Dialog.

Static Private Attributes

- static final ColorDrawable BACKGROUND = new ColorDrawable(Color.TRANSPARENT)

  ColorDrawable shown as the background of the Dialog (above the calling Activity).

5.36.1 Detailed Description

A Dialog class which is used for popups that are shown on top of the calling Activity.
The Dialog uses the layout specified in #layout with the #BACKGROUND color as the background. The view specified by #dismissViewId may be clicked to dismiss the dialog.
Definition at line 22 of file OverlayDialog.java.

5.36.2 Constructor & Destructor Documentation

5.36.2.1 OverlayDialog() [1/2]

```
OverlayDialog (
             @NonNull Context context,
             @LayoutRes int layout,
             @IdRes int dismissViewId )  [protected]
```

Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
When dismissViewId is equal to #NO_CANCEL_VIEW, the Dialog is not cancelable by clicking on a certain view.

Parameters

context	Context of the created OverlayDialog.
layout	Integer referencing the layout shown by the created OverlayDialog.
dismissViewId	Integer referencing the cancel view or #NO_CANCEL_VIEW.

Definition at line 54 of file OverlayDialog.java.
References OverlayDialog.dismissViewId, and OverlayDialog.layout.

5.36.2.2 OverlayDialog() [2/2]

```
OverlayDialog (
             @NonNull Context context,
             @LayoutRes int layout )  [protected]
```

Constructor of an OverlayDialog in the given Context and with the given layout.
Wrapper of #OverlayDialog(Context, int, int).

See also

#OverlayDialog(Context, int, int)

Parameters

context	Context of the created OverlayDialog.
layout	Integer referencing the layout shown by the created OverlayDialog.

Definition at line 68 of file OverlayDialog.java.

References OverlayDialog.layout, and OverlayDialog.NO_DISMISS_VIEW.

5.36.3 Member Function Documentation

5.36.3.1 onCreate()

```
void onCreate (
             Bundle savedInstanceState )  [protected]
```

Initializes #layout using #setContentView(int).
Sets View.OnClickListener canceling the dialog for the View referenced by #dismissViewId if it is set. Switches whether the Dialog is cancelable using #setCancelable().

Parameters

savedInstanceState	If this dialog is being reinitialized after the hosting activity was previously shut down, holds the result from the most recent call to `onSaveInstanceState`, or null if this is the first time.

See also

#setContentView(int)

#setCancelable(boolean)

Reimplemented in AddCardDialog, ShowUnitsDialog, ChangeFactionDialog, and CoinFlipDialog.
Definition at line 85 of file OverlayDialog.java.
References OverlayDialog.BACKGROUND, OverlayDialog.dismissViewId, OverlayDialog.layout, and OverlayDialog.NO_DISMISS_VI

5.36.4 Member Data Documentation

5.36.4.1 BACKGROUND

```
final ColorDrawable BACKGROUND = new ColorDrawable(Color.TRANSPARENT)  [static], [private]
```

ColorDrawable shown as the background of the Dialog (above the calling Activity).
Definition at line 33 of file OverlayDialog.java.
Referenced by OverlayDialog.onCreate().

5.36.4.2 dismissViewId

```
final int dismissViewId  [private]
```

Integer referencing a view that can be clicked to dismiss the Dialog.
Should be equal to #NO_CANCEL_VIEW, when the Dialog is not cancelable.

See also

#NO_DISMISS_VIEW

Definition at line 45 of file OverlayDialog.java.
Referenced by OverlayDialog.OverlayDialog(), and OverlayDialog.onCreate().

5.36.4.3 layout

```
final int layout  [private]
```

Integer referencing the layout shown by the Dialog.
Definition at line 38 of file OverlayDialog.java.
Referenced by OverlayDialog.OverlayDialog(), OverlayDialog.OverlayDialog(), and OverlayDialog.onCreate().

5.36.4.4 NO_DISMISS_VIEW

```
final int NO_DISMISS_VIEW = -1  [static], [package]
```

Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

See also

#dismissViewId

Definition at line 28 of file OverlayDialog.java.
Referenced by OverlayDialog.OverlayDialog(), and OverlayDialog.onCreate().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/OverlayDialog.java

5.37 RemoveUnitsUseCase Class Reference

A use case class responsible for removing units from a UnitRepository.
Collaboration diagram for RemoveUnitsUseCase:

Static Public Member Functions

- static Completable remove (@NonNull Context context, @NonNull UnitRepository repository, @NonNull Collection< UnitEntity > units, @NonNull SoundManager soundManager)

  Removes the given UnitEntity objects from the given UnitRepository.
- static Completable remove (@NonNull Context context, @NonNull UnitRepository repository, int id, @Non←Null SoundManager soundManager)

  Removes the unit with the given id from the given UnitRepository.

Static Private Member Functions

- static Dialog getRevengeDialog (@NonNull Context context, @NonNull UnitRepository repository, @NonNull CompletableEmitter emitter, @NonNull Collection< UnitEntity > units, @IntRange(from=1) int revengeUnits, @NonNull SoundManager soundManager)

  Creates a Dialog asking whether the Ability::REVENGE ability should be activated.

5.37.1 Detailed Description

A use case class responsible for removing units from a UnitRepository.
Capable of invoking a Dialog if a UnitEntity with the Ability::REVENGE ability is removed. Should not be used directly by the UI layer.

See also

BurnDialogUseCase

Definition at line 27 of file RemoveUnitsUseCase.java.

5.37.2 Member Function Documentation

5.37.2.1 getRevengeDialog()

```
static Dialog getRevengeDialog (
             @NonNull Context context,
             @NonNull UnitRepository repository,
             @NonNull CompletableEmitter emitter,
             @NonNull Collection< UnitEntity > units,
             @IntRange(from=1) int revengeUnits,
             @NonNull SoundManager soundManager )  [static], [private]
```

Creates a Dialog asking whether the Ability::REVENGE ability should be activated.
The Dialog is created using an RevengeAlertDialogBuilderAdapter.

Parameters

context	Context of the shown Dialog.
repository	UnitRepository where the UnitEntity objects are removed and avengers are inserted.
emitter	CompletableEmitter where CompletableEmitter::onComplete must be called, when the user makes a decision.
units	Collection of UnitEntity objects that are removed.
revengeUnits	Long representing the number of revenge units.
soundManager	SoundManager used when an Avenger is added.

Returns

A Dialog asking whether the Ability::REVENGE ability should be activated.

See also

RevengeAlertDialogBuilderAdapter::insertAvengers(UnitRepository, int, SoundManager)

Definition at line 89 of file RemoveUnitsUseCase.java.
References RevengeAlertDialogBuilderAdapter.create(), RevengeAlertDialogBuilderAdapter.insertAvengers(), and RevengeAlertDialogBuilderAdapter.setPositiveCallback().
Referenced by RemoveUnitsUseCase.remove().

5.37.2.2 remove() [1/2]

```
static Completable remove (
            @NonNull Context context,
            @NonNull UnitRepository repository,
            @NonNull Collection< UnitEntity > units,
            @NonNull SoundManager soundManager )  [static]
```

Removes the given UnitEntity objects from the given UnitRepository.
If a UnitEntity has the Ability::REVENGE ability, a Dialog asking whether the ability should be used is shown.

Parameters

context	Context of the shown Dialog.
repository	UnitRepository where the UnitEntity objects are removed.
units	Collection of UnitEntity objects that are removed.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Completable tracking operation status.

Exceptions

NullPointerException	When units contains a null value.

See also

#getRevengeDialog(Context, UnitRepository, CompletableEmitter, Collection, int, SoundManager)

UnitRepository::delete(Collection)

Definition at line 43 of file RemoveUnitsUseCase.java.
References RemoveUnitsUseCase.getRevengeDialog().

5.37.2.3 remove() [2/2]

```
static Completable remove (
            @NonNull Context context,
            @NonNull UnitRepository repository,
            int id,
            @NonNull SoundManager soundManager )  [static]
```

Removes the unit with the given id from the given UnitRepository.
If the unit has the Ability::REVENGE ability, a Dialog asking whether the ability should be used is shown. Wrapper of #remove(Context, UnitRepository, Collection).

Parameters

context	Context of the shown Dialog.
repository	UnitRepository where the UnitEntity objects are removed.
id	Integer

Returns

A Completable tracking operation status.

See also

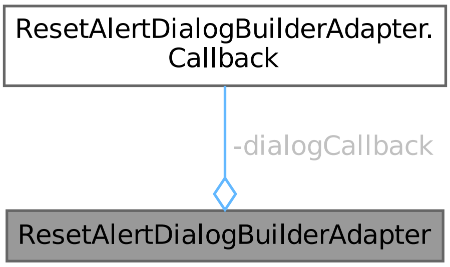
#remove(Context, UnitRepository, Collection, SoundManager)

Definition at line 68 of file RemoveUnitsUseCase.java.
The documentation for this class was generated from the following file:

- gwent/domain/cases/RemoveUnitsUseCase.java

5.38 ResetAlertDialogBuilderAdapter Class Reference

An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether he really wants to reset.
Collaboration diagram for ResetAlertDialogBuilderAdapter:


ResetAlertDialogBuilderAdapter.
Callback
-dialogCallback
ResetAlertDialogBuilderAdapter

Classes

- interface Callback

  An interface defining functions to propagate the user's decision back to the creator.

Package Functions

- ResetAlertDialogBuilderAdapter (@NonNull Context context, @NonNull Callback dialogCallback)

  Constructor of a ResetAlertDialogBuilderAdapter.
- AlertDialog create ()

  Creates an AlertDialog with the arguments supplied to this builder.
- ResetAlertDialogBuilderAdapter setTrigger (@NonNull ResetDialogUseCase.Trigger trigger)

  Changes the shown message of the AlertDialog and whether it is cancelable depending on the trigger of the dialog.
- ResetAlertDialogBuilderAdapter setMonsterDialog (boolean monsterDialog)

  Changes the positive button callback depending on whether it is a monsterDialog or not.

Private Attributes

- final AlertDialog.Builder adapteeBuilder

  AlertDialog.Builder that is adapted by this class.
- final Callback dialogCallback

  Callback used to propagate the user's answer to the built AlertDialog back to its creator.

5.38.1 Detailed Description

An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether he really wants to reset.
Definition at line 16 of file ResetAlertDialogBuilderAdapter.java.

5.38.2 Constructor & Destructor Documentation

5.38.2.1 ResetAlertDialogBuilderAdapter()

```
ResetAlertDialogBuilderAdapter (
            @NonNull Context context,
            @NonNull Callback dialogCallback )  [package]
```

Constructor of a ResetAlertDialogBuilderAdapter.
The given Callback is called when the user responds to the built AlertDialog.

Parameters

context	Context of the built AlertDialog.
dialogCallback	Callback used to propagate the user's answer to the creator.

Definition at line 36 of file ResetAlertDialogBuilderAdapter.java.
References ResetAlertDialogBuilderAdapter.dialogCallback, and ResetAlertDialogBuilderAdapter.Callback.reset().

5.38.3 Member Function Documentation

5.38.3.1 create()

```
AlertDialog create ( )  [package]
```

Creates an AlertDialog with the arguments supplied to this builder.
Basically just calls AlertDialog.Builder::create() on #adapteeBuilder.

See also

AlertDialog.Builder::create()

Definition at line 52 of file ResetAlertDialogBuilderAdapter.java.
References ResetAlertDialogBuilderAdapter.adapteeBuilder.
Referenced by ResetDialogUseCase.reset().

5.38.3.2 setMonsterDialog()

```
ResetAlertDialogBuilderAdapter setMonsterDialog (
            boolean monsterDialog )  [package]
```

Changes the positive button callback depending on whether it is a monsterDialog or not.
When it is a monsterDialog, a checkbox is shown defining whether the perk of monster faction should be activated.

Parameters

monsterDialog	Boolean defining whether a monsterDialog should be shown.

Returns

The ResetAlertDialogBuilderAdapter with the changed trigger.

Definition at line 81 of file ResetAlertDialogBuilderAdapter.java.
References ResetAlertDialogBuilderAdapter.adapteeBuilder, ResetAlertDialogBuilderAdapter.dialogCallback, and ResetAlertDialogBuilderAdapter.Callback.reset().
Referenced by ResetDialogUseCase.reset().

5.38.3.3 setTrigger()

```
ResetAlertDialogBuilderAdapter setTrigger (
            @NonNull ResetDialogUseCase.Trigger trigger )  [package]
```

Changes the shown message of the AlertDialog and whether it is cancelable depending on the trigger of the dialog.

Parameters

trigger	`ResetDialogUseCase.Trigger` defining what triggered the reset.

Returns

The ResetAlertDialogBuilderAdapter with the changed trigger.

Definition at line 64 of file ResetAlertDialogBuilderAdapter.java.
References ResetAlertDialogBuilderAdapter.adapteeBuilder, and ResetDialogUseCase.Trigger.FACTION_SWITCH.

5.38.4 Member Data Documentation

5.38.4.1 adapteeBuilder

```
final AlertDialog.Builder adapteeBuilder  [private]
```

AlertDialog.Builder that is adapted by this class.
Definition at line 21 of file ResetAlertDialogBuilderAdapter.java.
Referenced by ResetAlertDialogBuilderAdapter.create(), ResetAlertDialogBuilderAdapter.setMonsterDialog(), and ResetAlertDialogBuilderAdapter.setTrigger().

5.38.4.2 dialogCallback

```
final Callback dialogCallback  [private]
```

Callback used to propagate the user's answer to the built AlertDialog back to its creator.
Definition at line 27 of file ResetAlertDialogBuilderAdapter.java.

Referenced by ResetAlertDialogBuilderAdapter.ResetAlertDialogBuilderAdapter(), and ResetAlertDialogBuilderAdapter.setMonsterDi

The documentation for this class was generated from the following file:

- gwent/domain/cases/ResetAlertDialogBuilderAdapter.java

5.39 ResetDialogUseCase Class Reference

A use case class responsible for dispatching a reset call to ResetRepositoryUseCase, possibly after a confirmation by the user obtained from a Dialog.

Collaboration diagram for ResetDialogUseCase:

Classes

- enum DialogType

  An `Enum` defining which form of Dialog should be shown.

- enum Trigger

  An `Enum` listing the possible triggers of a reset.

Static Public Member Functions

- static Single< Boolean > reset (@NonNull Context context, @NonNull Trigger trigger, @NonNull SoundManager soundManager)

  Resets the given UnitRepository.

Static Protected Member Functions

- static Single< Boolean > reset (@NonNull Context context, @NonNull UnitRepository repository, @NonNull Trigger trigger, @NonNull SoundManager soundManager)

  Resets the given UnitRepository.

Static Private Member Functions

- static Single< DialogType > getDialogType (@NonNull Context context, @NonNull UnitRepository repository, @NonNull Trigger trigger)

  Returns a DialogType defining which kind of Dialog should be invoked.

5.39.1 Detailed Description

A use case class responsible for dispatching a reset call to ResetRepositoryUseCase, possibly after a confirmation by the user obtained from a Dialog.

See also

ResetRepositoryUseCase

Definition at line 31 of file ResetDialogUseCase.java.

5.39.2 Member Function Documentation

5.39.2.1 getDialogType()

```
static Single< DialogType > getDialogType (
             @NonNull Context context,
             @NonNull UnitRepository repository,
             @NonNull Trigger trigger )  [static], [private]
```

Returns a DialogType defining which kind of Dialog should be invoked.

Parameters

context	Context used for retrieval of SharedPreferences.
repository	UnitRepository used to check if a certain DialogType is even necessary.
trigger	`Trigger` defining what triggered the reset.

Returns

A DialogType defining the kind of Dialog.

See also

DialogType

Definition at line 102 of file ResetDialogUseCase.java.
References ResetDialogUseCase.DialogType.DEFAULT, and ResetDialogUseCase.DialogType.NONE.
Referenced by ResetDialogUseCase.reset().

5.39.2.2 reset() [1/2]

```
static Single< Boolean > reset (
             @NonNull Context context,
             @NonNull Trigger trigger,
             @NonNull SoundManager soundManager )  [static]
```

Resets the given UnitRepository.
May invoke a Dialog asking whether the user really wants to reset depending on the given trigger and warning settings. ResetRepositoryUseCase is used for resetting. Wrapper for #reset(Context, UnitRepository, Trigger).

Parameters

context	Context where a Dialog can be inflated.
trigger	`Trigger` defining what triggered this reset.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Single emitting a Boolean defining whether the reset really took place.

See also

#reset(Context, UnitRepository, Trigger, SoundManager)

ResetRepositoryUseCase::reset(Context, UnitRepository, boolean, SoundManager)

Definition at line 47 of file ResetDialogUseCase.java.
References GwentApplication.getRepository(), and ResetDialogUseCase.reset().
Referenced by ResetDialogUseCase.reset().

5.39.2.3 reset() [2/2]

```
static Single< Boolean > reset (
              @NonNull Context context,
              @NonNull UnitRepository repository,
              @NonNull Trigger trigger,
              @NonNull SoundManager soundManager )  [static], [protected]
```

Resets the given UnitRepository.

May invoke a Dialog asking whether the user really wants to reset depending on the given trigger and warning settings. ResetRepositoryUseCase is used for resetting.

Parameters

context	Context where a Dialog can be inflated.
repository	UnitRepository that is reset.
trigger	`Trigger` defining what triggered this reset.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Single emitting a Boolean defining whether the reset really took place.

See also

ResetRepositoryUseCase::reset(Context, UnitRepository, boolean, SoundManager)

Definition at line 66 of file ResetDialogUseCase.java.

References ResetAlertDialogBuilderAdapter.create(), ResetDialogUseCase.getDialogType(), ResetDialogUseCase.DialogType.MONS
ResetDialogUseCase.DialogType.NONE, ResetRepositoryUseCase.reset(), and ResetAlertDialogBuilderAdapter.setMonsterDialog().

The documentation for this class was generated from the following file:

- gwent/domain/cases/ResetDialogUseCase.java

5.40 ResetRepositoryUseCase Class Reference

A use case class responsible for resetting the UnitRepository.

Collaboration diagram for ResetRepositoryUseCase:

Static Public Member Functions

- static Completable reset (@NonNull Context context, @NonNull UnitRepository repository, @NonNull SoundManager soundManager)

  Resets the given UnitRepository.

Static Package Functions

- static Maybe< UnitEntity > reset (@NonNull Context context, @NonNull UnitRepository repository, boolean keepUnit, @NonNull SoundManager soundManager)

  Resets the given UnitRepository and keeps one random unit if keepUnit is true.

Static Private Member Functions

- static Optional< UnitEntity > getRandomUnit (@NonNull List< UnitEntity > units)

  Selects a random unit that is not epic.
- static Dialog getRevengeDialog (@NonNull Context context, @NonNull UnitRepository repository, @NonNull CompletableEmitter emitter, @Nullable UnitEntity keptUnit, @IntRange(from=1) int revengeUnits, @NonNull SoundManager soundManager)

  Creates a Dialog asking whether the Ability::REVENGE ability should be activated.

5.40.1 Detailed Description

A use case class responsible for resetting the UnitRepository.
Capable of invoking a Dialog if a UnitEntity with the Ability::REVENGE ability is removed. Should not be used directly by the UI layer.

See also

ResetDialogUseCase

Definition at line 31 of file ResetRepositoryUseCase.java.

5.40.2 Member Function Documentation

5.40.2.1 getRandomUnit()

```
static Optional< UnitEntity > getRandomUnit (
             @NonNull List< UnitEntity > units )  [static], [private]
```

Selects a random unit that is not epic.
If all units are epic or if there are no units at all, an empty Optional is returned.

Parameters

units	List of UnitEntity objects, one of which is selected.

Returns

An Optional containing the selected unit or nothing if no unit could be selected.

Definition at line 71 of file ResetRepositoryUseCase.java.
Referenced by ResetRepositoryUseCase.reset().

5.40.2.2 getRevengeDialog()

```
static Dialog getRevengeDialog (
             @NonNull Context context,
             @NonNull UnitRepository repository,
             @NonNull CompletableEmitter emitter,
             @Nullable UnitEntity keptUnit,
             @IntRange(from=1) int revengeUnits,
             @NonNull SoundManager soundManager )  [static], [private]
```

Creates a Dialog asking whether the Ability::REVENGE ability should be activated.
The Dialog is created using an RevengeAlertDialogBuilderAdapter.

Parameters

context	Context of the shown Dialog.
repository	UnitRepository where the UnitEntity objects are removed and avengers are inserted.
emitter	CompletableEmitter where CompletableEmitter::onComplete must be called, when the user makes a decision.
keptUnit	UnitEntity that should be kept.

Parameters

revengeUnits	Long representing the number of revenge units.
soundManager	SoundManager used, if an Avenger is added.

Returns

A Dialog asking whether the Ability::REVENGE ability should be activated.

See also

RevengeAlertDialogBuilderAdapter::insertAvengers(UnitRepository, int, SoundManager)

Definition at line 93 of file ResetRepositoryUseCase.java.
References RevengeAlertDialogBuilderAdapter.create(), RevengeAlertDialogBuilderAdapter.insertAvengers(), and RevengeAlertDialogBuilderAdapter.setPositiveCallback().
Referenced by ResetRepositoryUseCase.reset().

5.40.2.3 reset() [1/2]

```
static Completable reset (
            @NonNull Context context,
            @NonNull UnitRepository repository,
            @NonNull SoundManager soundManager )  [static]
```

Resets the given UnitRepository.
If a removed UnitEntity has the Ability::REVENGE ability, a Dialog asking whether the ability should be used is shown. Wrapper of #reset(Context, UnitRepository, boolean, SoundManager).

Parameters

context	Context of the shown Dialog.
repository	UnitRepository that is being reset.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Completable tracking operation status.

See also

#reset(Context, UnitRepository, boolean, SoundManager)

Definition at line 123 of file ResetRepositoryUseCase.java.
References ResetRepositoryUseCase.reset().

5.40.2.4 reset() [2/2]

```
static Maybe< UnitEntity > reset (
            @NonNull Context context,
            @NonNull UnitRepository repository,
            boolean keepUnit,
            @NonNull SoundManager soundManager )  [static], [package]
```

Resets the given UnitRepository and keeps one random unit if keepUnit is true.
If a removed UnitEntity has the Ability::REVENGE ability, a Dialog asking whether the ability should be used is shown.

Parameters

context	Context of the shown Dialog.

Parameters

repository	UnitRepository that is being reset.
keepUnit	Boolean defining whether a single UnitEntity should be kept.
soundManager	SoundManager used, if an UnitEntity has the Ability::REVENGE ability.

Returns

A Maybe emitting the kept UnitEntity or nothing if keepUnit is false.

See also

#getRevengeDialog(Context, UnitRepository, CompletableEmitter, UnitEntity, int, SoundManager)

UnitRepository::reset(UnitEntity)

Definition at line 46 of file ResetRepositoryUseCase.java.
References ResetRepositoryUseCase.getRandomUnit(), and ResetRepositoryUseCase.getRevengeDialog().
Referenced by ResetRepositoryUseCase.reset(), and ResetDialogUseCase.reset().
The documentation for this class was generated from the following file:

- gwent/domain/cases/ResetRepositoryUseCase.java

5.41 RevengeAlertDialogBuilderAdapter Class Reference

An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether they want to invoke the Ability::REVENGE ability.
Collaboration diagram for RevengeAlertDialogBuilderAdapter:

Static Public Member Functions

- static Completable insertAvengers (@NonNull UnitRepository repository, @IntRange(from=0) int numberOfAvengers, @NonNull SoundManager soundManager)

  Inserts numberOfAvengers avenger units into the given UnitRepository.

Package Functions

- RevengeAlertDialogBuilderAdapter (@NonNull Context context)

  Constructor of a RevengeAlertDialogBuilderAdapter.
- AlertDialog create ()

  Creates an AlertDialog with the arguments supplied to this builder.
- RevengeAlertDialogBuilderAdapter setPositiveCallback (@NonNull DialogInterface.OnClickListener onPositiveButtonClick)

  Adds the given callback to the positive button of the built Dialog.
- RevengeAlertDialogBuilderAdapter setNegativeCallback (@NonNull DialogInterface.OnClickListener onNegativeButtonClick)

  Adds the given callback to the negative button of the built Dialog.

Private Attributes

- final AlertDialog.Builder adapteeBuilder

  AlertDialog.Builder that is adapted by this class.

Static Private Attributes

- static final boolean AVENGER_EPIC = false

  Boolean constant defining whether the default UnitEntity summoned by the Ability::REVENGE ability is epic.
- static final Ability AVENGER_ABILITY = Ability.NONE

  Ability constant defining the Ability of the default UnitEntity summoned by the Ability::REVENGE ability.
- static final int AVENGER_DAMAGE = 8

  Integer constant defining the damage of the default UnitEntity summoned by the Ability::REVENGE ability.
- static final Integer AVENGER_SQUAD = null

  Integer constant defining the squad of the default UnitEntity summoned by the Ability::REVENGE ability.
- static final RowType AVENGER_ROW = RowType.MELEE

  RowType constant defining the row of the default UnitEntity summoned by the Ability::REVENGE ability.

5.41.1 Detailed Description

An adapter class adapting AlertDialog.Builder to provide an interface for creating an AlertDialog asking the user whether they want to invoke the Ability::REVENGE ability.
Definition at line 23 of file RevengeAlertDialogBuilderAdapter.java.

5.41.2 Constructor & Destructor Documentation

5.41.2.1 RevengeAlertDialogBuilderAdapter()

```
RevengeAlertDialogBuilderAdapter (
            @NonNull Context context )  [package]
```

Constructor of a RevengeAlertDialogBuilderAdapter.
Initializes the buttons with empty callbacks.

Parameters

context	Context of the built AlertDialog.

Definition at line 76 of file RevengeAlertDialogBuilderAdapter.java.

5.41.3 Member Function Documentation

5.41.3.1 create()

```
AlertDialog create ( )  [package]
```

Creates an AlertDialog with the arguments supplied to this builder.
Basically just calls AlertDialog.Builder::create() on #adapteeBuilder.

See also

AlertDialog.Builder::create()

Definition at line 108 of file RevengeAlertDialogBuilderAdapter.java.
References RevengeAlertDialogBuilderAdapter.adapteeBuilder.
Referenced by RemoveUnitsUseCase.getRevengeDialog(), and ResetRepositoryUseCase.getRevengeDialog().

5.41.3.2 insertAvengers()

```
static Completable insertAvengers (
            @NonNull UnitRepository repository,
            @IntRange(from=0) int numberOfAvengers,
            @NonNull SoundManager soundManager )  [static]
```

Inserts numberOfAvengers avenger units into the given UnitRepository.
The inserted UnitEntity objects have the attributes defined in #AVENGER_EPIC, #AVENGER_DAMAGE, #↩AVENGER_ABILITY, #AVENGER_SQUAD, and #AVENGER_ROW.

Parameters

repository	UnitRepository where avengers are inserted.
numberOfAvengers	Integer defining how many avengers are inserted.

Returns

A Completable tracking operation status.

Definition at line 96 of file RevengeAlertDialogBuilderAdapter.java.
References RevengeAlertDialogBuilderAdapter.AVENGER_ABILITY, RevengeAlertDialogBuilderAdapter.AVENGER_DAMAGE, RevengeAlertDialogBuilderAdapter.AVENGER_EPIC, RevengeAlertDialogBuilderAdapter.AVENGER_ROW, and RevengeAlertDialogBuilderAdapter.AVENGER_SQUAD.
Referenced by RemoveUnitsUseCase.getRevengeDialog(), and ResetRepositoryUseCase.getRevengeDialog().

5.41.3.3 setNegativeCallback()

```
RevengeAlertDialogBuilderAdapter setNegativeCallback (
            @NonNull DialogInterface.OnClickListener onNegativeButtonClick )  [package]
```

Adds the given callback to the negative button of the built Dialog.

Parameters

onNegativeButtonClick	DialogInterface.OnClickListener that is called, when the negative button is clicked.

Returns

The RevengeAlertDialogBuilder with the updated negative callback.

Definition at line 133 of file RevengeAlertDialogBuilderAdapter.java.
References RevengeAlertDialogBuilderAdapter.adapteeBuilder.

5.41.3.4 setPositiveCallback()

```
RevengeAlertDialogBuilderAdapter setPositiveCallback (
            @NonNull DialogInterface.OnClickListener onPositiveButtonClick )  [package]
```

Adds the given callback to the positive button of the built Dialog.
Callback should call insertAvengers().

Parameters

onPositiveButtonClick	DialogInterface.OnClickListener that is called, when the positive button is clicked.

Returns

The RevengeAlertDialogBuilder with the updated positive callback.

See also

#insertAvengers(UnitRepository, int, SoundManager)

Definition at line 121 of file RevengeAlertDialogBuilderAdapter.java.
References RevengeAlertDialogBuilderAdapter.adapteeBuilder.
Referenced by RemoveUnitsUseCase.getRevengeDialog(), and ResetRepositoryUseCase.getRevengeDialog().

5.41.4 Member Data Documentation

5.41.4.1 adapteeBuilder

`final AlertDialog.Builder adapteeBuilder [private]`

AlertDialog.Builder that is adapted by this class.
Definition at line 68 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.create(), RevengeAlertDialogBuilderAdapter.setNegativeCallback(), and RevengeAlertDialogBuilderAdapter.setPositiveCallback().

5.41.4.2 AVENGER_ABILITY

`final Ability AVENGER_ABILITY = Ability.NONE [static], [private]`

Ability constant defining the Ability of the default UnitEntity summoned by the Ability::REVENGE ability.

See also

Ability::REVENGE

Definition at line 38 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.insertAvengers().

5.41.4.3 AVENGER_DAMAGE

`final int AVENGER_DAMAGE = 8 [static], [private]`

Integer constant defining the damage of the default UnitEntity summoned by the Ability::REVENGE ability.

See also

Ability::REVENGE

Definition at line 46 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.insertAvengers().

5.41.4.4 AVENGER_EPIC

`final boolean AVENGER_EPIC = false [static], [private]`

Boolean constant defining whether the default UnitEntity summoned by the Ability::REVENGE ability is epic.

See also

Ability::REVENGE

Definition at line 30 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.insertAvengers().

5.41.4.5 AVENGER_ROW

`final RowType AVENGER_ROW = RowType.MELEE [static], [private]`

RowType constant defining the row of the default UnitEntity summoned by the Ability::REVENGE ability.

See also

Ability::REVENGE

Definition at line 62 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.insertAvengers().

5.41.4.6 AVENGER_SQUAD

`final Integer AVENGER_SQUAD = null [static], [private]`

Integer constant defining the squad of the default UnitEntity summoned by the Ability::REVENGE ability.

See also

Ability::REVENGE

Definition at line 54 of file RevengeAlertDialogBuilderAdapter.java.
Referenced by RevengeAlertDialogBuilderAdapter.insertAvengers().
The documentation for this class was generated from the following file:

- gwent/domain/cases/RevengeAlertDialogBuilderAdapter.java

5.42 RowDao Interface Reference

An interface defining data access operations related to `rows` table.
Collaboration diagram for RowDao:

Public Member Functions

- Completable insertRow (@NonNull RowEntity row)

  Inserts the given RowEntity into `rows`.
- Completable clearRows ()

  Deletes all RowEntity objects from `rows`.
- Completable updateWeather (@NonNull RowType row)

  Flips RowEntity::weather for the given RowType in `rows`.
- Completable clearWeather ()

  Sets RowEntity::weather to `false` for all RowEntity elements in `rows`.
- Completable updateHorn (@NonNull RowType row)

  Flips RowEntity::horn for the given RowType in `rows`.
- Single< Boolean > isWeather (@NonNull RowType row)

  Fetches RowEntity::weather for the given RowType in `rows`.
- Flowable< Boolean > isWeatherFlowable (@NonNull RowType row)

  Fetches a Flowable of RowEntity::weather for the given RowType in `rows`.
- Single< Boolean > isHorn (@NonNull RowType row)

  Fetches RowEntity::horn for the given RowType in `rows`.
- Flowable< Boolean > isHornFlowable (@NonNull RowType row)

  Fetches a Flowable of RowEntity::horn for the given RowType in `rows`.

5.42.1 Detailed Description

An interface defining data access operations related to `rows` table.
The implementation is generated by the Room framework.
Definition at line 19 of file RowDao.java.

5.42.2 Member Function Documentation

5.42.2.1 clearRows()

`Completable clearRows ( )`

Deletes all RowEntity objects from `rows`.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.reset().

5.42.2.2 clearWeather()

`Completable clearWeather ( )`

Sets RowEntity::weather to `false` for all RowEntity elements in `rows`.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.clearWeather().

5.42.2.3 insertRow()

```
Completable insertRow (
            @NonNull RowEntity row )
```

Inserts the given RowEntity into `rows`.
If the same RowType is already in the table, the new insert is ignored.

Parameters

row	RowEntity that is being inserted.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.initializeRows().

5.42.2.4 isHorn()

```
Single< Boolean > isHorn (
            @NonNull RowType row )
```

Fetches RowEntity::horn for the given RowType in `rows`.

See also

#isHornFlowable(RowType)

Parameters

row	RowType of the queried RowEntity.

Returns

A Single tracking operation status and returning the value.

Referenced by UnitRepository.isHorn().

5.42.2.5 isHornFlowable()

```
Flowable< Boolean > isHornFlowable (
            @NonNull RowType row )
```

Fetches a Flowable of RowEntity::horn for the given RowType in `rows`.

Parameters

row	RowType of the queried RowEntity.

Returns

A Flowable emitting the values.

See also

#isHorn(RowType)

Referenced by UnitRepository.isHornFlowable().

5.42.2.6 isWeather()

```
Single< Boolean > isWeather (
            @NonNull RowType row )
```

Fetches RowEntity::weather for the given RowType in `rows`.

See also

#isWeatherFlowable(RowType)

Parameters

row	RowType of the queried RowEntity.

Returns

A Single tracking operation status and returning the value.

Referenced by UnitRepository.isWeather().

5.42.2.7 isWeatherFlowable()

```
Flowable< Boolean > isWeatherFlowable (
            @NonNull RowType row )
```

Fetches a Flowable of RowEntity::weather for the given RowType in `rows`.

Parameters

row	RowType of the queried RowEntity.

Returns

A Flowable emitting the values.

See also

#isWeather(RowType)

Referenced by UnitRepository.isWeatherFlowable().

5.42.2.8 updateHorn()

```
Completable updateHorn (
            @NonNull RowType row )
```

Flips RowEntity::horn for the given RowType in `rows`.

Parameters

row	RowType of the updated RowEntity.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.switchHorn().

5.42.2.9 updateWeather()

```
Completable updateWeather (
            @NonNull RowType row )
```

Flips RowEntity::weather for the given RowType in `rows`.

Parameters

row	RowType of the updated RowEntity.

Returns

A Completable tracking operation status.

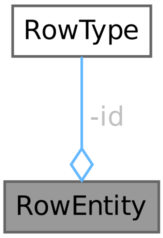
Referenced by UnitRepository.switchWeather().
The documentation for this interface was generated from the following file:

- gwent/data/RowDao.java

5.43 RowEntity Class Reference

A class representing the state of an attack row, i.e., weather and commanders horn.

Collaboration diagram for RowEntity:



Package Functions

- RowEntity (@NonNull RowType id)

  Constructor of a RowEntity.
- RowType getId ()

  Getter for #id.
- boolean isWeather ()

  Getter for #weather.
- void setWeather (boolean weather)

  Setter for #weather.
- boolean isHorn ()

  Getter for #horn.
- void setHorn (boolean horn)

  Setter for #horn.

Private Attributes

- final RowType id

  Defines the type of the row as one of the values in RowType, thereby limiting the number of different rows to three.
- boolean weather

  Defines whether the weather debuff is present in the represented row, i.e., if set to true, the damage value of units in the represented row is set to 1.
- boolean horn

  Defines whether the commanders horn buff is present in the represented row, i.e., if set to true, the damage value of units in the represented row is doubled.

5.43.1 Detailed Description

A class representing the state of an attack row, i.e., weather and commanders horn.
Is a persistent Entity and is therefore saved in a database table named `rows`.
Definition at line 14 of file RowEntity.java.

5.43.2 Constructor & Destructor Documentation

5.43.2.1 RowEntity()

```
RowEntity (
              @NonNull RowType id )   [package]
```

Constructor of a RowEntity.

#weather and #horn cannot be set here since a new row must start with the respective default values.

Parameters

id	RowType of the represented row.

Definition at line 47 of file RowEntity.java.

5.43.3 Member Function Documentation

5.43.3.1 getId()

```
RowType getId ( )  [package]
```

Getter for #id.
Only used by Room extension.

Returns

RowType of the represented row.

Definition at line 58 of file RowEntity.java.

5.43.3.2 isHorn()

```
boolean isHorn ( )  [package]
```

Getter for #horn.
Only used by Room extension.

Returns

Boolean representing the current status of the horn buff.

Definition at line 88 of file RowEntity.java.

5.43.3.3 isWeather()

```
boolean isWeather ( )  [package]
```

Getter for #weather.
Only used by Room extension.

Returns

Boolean representing the current status of the weather debuff.

Definition at line 68 of file RowEntity.java.

5.43.3.4 setHorn()

```
void setHorn (
             boolean horn )  [package]
```

Setter for #horn.
Only used by Room extension.

Parameters

horn	Boolean representing the new status of the horn buff.

Definition at line 98 of file RowEntity.java.

5.43.3.5 setWeather()

```
void setWeather (
             boolean weather )  [package]
```

Setter for #weather.

Only used by Room extension.

Parameters

weather	Boolean representing the new status of the weather debuff.

Definition at line 78 of file RowEntity.java.

5.43.4 Member Data Documentation

5.43.4.1 horn

`boolean horn [private]`

Defines whether the commanders horn buff is present in the represented row, i.e., if set to true, the damage value of units in the represented row is doubled.
Is set to `false` by default.
Definition at line 38 of file RowEntity.java.

5.43.4.2 id

`final RowType id [private]`

Defines the type of the row as one of the values in RowType, thereby limiting the number of different rows to three.
Primary key of the represented row.
Definition at line 22 of file RowEntity.java.

5.43.4.3 weather

`boolean weather [private]`

Defines whether the weather debuff is present in the represented row, i.e., if set to true, the damage value of units in the represented row is set to 1.
Is set to `false` by default.
Definition at line 30 of file RowEntity.java.
The documentation for this class was generated from the following file:

- gwent/data/RowEntity.java

5.44 RowType Enum Reference

An enum listing the different types of attack rows.
Collaboration diagram for RowType:

Public Attributes

- MELEE

  Represents the row containing melee combat units.
- RANGE

  Represents the row containing ranged combat units.
- SIEGE

Represents the row containing siege combat units.

5.44.1 Detailed Description

An enum listing the different types of attack rows.
Definition at line 6 of file RowType.java.

5.44.2 Member Data Documentation

5.44.2.1 MELEE

`MELEE`
Represents the row containing melee combat units.
Definition at line 10 of file RowType.java.

5.44.2.2 RANGE

`RANGE`
Represents the row containing ranged combat units.
Definition at line 15 of file RowType.java.

5.44.2.3 SIEGE

`SIEGE`
Represents the row containing siege combat units.
Definition at line 21 of file RowType.java.
The documentation for this enum was generated from the following file:

- gwent/data/RowType.java

5.45 RowUiState Class Reference

A data class encapsulating the visible state of a row in the main view of the application.
Collaboration diagram for RowUiState:

Public Member Functions

- RowUiState (@IntRange(from=0) int damage, boolean weather, boolean horn, @IntRange(from=0) int units)

  Constructor of a RowUiState encapsulating the given data.
- int getDamage ()

  Returns the summed-up damage of all units in this row.
- boolean isWeather ()

  Returns whether the weather debuff is active in this row.
- boolean isHorn ()

  Returns whether the commander's horn buff is active in this row.
- int getUnits ()

  Returns the number of units in this row.
- boolean equals (Object o)

Private Attributes

- final int damage

  Defines the summed-up damage of all units in this row.
- final boolean weather

  Defines whether the weather debuff is active in this row.
- final boolean horn

  Defines whether the commander's horn buff is active in this row.
- final int units

  Defines the number of units in this row.

5.45.1 Detailed Description

A data class encapsulating the visible state of a row in the main view of the application.
Definition at line 10 of file RowUiState.java.

5.45.2 Constructor & Destructor Documentation

5.45.2.1 RowUiState()

```
RowUiState (
            @IntRange(from=0) int damage,
            boolean weather,
            boolean horn,
            @IntRange(from=0) int units )
```

Constructor of a RowUiState encapsulating the given data.

Parameters

damage	Integer representing the summed-up damage of all units.
weather	Boolean defining whether the weather debuff is active.
horn	Boolean defining whether the commander's horn buff is active.
units	Integer representing the number of units.

Exceptions

IllegalArgumentException	When damage or units is negative.

Definition at line 41 of file RowUiState.java.
References RowUiState.damage, RowUiState.horn, RowUiState.units, and RowUiState.weather.

5.45.3 Member Function Documentation

5.45.3.1 equals()

```
boolean equals (
            Object o )
```

Definition at line 89 of file RowUiState.java.

5.45.3.2 getDamage()

```
int getDamage ( )
```

Returns the summed-up damage of all units in this row.

Returns

An Integer representing the summed-up damage of all units.

Definition at line 59 of file RowUiState.java.
References RowUiState.damage.
Referenced by GameBoardViewModel.getModel().

5.45.3.3 getUnits()

```
int getUnits ( )
```

Returns the number of units in this row.

Returns

An Integer representing the number of units in this row.

Definition at line 84 of file RowUiState.java.
References RowUiState.units.
Referenced by GameBoardViewModel.getModel().

5.45.3.4 isHorn()

```
boolean isHorn ( )
```

Returns whether the commander's horn buff is active in this row.

Returns

A Boolean whether the commander's horn buff is active.

Definition at line 75 of file RowUiState.java.
References RowUiState.horn.
Referenced by GameBoardViewModel.getModel().

5.45.3.5 isWeather()

```
boolean isWeather ( )
```

Returns whether the weather debuff is active in this row.

Returns

A Boolean defining whether the weather debuff is active.

Definition at line 67 of file RowUiState.java.
References RowUiState.weather.
Referenced by GameBoardViewModel.getModel().

5.45.4 Member Data Documentation

5.45.4.1 damage

```
final int damage  [private]
```

Defines the summed-up damage of all units in this row.
Definition at line 15 of file RowUiState.java.
Referenced by RowUiState.RowUiState(), and RowUiState.getDamage().

5.45.4.2 horn

```
final boolean horn  [private]
```

Defines whether the commander's horn buff is active in this row.
Definition at line 25 of file RowUiState.java.
Referenced by RowUiState.RowUiState(), and RowUiState.isHorn().

5.45.4.3 units

```
final int units [private]
```

Defines the number of units in this row.

Definition at line 31 of file RowUiState.java.

Referenced by RowUiState.RowUiState(), and RowUiState.getUnits().

5.45.4.4 weather

```
final boolean weather [private]
```

Defines whether the weather debuff is active in this row.

Definition at line 20 of file RowUiState.java.

Referenced by RowUiState.RowUiState(), and RowUiState.isWeather().

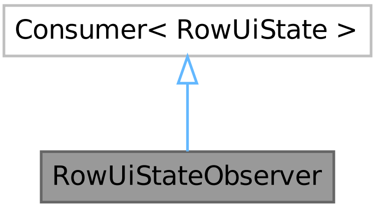
The documentation for this class was generated from the following file:

- gwent/ui/main/RowUiState.java

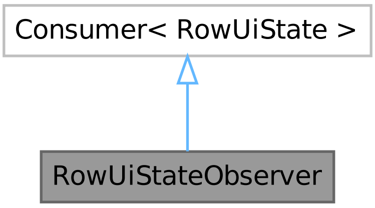
5.46 RowUiStateObserver Class Reference

An observer class responsible for updating the views of the row defined in #row when notified, i.e., when a new RowUiState is produced the observed Flowable.

Inheritance diagram for RowUiStateObserver:



Collaboration diagram for RowUiStateObserver:



Public Member Functions

- void accept (@NonNull RowUiState rowUiState)

    Changes values in #damageView and #unitView as well as the image in #weatherView and #hornView.

Static Public Member Functions

- static RowUiStateObserver getObserver (@NonNull RowType row, @NonNull TextView damageView, @NonNull ImageView weatherView, @NonNull ImageView hornView, @NonNull TextView unitView)

  Returns a new RowUiStateObserver for the given row updating the given views.

Private Member Functions

- RowUiStateObserver (@NonNull TextView damageView, @NonNull ImageView weatherView, @Drawable←Res int weatherResource, @NonNull ImageView hornView, @NonNull TextView unitView)

  Constructor of a RowUiStateObserver updating the given views when #onChanged() is called.

Private Attributes

- final TextView damageView

  A TextView showing the user the summed-up damage of all units in this row, i.e., RowUiState::damage.
- final ImageView weatherView

  An ImageView responsible for showing the current state of the weather debuff in this row.
- final int weatherResource

  An Integer representing the drawable resource shown by #weatherView when the weather debuff is active, i.e., when RowUiState::weather is `true`.
- final ImageView hornView

  An ImageView responsible for showing the current state of the commander's horn buff in this row.
- final TextView unitView

  A TextView showing the number of units in this row, i.e., RowUiState::units.

5.46.1 Detailed Description

An observer class responsible for updating the views of the row defined in #row when notified, i.e., when a new RowUiState is produced the observed Flowable.

See also

RowUiState

Definition at line 22 of file RowUiStateObserver.java.

5.46.2 Constructor & Destructor Documentation

5.46.2.1 RowUiStateObserver()

```
RowUiStateObserver (
              @NonNull TextView damageView,
              @NonNull ImageView weatherView,
              @DrawableRes int weatherResource,
              @NonNull ImageView hornView,
              @NonNull TextView unitView )  [private]
```

Constructor of a RowUiStateObserver updating the given views when #onChanged() is called.
Should only be called by #getObserver().

Parameters

damageView	TextView showing the summed-up damage of all units in the observed row.
weatherView	ImageView showing the current state of the weather debuff of the observed row.
weatherResource	Integer representing the drawable resource shown when the weather debuff is active.
hornView	ImageView showing the current state of the commander's horn buff of the observed row.
unitView	TextView showing the number of units in the observed row.

See also

#getObserver(RowType, TextView, ImageView, ImageView, TextView)

Definition at line 74 of file RowUiStateObserver.java.
References RowUiStateObserver.damageView, RowUiStateObserver.hornView, RowUiStateObserver.unitView, RowUiStateObserver.weatherResource, and RowUiStateObserver.weatherView.

5.46.3 Member Function Documentation

5.46.3.1 accept()

```
void accept (
            @NonNull RowUiState rowUiState )
```

Changes values in #damageView and #unitView as well as the image in #weatherView and #hornView.
Image switches are animated using ImageViewSwitchAnimator.

Parameters

rowUiState	RowUiState representing the updated state of the row.

See also

ImageViewSwitchAnimator

Definition at line 127 of file RowUiStateObserver.java.
References ImageViewSwitchAnimator.animatedSwitch(), RowUiStateObserver.damageView, RowUiStateObserver.hornView, RowUiStateObserver.unitView, RowUiStateObserver.weatherResource, and RowUiStateObserver.weatherView.

5.46.3.2 getObserver()

```
static RowUiStateObserver getObserver (
            @NonNull RowType row,
            @NonNull TextView damageView,
            @NonNull ImageView weatherView,
            @NonNull ImageView hornView,
            @NonNull TextView unitView )  [static]
```

Returns a new RowUiStateObserver for the given row updating the given views.
Factory method for RowUiStateObserver. #weatherResource is defined according to the given RowType.

Parameters

row	RowType defining which row is being observed.
damageView	TextView showing the summed-up damage of all units in the observed row.
weatherView	ImageView showing the current state of the weather debuff of the observed row.
hornView	ImageView showing the current state of the commander's horn buff of the observed row.
unitView	TextView showing the number of units in the observed row.

Returns

A RowUiStateObserver updating the given views.

See also

#RowUiStateObserver(TextView, ImageView, int, ImageView, TextView)

Definition at line 99 of file RowUiStateObserver.java.
References RowUiStateObserver.damageView, RowUiStateObserver.hornView, RowUiStateObserver.unitView, RowUiStateObserver.weatherResource, and RowUiStateObserver.weatherView.
Referenced by MainActivity.initializeViewModel().

5.46.4 Member Data Documentation

5.46.4.1 damageView

`final TextView damageView [private]`

A TextView showing the user the summed-up damage of all units in this row, i.e., RowUiState::damage.
Definition at line 28 of file RowUiStateObserver.java.
Referenced by RowUiStateObserver.RowUiStateObserver(), RowUiStateObserver.accept(), and RowUiStateObserver.getObserver().

5.46.4.2 hornView

`final ImageView hornView [private]`

An ImageView responsible for showing the current state of the commander's horn buff in this row.
Is gray when RowUiState::horn is `false`.

See also

#weatherResource

Definition at line 55 of file RowUiStateObserver.java.
Referenced by RowUiStateObserver.RowUiStateObserver(), RowUiStateObserver.accept(), and RowUiStateObserver.getObserver().

5.46.4.3 unitView

`final TextView unitView [private]`

A TextView showing the number of units in this row, i.e., RowUiState::units.
Definition at line 61 of file RowUiStateObserver.java.
Referenced by RowUiStateObserver.RowUiStateObserver(), RowUiStateObserver.accept(), and RowUiStateObserver.getObserver().

5.46.4.4 weatherResource

`final int weatherResource [private]`

An Integer representing the drawable resource shown by #weatherView when the weather debuff is active, i.e., when RowUiState::weather is `true`.

See also

#weatherView

Definition at line 46 of file RowUiStateObserver.java.
Referenced by RowUiStateObserver.RowUiStateObserver(), RowUiStateObserver.accept(), and RowUiStateObserver.getObserver().

5.46.4.5 weatherView

`final ImageView weatherView [private]`

An ImageView responsible for showing the current state of the weather debuff in this row.
Displays the resource in #weatherResource when RowUiState::weather is `true`.

See also

#weatherResource

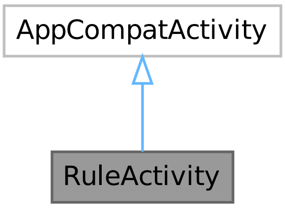
Definition at line 37 of file RowUiStateObserver.java.
Referenced by RowUiStateObserver.RowUiStateObserver(), RowUiStateObserver.accept(), and RowUiStateObserver.getObserver().
The documentation for this class was generated from the following file:

- gwent/ui/main/RowUiStateObserver.java

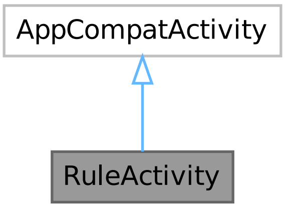
5.47 RuleActivity Class Reference

An `AppCompatActivity` used to present the rules of the game.

Inheritance diagram for RuleActivity:



Collaboration diagram for RuleActivity:



Public Member Functions

- void onCreate (Bundle savedInstanceState)

  Initializes the content of the selected rule.
- boolean onOptionsItemSelected (@NonNull MenuItem item)

  Called whenever a `MenuItem` in the options menu is selected.

Static Public Attributes

- static final String INTENT_EXTRA_TAG = "rule_section"

  `String` constant defining the identifier where the requested `RuleSection` is provided in the calling `android.content.Intent`.

5.47.1 Detailed Description

An `AppCompatActivity` used to present the rules of the game.
Can be accessed from the `SettingsActivity`. When called, a selected `RuleSection` must always be given through the calling `android.content.Intent` at the key defined in `INTENT_EXTRA_TAG`.
Definition at line 22 of file RuleActivity.java.

5.47.2 Member Function Documentation

5.47.2.1 onCreate()

```
void onCreate (
              Bundle savedInstanceState )
```

Initializes the content of the selected rule.

Parameters

savedInstanceState	If the activity is being re-initialized after previously being shut down then this Bundle contains the data it most recently supplied in `onSaveInstanceState`. ***Note: Otherwise it is null.***

Definition at line 37 of file RuleActivity.java.
References RuleActivity.INTENT_EXTRA_TAG.

5.47.2.2 onOptionsItemSelected()

```
boolean onOptionsItemSelected (
              @NonNull MenuItem item )
```

Called whenever a `MenuItem` in the options menu is selected.
Returns to the calling `android.app.Activity` when the `android.R.id#home` item was selected.

Parameters

item	`MenuItem` that was selected.

Returns

`Boolean` defining whether the call has been handled.

Definition at line 97 of file RuleActivity.java.

5.47.3 Member Data Documentation

5.47.3.1 INTENT_EXTRA_TAG

```
final String INTENT_EXTRA_TAG = "rule_section"  [static]
```

`String` constant defining the identifier where the requested `RuleSection` is provided in the calling `android.content.Intent`.
Definition at line 27 of file RuleActivity.java.
Referenced by SettingsRuleFragment.getSectionClickListener(), and RuleActivity.onCreate().
The documentation for this class was generated from the following file:

- gwent/ui/settings/RuleActivity.java

5.48 RuleSection Enum Reference

An `Enum` used to discern the different possible sections shown by `RuleActivity`.
Collaboration diagram for RuleSection:

Public Attributes

- GENERAL

  Represents a section containing general background-information on the game Gwent.
- COURSE

  Represents a section containing information on the course of a game of Gwent.
- FACTIONS

  Represents a section containing information on the factions playable in Gwent.
- COMMANDER

  Represents a section containing information on the available commanders of the factions and their abilities.
- CARDS

  Represents a section containing information on the different types of cards in Gwent.
- CARD_ABILITIES

  Represents a section containing information on the possible abilities of cards in Gwent.
- SPECIAL_CARDS

  Represents a section containing information on special cards available in Gwent.

5.48.1 Detailed Description

An `Enum` used to discern the different possible sections shown by `RuleActivity`.
Definition at line 6 of file RuleSection.java.

5.48.2 Member Data Documentation

5.48.2.1 CARD_ABILITIES

`CARD_ABILITIES`
Represents a section containing information on the possible abilities of cards in Gwent.
Definition at line 35 of file RuleSection.java.

5.48.2.2 CARDS

`CARDS`
Represents a section containing information on the different types of cards in Gwent.
Definition at line 30 of file RuleSection.java.

5.48.2.3 COMMANDER

`COMMANDER`
Represents a section containing information on the available commanders of the factions and their abilities.
Definition at line 25 of file RuleSection.java.

5.48.2.4 COURSE

`COURSE`
Represents a section containing information on the course of a game of Gwent.
Definition at line 15 of file RuleSection.java.

5.48.2.5 FACTIONS

`FACTIONS`
Represents a section containing information on the factions playable in Gwent.
Definition at line 20 of file RuleSection.java.

5.48.2.6 GENERAL

`GENERAL`
Represents a section containing general background-information on the game Gwent.
Definition at line 10 of file RuleSection.java.

5.48.2.7 SPECIAL_CARDS

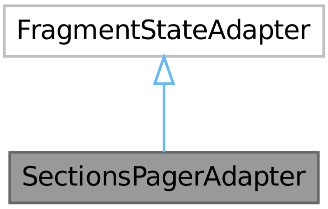
`SPECIAL_CARDS`

Represents a section containing information on special cards available in Gwent.
Definition at line 41 of file RuleSection.java.
The documentation for this enum was generated from the following file:

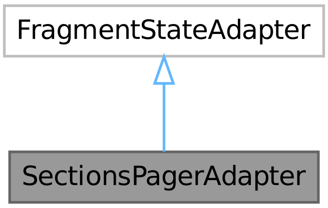
- gwent/ui/settings/RuleSection.java

5.49 SectionsPagerAdapter Class Reference

A `FragmentStateAdapter` that returns an `IntroductionFragment` corresponding to one of the sections/tabs/pages.
Inheritance diagram for SectionsPagerAdapter:



Collaboration diagram for SectionsPagerAdapter:



Public Member Functions

- SectionsPagerAdapter (@NonNull FragmentActivity fragmentActivity)

  Constructor of a SectionsPagerAdapter for the given `FragmentActivity`.
- int getItemCount ()

  Returns the total number of items in the data set held by the adapter.
- Fragment createFragment (int position)

  Provides a new `IntroductionFragment` associated with the specified position.

5.49.1 Detailed Description

A `FragmentStateAdapter` that returns an `IntroductionFragment` corresponding to one of the sections/tabs/pages.
Definition at line 12 of file SectionsPagerAdapter.java.

5.49.2 Constructor & Destructor Documentation

5.49.2.1 SectionsPagerAdapter()

```
SectionsPagerAdapter (
                @NonNull FragmentActivity fragmentActivity )
```

Constructor of a SectionsPagerAdapter for the given `FragmentActivity`.

Parameters

fragmentActivity	`FragmentActivity` that uses this adapter.

Definition at line 18 of file SectionsPagerAdapter.java.

5.49.3 Member Function Documentation

5.49.3.1 createFragment()

```
Fragment createFragment (
                int position )
```

Provides a new `IntroductionFragment` associated with the specified position.

Parameters

position	`Integer` defining the page that is queried.

Returns

A `Fragment` that will be shown at the specified position.

Definition at line 39 of file SectionsPagerAdapter.java.
References IntroductionFragment.newInstance().

5.49.3.2 getItemCount()

```
int getItemCount ( )
```

Returns the total number of items in the data set held by the adapter.
Essentially queries `IntroductionFragment#PAGES_COUNT`.

Returns

An `Integer` defining the number of pages.

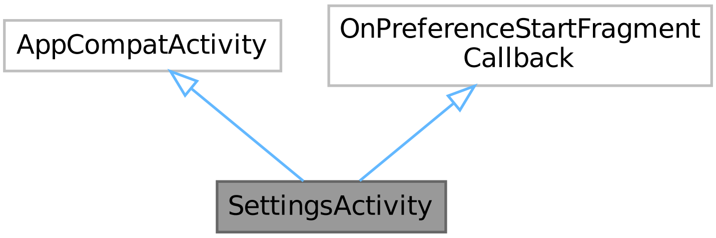
Definition at line 28 of file SectionsPagerAdapter.java.
References IntroductionFragment.PAGES_COUNT.
The documentation for this class was generated from the following file:

- gwent/ui/introduction/SectionsPagerAdapter.java

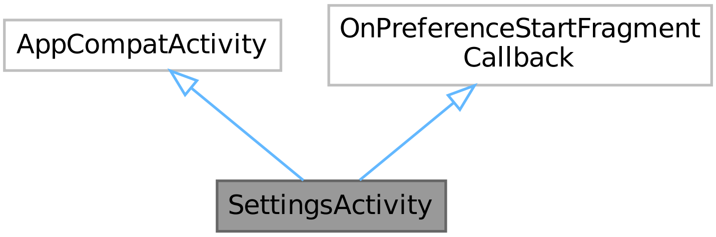
5.50 SettingsActivity Class Reference

An `AppCompatActivity` implementing `PreferenceFragmentCompat.OnPreferenceStartFragmentCallback` that is used by the user to manage the `SharedPreferences` of the application.

Inheritance diagram for SettingsActivity:



Collaboration diagram for SettingsActivity:



Public Member Functions

- boolean onOptionsItemSelected (@NonNull MenuItem item)

  Called whenever a `MenuItem` in the options menu is selected.
- boolean onPreferenceStartFragment (@NonNull PreferenceFragmentCompat caller, @NonNull Preference pref)

  Called when the user has clicked on a `Preference` that has a `Fragment` class name associated with it.

Protected Member Functions

- void onCreate (@Nullable Bundle savedInstanceState)

  Initializes layout and `ActionBar` as well as creates and displays a new `SettingsHeaderFragment`.
- void onSaveInstanceState (@NonNull Bundle outState)

  Saves the currently visible `Fragment` into the given `Bundle`.

Static Private Attributes

- static final String CURRENT_FRAGMENT_KEY = "currentFragment"

  `String` constant defining the key in savedInstanceState `Bundle`s where the last shown `Fragment` is saved.

5.50.1 Detailed Description

An `AppCompatActivity` implementing `PreferenceFragmentCompat.OnPreferenceStartFragmentCallback` that is used by the user to manage the `SharedPreferences` of the application.
Definition at line 24 of file SettingsActivity.java.

5.50.2 Member Function Documentation

5.50.2.1 onCreate()

```
void onCreate (
            @Nullable Bundle savedInstanceState )  [protected]
```

Initializes layout and `ActionBar` as well as creates and displays a new `SettingsHeaderFragment`.

Parameters

savedInstanceState	If the activity is being re-initialized after previously being shut down then this Bundle contains the data it most recently supplied in `onSaveInstanceState`. ***Note: Otherwise it is null.***

Definition at line 38 of file SettingsActivity.java.
References SettingsActivity.CURRENT_FRAGMENT_KEY.

5.50.2.2 onOptionsItemSelected()

```
boolean onOptionsItemSelected (
            @NonNull MenuItem item )
```

Called whenever a `MenuItem` in the options menu is selected.
Returns to the calling `android.app.Activity` when the `android.R.id#home` item was selected.

Parameters

item	`MenuItem` that was selected.

Returns

`Boolean` defining whether the call has been handled.

Definition at line 85 of file SettingsActivity.java.

5.50.2.3 onPreferenceStartFragment()

```
boolean onPreferenceStartFragment (
            @NonNull PreferenceFragmentCompat caller,
            @NonNull Preference pref )
```

Called when the user has clicked on a `Preference` that has a `Fragment` class name associated with it.
Switches to an instance of the given `Fragment`.

Parameters

caller	`PreferenceFragmentCompat` requesting navigation.
pref	`Preference` requesting the `Fragment`.

Returns

`Boolean` defining whether the `Fragment` creation has been handled.

Definition at line 102 of file SettingsActivity.java.

5.50.2.4 onSaveInstanceState()

```
void onSaveInstanceState (
            @NonNull Bundle outState )  [protected]
```

Saves the currently visible `Fragment` into the given `Bundle`.

Parameters

outState	Bundle in which the currently used `Fragment` is saved.

Definition at line 66 of file SettingsActivity.java.
References SettingsActivity.CURRENT_FRAGMENT_KEY.

5.50.3 Member Data Documentation

5.50.3.1 CURRENT_FRAGMENT_KEY

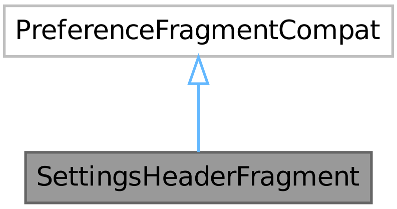
```
final String CURRENT_FRAGMENT_KEY = "currentFragment"  [static], [private]
```

`String` constant defining the key in savedInstanceState `Bundle`s where the last shown `Fragment` is saved.
Definition at line 29 of file SettingsActivity.java.
Referenced by SettingsActivity.onCreate(), and SettingsActivity.onSaveInstanceState().
The documentation for this class was generated from the following file:

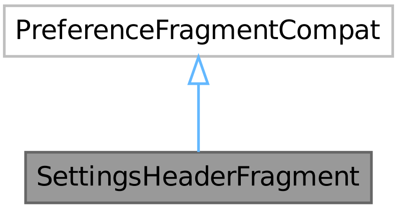
- gwent/ui/settings/SettingsActivity.java

5.51 SettingsHeaderFragment Class Reference

A `PreferenceFragmentCompat` class encapsulating the main preference screen, i.e., the `Preference`s defined in `R.xml#header_preferences`.
Inheritance diagram for SettingsHeaderFragment:



Collaboration diagram for SettingsHeaderFragment:



Public Member Functions

- void onCreatePreferences (Bundle savedInstanceState, String rootKey)

  Called during `onCreate(Bundle)` to supply the preferences for this fragment.

5.51.1 Detailed Description

A `PreferenceFragmentCompat` class encapsulating the main preference screen, i.e., the `Preference`s defined in `R.xml#header_preferences`.
Definition at line 18 of file SettingsHeaderFragment.java.

5.51.2 Member Function Documentation

5.51.2.1 onCreatePreferences()

```
void onCreatePreferences (
            Bundle savedInstanceState,
            String rootKey )
```

Called during `onCreate(Bundle)` to supply the preferences for this fragment.
Sets shown `Preference`s from `R.xml#header_preferences` and registers an `androidx.preference.Preference` on the `Preference` at `R.string#preference_key_introduction` to start a new `IntroductionActivity`.

Parameters

savedInstanceState	If the fragment is being re-created from a previous saved state, this is the state.
rootKey	If non-null, this preference fragment should be rooted at the `androidx.preference.PreferenceScreen` with this key.

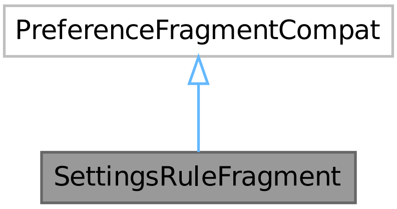
Definition at line 31 of file SettingsHeaderFragment.java.
The documentation for this class was generated from the following file:

- gwent/ui/settings/SettingsHeaderFragment.java

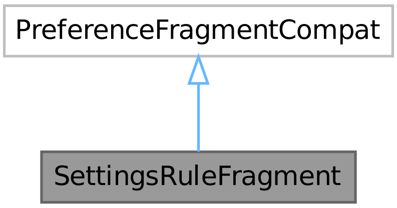
5.52 SettingsRuleFragment Class Reference

A `PreferenceFragmentCompat` class encapsulating the rule preference screen, i.e., the rule sections defined in `R.xml#rule_preferences`.

Inheritance diagram for SettingsRuleFragment:



Collaboration diagram for SettingsRuleFragment:



Public Member Functions

- void onCreatePreferences (Bundle savedInstanceState, String rootKey)

  Called during `onCreate(Bundle)` to supply the preferences for this fragment.

Private Member Functions

- Preference.OnPreferenceClickListener getSectionClickListener (@NonNull RuleSection section)

  Creates an `Preference.OnPreferenceClickListener` that starts a new `RuleActivity` for the given `RuleSection` using an `Intent`.

5.52.1 Detailed Description

A `PreferenceFragmentCompat` class encapsulating the rule preference screen, i.e., the rule sections defined in `R.xml#rule_preferences`.

Definition at line 31 of file SettingsRuleFragment.java.

5.52.2 Member Function Documentation

5.52.2.1 getSectionClickListener()

```
Preference.OnPreferenceClickListener getSectionClickListener (
            @NonNull RuleSection section ) [private]
```

Creates an `Preference.OnPreferenceClickListener` that starts a new `RuleActivity` for the given `RuleSection` using an `Intent`.

The Intent provides the requested RuleSection to the RuleActivity using Intent#putExtra(String, Serializ
with RuleActivity#INTENT_EXTRA_TAG as a tag.

Parameters

section	RuleSection that is requested.

Returns

An Preference.OnPreferenceClickListener calling a RuleActivity.

Definition at line 43 of file SettingsRuleFragment.java.
References RuleActivity.INTENT_EXTRA_TAG.
Referenced by SettingsRuleFragment.onCreatePreferences().

5.52.2.2 onCreatePreferences()

```
void onCreatePreferences (
            Bundle savedInstanceState,
            String rootKey )
```

Called during onCreate(Bundle) to supply the preferences for this fragment.
Sets shown Preferences from R.xml#rule_preferences. Also provides each element with an Preference.OnPreferenceClickListener that starts a new RuleActivity for the respective RuleSection.

Parameters

savedInstanceState	If the fragment is being re-created from a previous saved state, this is the state.
rootKey	If non-null, this preference fragment should be rooted at the androidx.preference.PreferenceScreen with this key.

See also

#getSectionClickListener(RuleSection)

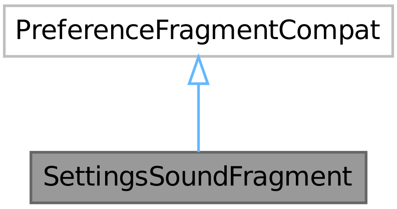
Definition at line 66 of file SettingsRuleFragment.java.
References SettingsRuleFragment.getSectionClickListener().
The documentation for this class was generated from the following file:

- gwent/ui/settings/SettingsRuleFragment.java

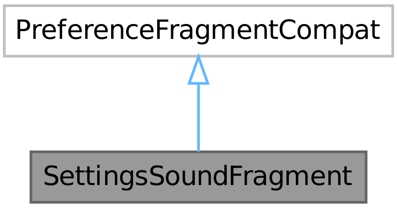
5.53 SettingsSoundFragment Class Reference

A PreferenceFragmentCompat class encapsulating the sound preference screen, i.e., the Preferences defined in R.xml#sound_preferences.

Inheritance diagram for SettingsSoundFragment:



Collaboration diagram for SettingsSoundFragment:



Public Member Functions

- void onCreatePreferences (Bundle savedInstanceState, String rootKey)

    Called during `onCreate(Bundle)` to supply the preferences for this fragment.

5.53.1 Detailed Description

A `PreferenceFragmentCompat` class encapsulating the sound preference screen, i.e., the `Preference`s defined in `R.xml#sound_preferences`.
Definition at line 16 of file SettingsSoundFragment.java.

5.53.2 Member Function Documentation

5.53.2.1 onCreatePreferences()

```
void onCreatePreferences (
            Bundle savedInstanceState,
            String rootKey )
```

Called during `onCreate(Bundle)` to supply the preferences for this fragment.
Sets shown `Preference`s from `R.xml#sound_preferences`.

Parameters

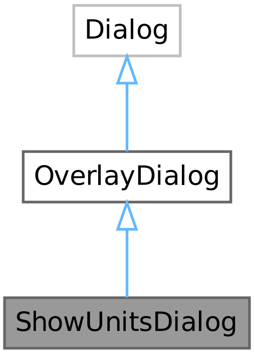
savedInstanceState	If the fragment is being re-created from a previous saved state, this is the state.
rootKey	If non-null, this preference fragment should be rooted at the `androidx.preference.PreferenceScreen` with this key.

Definition at line 27 of file SettingsSoundFragment.java.
The documentation for this class was generated from the following file:

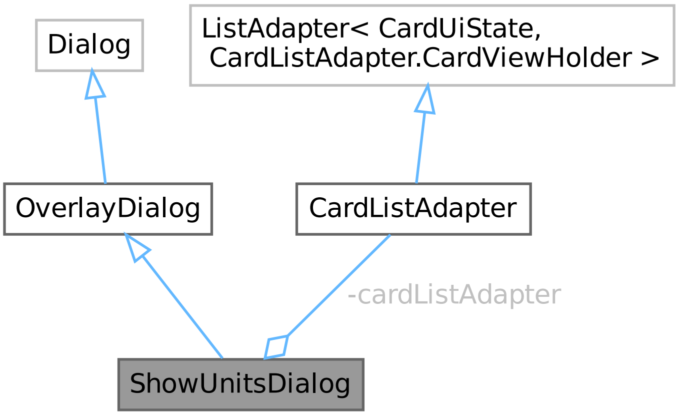
- gwent/ui/settings/SettingsSoundFragment.java

5.54 ShowUnitsDialog Class Reference

An OverlayDialog used to list the units of a certain row and enabling the user to copy, add, or delete new units.
Inheritance diagram for ShowUnitsDialog:



Collaboration diagram for ShowUnitsDialog:



Public Member Functions

- RowType getRow ()

  Returns the row this ShowUnitsDialog is representing.

Static Public Member Functions

- static Single< ShowUnitsDialog > getDialog (@NonNull Context context, @NonNull RowType row, @NonNull SoundManager soundManager)

  Creates a new ShowUnitsDialog in the given Context and for the given row.

Protected Member Functions

- void onCreate (Bundle savedInstanceState)

  Initializes the RecyclerView by connecting it to the #cardListAdapter and sets View.OnClickListener for each button.

Protected Member Functions inherited from OverlayDialog

- OverlayDialog (@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)

  Constructor of an OverlayDialog in the given Context, with the given layout and dismissViewId.
- OverlayDialog (@NonNull Context context, @LayoutRes int layout)

  Constructor of an OverlayDialog in the given Context and with the given layout.

Private Member Functions

- ShowUnitsDialog (@NonNull Context context, @NonNull RowType row, @NonNull CardListAdapter cardListAdapter, @NonNull SoundManager soundManager)

  Constructor of a ShowUnitsDialog shown in the given Context and with the given CardListAdapter.

Private Attributes

- final RowType row

  RowType defining which row is represented by this Dialog.
- final CardListAdapter cardListAdapter

  CardListAdapter offering a list of CardUiState objects to the UI.
- final SoundManager soundManager

  SoundManager used to play a Sound whenever a card is added or removed.
- final CompositeDisposable disposables = new CompositeDisposable()

  CompositeDisposable keeping track of all subscriptions to observables made by this class.
- RecyclerView unitRecyclerView

  `RecyclerView` *presenting the units provided by the* `cardListAdapter`*.*

Additional Inherited Members

Static Package Attributes inherited from OverlayDialog

- static final int NO_DISMISS_VIEW = -1

  Integer used as #dismissViewId, when the Dialog shouldn't be cancelable by clicking on a View.

5.54.1 Detailed Description

An OverlayDialog used to list the units of a certain row and enabling the user to copy, add, or delete new units.

See also

CardListAdapter

Definition at line 31 of file ShowUnitsDialog.java.

5.54.2 Constructor & Destructor Documentation

5.54.2.1 ShowUnitsDialog()

```
ShowUnitsDialog (
             @NonNull Context context,
             @NonNull RowType row,
             @NonNull CardListAdapter cardListAdapter,
             @NonNull SoundManager soundManager )  [private]
```

Constructor of a ShowUnitsDialog shown in the given Context and with the given CardListAdapter.

Parameters

context	Context this Dialog is shown in.
row	RowType defining which row all shown units belong to.
cardListAdapter	CardListAdapter providing an always up-to-date list of CardUiState objects for a certain row.
soundManager	SoundManager used to play sounds when cards are added or removed.

Definition at line 78 of file ShowUnitsDialog.java.
References ShowUnitsDialog.cardListAdapter, ShowUnitsDialog.row, and ShowUnitsDialog.soundManager.

5.54.3 Member Function Documentation

5.54.3.1 getDialog()

```
static Single< ShowUnitsDialog > getDialog (
             @NonNull Context context,
             @NonNull RowType row,
             @NonNull SoundManager soundManager )  [static]
```

Creates a new ShowUnitsDialog in the given Context and for the given row.

Parameters

context	Context the Dialog is shown in.
row	RowType defining the row that is represented by this Dialog.
soundManager	SoundManager used to play sounds when cards are added or removed.

Returns

A Single emitting the created ShowUnitsDialog.

Definition at line 94 of file ShowUnitsDialog.java.
References ShowUnitsDialog.disposables, GwentApplication.getRepository(), ShowUnitsDialog.row, and ShowUnitsDialog.soundManager.

5.54.3.2 getRow()

```
RowType getRow ( )
```

Returns the row this ShowUnitsDialog is representing.

Returns

A RowType defining the represented row.

See also

#row

Definition at line 188 of file ShowUnitsDialog.java.
References ShowUnitsDialog.row.

5.54.3.3 onCreate()

```
void onCreate (
             Bundle savedInstanceState )  [protected]
```

Initializes the RecyclerView by connecting it to the #cardListAdapter and sets View.OnClickListener for each button. Also registers a RecyclerView.AdapterDataObserver responsible for scrolling to the end of the RecyclerView whenever an item is being inserted.

See also

CardListAdapter::registerAdapterDataObserver(RecyclerView.AdapterDataObserver)

Parameters

savedInstanceState	If this dialog is being reinitialized after the hosting activity was previously shut down, holds the result from the most recent call to `onSaveInstanceState`, or null if this is the first time.

Smoothly scrolls to the last item in the RecyclerView to show the user that an item has been inserted.

Parameters

positionStart	Integer defining the first position from where new items are inserted.
itemCount	Integer defining how many items have been inserted.

See also

RecyclerView::smoothScrollToPosition(int)

Reimplemented from OverlayDialog.
Definition at line 142 of file ShowUnitsDialog.java.
References ShowUnitsDialog.cardListAdapter, ShowUnitsDialog.disposables, ShowUnitsDialog.soundManager, and ShowUnitsDialog.unitRecyclerView.

5.54.4 Member Data Documentation

5.54.4.1 cardListAdapter

```
final CardListAdapter cardListAdapter  [private]
```

CardListAdapter offering a list of CardUiState objects to the UI.
Definition at line 44 of file ShowUnitsDialog.java.
Referenced by ShowUnitsDialog.ShowUnitsDialog(), and ShowUnitsDialog.onCreate().

5.54.4.2 disposables

```
final CompositeDisposable disposables = new CompositeDisposable()  [private]
```

CompositeDisposable keeping track of all subscriptions to observables made by this class.
Is being disposed in an android.content.DialogInterface.OnDismissListener that is set in #onCreate().

See also

android.content.DialogInterface.OnDismissListener

#onCreate(Bundle)

Definition at line 63 of file ShowUnitsDialog.java.
Referenced by ShowUnitsDialog.getDialog(), and ShowUnitsDialog.onCreate().

5.54.4.3 row

```
final RowType row  [private]
```

RowType defining which row is represented by this Dialog.

See also

#getRow()

Definition at line 38 of file ShowUnitsDialog.java.
Referenced by ShowUnitsDialog.ShowUnitsDialog(), ShowUnitsDialog.getDialog(), and ShowUnitsDialog.getRow().

5.54.4.4 soundManager

`final SoundManager soundManager [private]`
SoundManager used to play a Sound whenever a card is added or removed.

See also

SoundManager::playCardAddSound(RowType, boolean)

SoundManager::playCardRemovedSound()

Definition at line 53 of file ShowUnitsDialog.java.
Referenced by ShowUnitsDialog.ShowUnitsDialog(), ShowUnitsDialog.getDialog(), and ShowUnitsDialog.onCreate().

5.54.4.5 unitRecyclerView

`RecyclerView unitRecyclerView [private]`
`RecyclerView` presenting the units provided by the `cardListAdapter`.
Definition at line 68 of file ShowUnitsDialog.java.
Referenced by ShowUnitsDialog.onCreate().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/cards/ShowUnitsDialog.java

5.55 Sound Class Reference

A data class encapsulating information on a certain sound effect, e.g., its id in a SoundPool and whether it is activated in settings.
Collaboration diagram for Sound:

Package Functions

- String getPreferenceKey ()

  Returns the key where this Sound is activated in the SharedPreferences.
- int getSoundId ()

  Returns the id of this sound effect at the SoundPool.
- boolean isActivated ()

  Returns whether the sound is activated in the settings.
- void setActivated (boolean activated)

  Changes the value of #activated to the given Boolean.
- void setActivated (@NonNull SharedPreferences sharedPreferences)

  Changes the value of #activated based on the given SharedPreferences.

Static Package Functions

- static Sound createSound (@NonNull Context context, @StringRes int preferenceRes, @NonNull SoundPool soundPool, @RawRes int resId)

  Returns a new Sound object encapsulating data for a newly registered sound effect.

Private Member Functions

- Sound (@NonNull String preferenceKey, int soundId, boolean activated)

  Constructor of a Sound encapsulating the given values.

Private Attributes

- final String preferenceKey

  String representing a key of the default SharedPreferences.
- final int soundId

  Integer representing the id of the registered sound at a SoundPool.
- boolean activated

  Boolean defining whether the sound is muted in the settings.

5.55.1 Detailed Description

A data class encapsulating information on a certain sound effect, e.g., its id in a SoundPool and whether it is activated in settings.
Definition at line 20 of file Sound.java.

5.55.2 Constructor & Destructor Documentation

5.55.2.1 Sound()

```
Sound (
            @NonNull String preferenceKey,
            int soundId,
            boolean activated )  [private]
```

Constructor of a Sound encapsulating the given values.
Should only be called by factory method #createSound()

Parameters

preferenceKey	String representing the key where the activated status is defined in SharedPreferences.
soundId	Integer representing the id of the registered sound at a SoundPool.
activated	Boolean defining whether the sound is activated in the settings.

See also

#createSound(Context, int, SoundPool, int)

Definition at line 48 of file Sound.java.
References Sound.activated, Sound.preferenceKey, and Sound.soundId.

5.55.3 Member Function Documentation

5.55.3.1 createSound()

```
static Sound createSound (
            @NonNull Context context,
            @StringRes int preferenceRes,
            @NonNull SoundPool soundPool,
            @RawRes int resId )  [static], [package]
```

Returns a new Sound object encapsulating data for a newly registered sound effect.
Factory method for the Sound class. Gets #preferenceKey from the given Context. Registers the sound with the given resId at the given SoundPool. Decides the value of #activated from SharedPreferences obtained using the given Context.

Parameters

context	Context of the application where this Sound is used.
preferenceRes	Integer referencing the String resource of the preference key for this Sound.
soundPool	SoundPool where the Sound is registered.
resId	Integer referencing the raw resource of the Sound.

Returns

A Sound object referencing the newly registered sound effect.

Definition at line 69 of file Sound.java.
References Sound.activated, Sound.preferenceKey, and Sound.soundId.
Referenced by SoundManager.SoundManager().

5.55.3.2 getPreferenceKey()

```
String getPreferenceKey ( )  [package]
```

Returns the key where this Sound is activated in the SharedPreferences.

Returns

A String referencing the activation option.

Definition at line 86 of file Sound.java.
References Sound.preferenceKey.

5.55.3.3 getSoundId()

```
int getSoundId ( )  [package]
```

Returns the id of this sound effect at the SoundPool.

Returns

An Integer referencing the sound effect at the SoundPool.

Definition at line 95 of file Sound.java.
References Sound.soundId.

5.55.3.4 isActivated()

```
boolean isActivated ( )  [package]
```

Returns whether the sound is activated in the settings.

Returns

A Boolean defining whether the Sound is activated.

Definition at line 104 of file Sound.java.
References Sound.activated.

5.55.3.5 setActivated() [1/2]

```
void setActivated (
            @NonNull SharedPreferences sharedPreferences )  [package]
```

Changes the value of #activated based on the given SharedPreferences.
Only queries the boolean at key #preferenceKey, not whether all sounds are deactivated!

Parameters

sharedPreferences	SharedPreferences containing information on the activated sounds.

Definition at line 124 of file Sound.java.
References Sound.activated, Sound.preferenceKey, and Sound.setActivated().

5.55.3.6 setActivated() [2/2]

```
void setActivated (
            boolean activated )  [package]
```

Changes the value of #activated to the given Boolean.

Parameters

activated	Boolean defining whether the Sound is activated.

See also

#setActivated(SharedPreferences)

Definition at line 114 of file Sound.java.
References Sound.activated.
Referenced by Sound.setActivated().

5.55.4 Member Data Documentation

5.55.4.1 activated

```
boolean activated  [private]
```

Boolean defining whether the sound is muted in the settings.
Definition at line 37 of file Sound.java.
Referenced by Sound.Sound(), Sound.createSound(), Sound.isActivated(), Sound.setActivated(), and Sound.setActivated().

5.55.4.2 preferenceKey

```
final String preferenceKey  [private]
```

String representing a key of the default SharedPreferences.
The referenced preference decides whether the sound is activated (assuming that sounds are not completely disabled).
Definition at line 27 of file Sound.java.
Referenced by Sound.Sound(), Sound.createSound(), Sound.getPreferenceKey(), and Sound.setActivated().

5.55.4.3 soundId

```
final int soundId  [private]
```

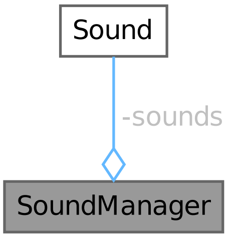
Integer representing the id of the registered sound at a SoundPool.
Definition at line 32 of file Sound.java.
Referenced by Sound.Sound(), Sound.createSound(), and Sound.getSoundId().
The documentation for this class was generated from the following file:

- gwent/ui/sounds/Sound.java

5.56 SoundManager Class Reference

A class responsible for initializing and playing sounds when they are enabled.

Collaboration diagram for SoundManager:



Public Member Functions

- SoundManager (@NonNull Context context)

  Constructor of a SoundManager in the given Context.
- void playSound (@IntRange(from=SOUND_WEATHER_GOOD, to=SOUND_COIN) int soundId)

  Plays the given Sound if it is #activated.
- void playClearWeatherSound ()

  Plays the clear weather sound.
- void playWeatherSound (@NonNull RowType row)

  Plays the weather sound of the given row.
- void playHornSound ()

  Plays the horn sound.
- void playCardAddSound (@NonNull RowType row, boolean epic)

  Plays the card add sound for the given row.
- void playCardRemovedSound ()

  Plays the reset sound.
- void playResetSound ()

  Plays the reset sound.
- void playBurnSound ()

  Plays the burn sound.
- void playCoinSound ()

  Plays the coin sound.

Static Public Attributes

- static final int SOUND_WEATHER_GOOD = 0

  Constant Integer representing the clear weather sound.
- static final int SOUND_WEATHER_FROST = 1

  Constant Integer representing the frost weather sound.
- static final int SOUND_WEATHER_FOG = 2

  Constant Integer representing the fog weather sound.
- static final int SOUND_WEATHER_RAIN = 3

  Constant Integer representing the rain weather sound.
- static final int SOUND_HORN = 4

  Constant Integer representing the horn sound.
- static final int SOUND_CARDS_EPIC = 5

Constant Integer representing the epic unit sound.
- static final int SOUND_CARDS_MELEE = 6

  Constant Integer representing the melee unit sound.
- static final int SOUND_CARDS_RANGE = 7

  Constant Integer representing the range unit sound.
- static final int SOUND_CARDS_SIEGE = 8

  Constant Integer representing the siege unit sound.
- static final int SOUND_RESET = 9

  Constant Integer representing the reset sound.
- static final int SOUND_BURN = 10

  Constant Integer representing the scorch sound.
- static final int SOUND_COIN = 11

  Constant Integer representing the coin-flip sound.

Private Attributes

- final Sound[ ] sounds = new Sound[SOUND_COIN + 1]

  Array of Sound objects representing the different sound effects.
- final SoundPool soundPool

  SoundPool where sound effects are registered.
- final SharedPreferences.OnSharedPreferenceChangeListener changeListener

  OnSharedPreferenceChangeListener that is registered for the default SharedPreferences and updates the Sound←↩ ::active attributes of all #sounds when a preference change occurs.

5.56.1 Detailed Description

A class responsible for initializing and playing sounds when they are enabled.

See also

Sound

Definition at line 23 of file SoundManager.java.

5.56.2 Constructor & Destructor Documentation

5.56.2.1 SoundManager()

```
SoundManager (
            @NonNull Context context )
```

Constructor of a SoundManager in the given Context.
Creates a new #soundPool and registers #sounds using Sound::createSound(). Also registers a new #change←↩ Listener that updates the Sound::activated status when sound settings are updated.

Parameters

context	Context, this SoundManager is used in.

See also

Sound::createSound(Context, int, SoundPool, int)

Definition at line 156 of file SoundManager.java.
References SoundManager.changeListener, Sound.createSound(), SoundManager.SOUND_BURN, SoundManager.SOUND_CARDS
SoundManager.SOUND_CARDS_MELEE, SoundManager.SOUND_CARDS_RANGE, SoundManager.SOUND_CARDS_SIEGE,
SoundManager.SOUND_COIN, SoundManager.SOUND_HORN, SoundManager.SOUND_RESET, SoundManager.SOUND_WEATH
SoundManager.SOUND_WEATHER_FROST, SoundManager.SOUND_WEATHER_GOOD, SoundManager.SOUND_WEATHER_RA
SoundManager.soundPool, and SoundManager.sounds.

5.56.3 Member Function Documentation

5.56.3.1 playBurnSound()

```
void playBurnSound ( )
```

Plays the burn sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 311 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_BURN.

5.56.3.2 playCardAddSound()

```
void playCardAddSound (
             @NonNull RowType row,
             boolean epic )
```

Plays the card add sound for the given row.
If the unit is epic, the epic sound is played. Wrapper for #playSound().

Parameters

row	RowType referencing the row for which the sound should be played.
epic	Boolean defining whether the added unit is epic.

See also

#playSound(int)

Definition at line 268 of file SoundManager.java.
References SoundManager.playSound(), SoundManager.SOUND_CARDS_EPIC, SoundManager.SOUND_CARDS_MELEE, SoundManager.SOUND_CARDS_RANGE, and SoundManager.SOUND_CARDS_SIEGE.

5.56.3.3 playCardRemovedSound()

```
void playCardRemovedSound ( )
```

Plays the reset sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 291 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_RESET.

5.56.3.4 playClearWeatherSound()

```
void playClearWeatherSound ( )
```

Plays the clear weather sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 225 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_WEATHER_GOOD.

5.56.3.5 playCoinSound()

```
void playCoinSound ( )
```

Plays the coin sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 320 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_COIN.

5.56.3.6 playHornSound()

```
void playHornSound ( )
```

Plays the horn sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 255 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_HORN.

5.56.3.7 playResetSound()

```
void playResetSound ( )
```

Plays the reset sound.
Wrapper for #playSound().

See also

#playSound(int)

Definition at line 301 of file SoundManager.java.
References SoundManager.playSound(), and SoundManager.SOUND_RESET.

5.56.3.8 playSound()

```
void playSound (
             @IntRange(from=SOUND_WEATHER_GOOD, to=SOUND_COIN) int soundId )
```

Plays the given Sound if it is #activated.
Given soundId must be either #SOUND_WEATHER_GOOD, #SOUND_WEATHER_FROST, #SOUND_WEATHER_FOG, #SOUND_WEATHER_RAIN, #SOUND_HORN, #SOUND_CARDS_EPIC, #SOUND_CARDS_MELEE, #SOUND_CARDS_RANGE, #SOUND_CARDS_SIEGE, #SOUND_RESET, #SOUND_BURN, or #SOUND_COIN, i.e., $0 \leq soundId \leq 11$ must be true. Alternately, #playClearWeatherSound(), #playWeatherSound(), #playHornSound(), #playCardAddSound(), #playCardRemovedSound(), #playResetSound(), #playBurnSound(), or #playCoinSound() may be used.

Parameters

soundId	Integer representing the Sound that should be played.

Exceptions

IndexOutOfBoundsException	When $0 \leq soundId \leq 11$ is not true.

Definition at line 213 of file SoundManager.java.

References SoundManager.soundPool, and SoundManager.sounds.
Referenced by SoundManager.playBurnSound(), SoundManager.playCardAddSound(), SoundManager.playCardRemovedSound(), SoundManager.playClearWeatherSound(), SoundManager.playCoinSound(), SoundManager.playHornSound(), SoundManager.playResetSound(), and SoundManager.playWeatherSound().

5.56.3.9 playWeatherSound()

```
void playWeatherSound (
            @NonNull RowType row )
```

Plays the weather sound of the given row.
Wrapper for #playSound().

Parameters

row	RowType referencing the row for which the sound should be played.

See also

#playSound(int)

Definition at line 236 of file SoundManager.java.
References SoundManager.playSound(), SoundManager.SOUND_WEATHER_FOG, SoundManager.SOUND_WEATHER_FROST, and SoundManager.SOUND_WEATHER_RAIN.

5.56.4 Member Data Documentation

5.56.4.1 changeListener

```
final SharedPreferences.OnSharedPreferenceChangeListener changeListener [private]
```

OnSharedPreferenceChangeListener that is registered for the default SharedPreferences and updates the Sound↩ ::active attributes of all #sounds when a preference change occurs.
Reference must be kept (even if not used) to avoid garbage collection of the registered listener (see `here` for more information).
Definition at line 145 of file SoundManager.java.
Referenced by SoundManager.SoundManager().

5.56.4.2 SOUND_BURN

```
final int SOUND_BURN = 10 [static]
```

Constant Integer representing the scorch sound.
Index of the respective Sound object in #sounds.

See also

#playBurnSound()

Definition at line 110 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playBurnSound().

5.56.4.3 SOUND_CARDS_EPIC

```
final int SOUND_CARDS_EPIC = 5 [static]
```

Constant Integer representing the epic unit sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 70 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playCardAddSound().

5.56.4.4 SOUND_CARDS_MELEE

```
final int SOUND_CARDS_MELEE = 6  [static]
```

Constant Integer representing the melee unit sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 78 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playCardAddSound().

5.56.4.5 SOUND_CARDS_RANGE

```
final int SOUND_CARDS_RANGE = 7  [static]
```

Constant Integer representing the range unit sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 86 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playCardAddSound().

5.56.4.6 SOUND_CARDS_SIEGE

```
final int SOUND_CARDS_SIEGE = 8  [static]
```

Constant Integer representing the siege unit sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 94 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playCardAddSound().

5.56.4.7 SOUND_COIN

```
final int SOUND_COIN = 11  [static]
```

Constant Integer representing the coin-flip sound.
Index of the respective Sound object in #sounds.

See also

#playCoinSound()

Definition at line 118 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playCoinSound().

5.56.4.8 SOUND_HORN

```
final int SOUND_HORN = 4  [static]
```

Constant Integer representing the horn sound.
Index of the respective Sound object in #sounds.

See also

#playHornSound()

Definition at line 62 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playHornSound().

5.56.4.9 SOUND_RESET

```
final int SOUND_RESET = 9  [static]
```

Constant Integer representing the reset sound.
Index of the respective Sound object in #sounds.

See also

#playResetSound()

Definition at line 102 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), SoundManager.playCardRemovedSound(), and SoundManager.playResetSound().

5.56.4.10 SOUND_WEATHER_FOG

```
final int SOUND_WEATHER_FOG = 2  [static]
```

Constant Integer representing the fog weather sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 46 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playWeatherSound().

5.56.4.11 SOUND_WEATHER_FROST

```
final int SOUND_WEATHER_FROST = 1  [static]
```

Constant Integer representing the frost weather sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 38 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playWeatherSound().

5.56.4.12 SOUND_WEATHER_GOOD

```
final int SOUND_WEATHER_GOOD = 0  [static]
```

Constant Integer representing the clear weather sound.
Index of the respective Sound object in #sounds.

See also

#playClearWeatherSound()

Definition at line 30 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playClearWeatherSound().

5.56.4.13 SOUND_WEATHER_RAIN

```
final int SOUND_WEATHER_RAIN = 3  [static]
```

Constant Integer representing the rain weather sound.
Index of the respective Sound object in #sounds.

See also

#playCardAddSound(RowType, boolean)

Definition at line 54 of file SoundManager.java.
Referenced by SoundManager.SoundManager(), and SoundManager.playWeatherSound().

5.56.4.14 soundPool

```
final SoundPool soundPool  [private]
```

SoundPool where sound effects are registered.

Definition at line 134 of file SoundManager.java.

Referenced by SoundManager.SoundManager(), and SoundManager.playSound().

5.56.4.15 sounds

```
final Sound [] sounds = new Sound[SOUND_COIN + 1]  [private]
```

Array of Sound objects representing the different sound effects.

Indices are the public constants defined here, i.e., #SOUND_WEATHER_GOOD, #SOUND_WEATHER_↩ FROST, #SOUND_WEATHER_FOG, #SOUND_WEATHER_RAIN, #SOUND_HORN, #SOUND_CARDS_EPIC, #SOUND_CARDS_MELEE, #SOUND_CARDS_RANGE, #SOUND_CARDS_SIEGE, #SOUND_RESET, #↩ SOUND_BURN, and #SOUND_COIN.

Definition at line 128 of file SoundManager.java.

Referenced by SoundManager.SoundManager(), and SoundManager.playSound().

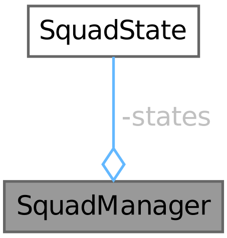
The documentation for this class was generated from the following file:

- gwent/ui/sounds/SoundManager.java

5.57 SquadManager Class Reference

A class used by the CardNumberPickerAdapter to determine the first squad with members or set the default damage according to the squads base damage.

Collaboration diagram for SquadManager:



Static Public Attributes

- static final int MAX_NR_SQUADS = 3

  Integer constant defining how many different squads there may be per row.

Package Functions

- SquadManager (@NonNull List< UnitEntity > units)

  Constructor of a new SquadManager storing information on the squads of the given units.
- int getFirstSquadWithMembers ()

  Returns the lowest squad number referring to a squad that has members.
- void onSquadChanged (@IntRange(from=1, to=MAX_NR_SQUADS) int newVal, @NonNull DamageValuePicker picker)

  Shows a Toast with information about the selected squad and sets the given picker to the SquadState::member↩ BaseDamage of said squad.

Private Attributes

- final SquadState[ ] states = new SquadState[MAX_NR_SQUADS]

  Array of SquadState objects containing one SquadState for each possible squad.

5.57.1 Detailed Description

A class used by the CardNumberPickerAdapter to determine the first squad with members or set the default damage according to the squads base damage.

See also

CardNumberPickerAdapter

Definition at line 24 of file SquadManager.java.

5.57.2 Constructor & Destructor Documentation

5.57.2.1 SquadManager()

```
SquadManager (
            @NonNull List< UnitEntity > units ) [package]
```

Constructor of a new SquadManager storing information on the squads of the given units.

Parameters

units	List of UnitEntity objects that is used to fetch squad information.

Definition at line 41 of file SquadManager.java.
References SquadState.getState(), SquadManager.MAX_NR_SQUADS, and SquadManager.states.

5.57.3 Member Function Documentation

5.57.3.1 getFirstSquadWithMembers()

```
int getFirstSquadWithMembers ( ) [package]
```

Returns the lowest squad number referring to a squad that has members.
If no squad has members, i.e., there are no units with the Ability::BINDING ability, 1 is returned.

Returns

An Integer referencing to the first squad with members.

Definition at line 55 of file SquadManager.java.
References SquadState.getSquadNumber(), SquadState.hasMembers(), and SquadManager.states.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), and CardNumberPickerAdapter.addSelectedUnits().

5.57.3.2 onSquadChanged()

```
void onSquadChanged (
            @IntRange(from=1, to=MAX_NR_SQUADS) int newVal,
            @NonNull DamageValuePicker picker ) [package]
```

Shows a Toast with information about the selected squad and sets the given picker to the SquadState::member↩BaseDamage of said squad.

Parameters

newVal	Integer representing the newly selected squad.
picker	DamageValuePicker that is updated.

Exceptions

IllegalArgumentException	When newVal is not between 1 and #MAX_NR_SQUADS.

See also

SquadState::getMemberBaseDamage()

Definition at line 72 of file SquadManager.java.
References SquadState.getMemberBaseDamage(), SquadState.getSquadMembers(), SquadState.getSquadNumber(), SquadState.hasMembers(), SquadManager.MAX_NR_SQUADS, and SquadManager.states.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter().

5.57.4 Member Data Documentation

5.57.4.1 MAX_NR_SQUADS

```
final int MAX_NR_SQUADS = 3  [static]
```

Integer constant defining how many different squads there may be per row.
Definition at line 28 of file SquadManager.java.
Referenced by CardNumberPickerAdapter.CardNumberPickerAdapter(), SquadManager.SquadManager(), and SquadManager.onSquadChanged().

5.57.4.2 states

```
final SquadState [] states = new SquadState[MAX_NR_SQUADS]  [private]
```

Array of SquadState objects containing one SquadState for each possible squad.
Definition at line 34 of file SquadManager.java.
Referenced by SquadManager.SquadManager(), SquadManager.getFirstSquadWithMembers(), and SquadManager.onSquadChange
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/SquadManager.java

5.58 SquadState Class Reference

A data class encapsulating information about the squad defined by #squadNumber, i.e., how many #squadMembers are there and what is the #memberBaseDamage of units in this squad.
Collaboration diagram for SquadState:

Package Functions

- boolean hasMembers ()

  Returns whether the represented squad has members, i.e., whether #squadMembers is greater than 0.
- int getSquadNumber ()

  Returns the number of the represented squad.
- int getSquadMembers ()

  Returns the number of members in the represented squad.
- int getMemberBaseDamage ()

  Returns the base damage of members in the represented squad.

Static Package Functions

- static SquadState getState (@IntRange(from=1, to=MAX_NR_SQUADS) int squadNumber, @NonNull List< UnitEntity > units)

  Creates a new SquadState representing the squad with the given #squadNumber.

Private Member Functions

- SquadState (@IntRange(from=1, to=MAX_NR_SQUADS) int squadNumber, @IntRange(from=0) int squadMembers, @IntRange(from=0, to=NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int memberBaseDamage)

  Constructor of a SquadState with the given #squadNumber, #squadMembers, and #memberBaseDamage.

Private Attributes

- final int squadNumber

  Integer containing the number of the represented squad.
- final int squadMembers

  Integer representing the number of members in the squad.
- final int memberBaseDamage

  Integer containing the base damage of members of this squad.

5.58.1 Detailed Description

A data class encapsulating information about the squad defined by #squadNumber, i.e., how many #squadMembers are there and what is the #memberBaseDamage of units in this squad.

See also

SquadManager

Definition at line 19 of file SquadState.java.

5.58.2 Constructor & Destructor Documentation

5.58.2.1 SquadState()

```
SquadState (
            @IntRange(from=1, to=MAX_NR_SQUADS) int squadNumber,
            @IntRange(from=0) int squadMembers,
            @IntRange(from=0, to=NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int memberBaseDamage )
[private]
```

Constructor of a SquadState with the given #squadNumber, #squadMembers, and #memberBaseDamage.
Should only be used by #getState().

Parameters

squadNumber	Integer containing the number of the represented squad.
squadMembers	Integer representing the number of members in the squad.
memberBaseDamage	Integer containing the base damage of members of this squad.

See also

#getState(int, List)

Exceptions

IllegalArgumentException	When one of the parameters doesn't meet its IntRange constraint.

Definition at line 55 of file SquadState.java.
References SquadState.memberBaseDamage, SquadState.squadMembers, and SquadState.squadNumber.

5.58.3 Member Function Documentation

5.58.3.1 getMemberBaseDamage()

```
int getMemberBaseDamage ( )  [package]
```

Returns the base damage of members in the represented squad.

See also

#memberBaseDamage

Returns

An Integer defining the base damage of units in the represented squad.

Definition at line 127 of file SquadState.java.
References SquadState.memberBaseDamage.
Referenced by SquadManager.onSquadChanged().

5.58.3.2 getSquadMembers()

```
int getSquadMembers ( )  [package]
```

Returns the number of members in the represented squad.

See also

#squadMembers

Returns

An Integer defining how many members are in the represented squad.

Definition at line 117 of file SquadState.java.
References SquadState.squadMembers.
Referenced by SquadManager.onSquadChanged().

5.58.3.3 getSquadNumber()

```
int getSquadNumber ( )  [package]
```

Returns the number of the represented squad.

See also

#squadNumber

Returns

An Integer defining which squad is represented.

Definition at line 107 of file SquadState.java.
References SquadState.squadNumber.
Referenced by SquadManager.getFirstSquadWithMembers(), and SquadManager.onSquadChanged().

5.58.3.4 getState()

```
static SquadState getState (
            @IntRange(from=1, to=MAX_NR_SQUADS) int squadNumber,
            @NonNull List< UnitEntity > units )  [static], [package]
```

Creates a new SquadState representing the squad with the given #squadNumber.
Information for #squadMembers and #memberBaseDamage is retrieved from the given List of UnitEntity objects.

Parameters

squadNumber	Integer containing the number of the represented squad.
units	List of UnitEntity objects used to count squad-members.

Returns

A SquadState object that is newly created from the given List of units.

Exceptions

IllegalArgumentException	When the given squad number is not between 1 and #MAX_NR_SQUADS.

Definition at line 80 of file SquadState.java.
References SquadState.squadNumber.
Referenced by SquadManager.SquadManager().

5.58.3.5 hasMembers()

```
boolean hasMembers ( )  [package]
```

Returns whether the represented squad has members, i.e., whether #squadMembers is greater than 0.

Returns

A Boolean defining whether the represented squad has members or not.

Definition at line 97 of file SquadState.java.
References SquadState.squadMembers.
Referenced by SquadManager.getFirstSquadWithMembers(), and SquadManager.onSquadChanged().

5.58.4 Member Data Documentation

5.58.4.1 memberBaseDamage

```
final int memberBaseDamage  [private]
```

Integer containing the base damage of members of this squad.
If members have different UnitEntity::damage values, it is undefined which of them is picked. If #squadMembers is 0, #memberBaseDamage defaults to 5.

See also

#getMemberBaseDamage()

Definition at line 43 of file SquadState.java.
Referenced by SquadState.SquadState(), and SquadState.getMemberBaseDamage().

5.58.4.2 squadMembers

```
final int squadMembers  [private]
```

Integer representing the number of members in the squad.

See also

#getSquadMembers()

Definition at line 34 of file SquadState.java.
Referenced by SquadState.SquadState(), SquadState.getSquadMembers(), and SquadState.hasMembers().

5.58.4.3 squadNumber

`final int squadNumber [private]`

Integer containing the number of the represented squad.
Equivalent to UnitEntity::squad of the units in the squad.

See also

#getSquadNumber()

Definition at line 27 of file SquadState.java.
Referenced by SquadState.SquadState(), SquadState.getSquadNumber(), and SquadState.getState().
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/SquadState.java

5.59 StringValuePicker< T extends Comparable Class Template Reference

A ValuePicker displaying a resource String for each element in selectableValues.
Collaboration diagram for StringValuePicker< T extends Comparable:

5.59.1 Detailed Description

A ValuePicker displaying a resource String for each element in selectableValues.
The resource String is retrieved from displayIntegers by using the associated Integer as the String id in #get↩ DisplayString().

See also

#getDisplayString(Comparable)

Definition at line 17 of file StringValuePicker.java.
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/StringValuePicker.java

5.60 ResetDialogUseCase.Trigger Enum Reference

An `Enum` listing the possible triggers of a reset.
Collaboration diagram for ResetDialogUseCase.Trigger:

Public Attributes

- BUTTON_CLICK

  Represents, that a reset was triggered by a click on the reset button.
- FACTION_SWITCH

  Represents that a reset was triggered by a faction switch.

5.60.1 Detailed Description

An `Enum` listing the possible triggers of a reset.
Definition at line 130 of file ResetDialogUseCase.java.

5.60.2 Member Data Documentation

5.60.2.1 BUTTON_CLICK

`BUTTON_CLICK`
Represents, that a reset was triggered by a click on the reset button.
Definition at line 134 of file ResetDialogUseCase.java.

5.60.2.2 FACTION_SWITCH

`FACTION_SWITCH`
Represents that a reset was triggered by a faction switch.
Only relevant if faction reset is activated, i.e., the preference at the key referenced by `R.string#preference_key_faction_r`
is true.
Definition at line 141 of file ResetDialogUseCase.java.
Referenced by ResetAlertDialogBuilderAdapter.setTrigger().
The documentation for this enum was generated from the following file:

- gwent/domain/cases/ResetDialogUseCase.java

5.61 UnitDao Interface Reference

An interface defining data access operations related to `units` table.
Collaboration diagram for UnitDao:

Public Member Functions

- Completable insertUnit (@NonNull UnitEntity unit)

  Inserts the given UnitEntity into `units`.
- Completable insertUnit (boolean epic, @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability, @IntRange(from=1) @Nullable Integer squad, @NonNull RowType row)

  Inserts a UnitEntity with the given values into `units`.
- Completable deleteUnit (int id)

  Deletes the UnitEntity object with the given id from `units`.
- Completable deleteUnits (@NonNull Collection< UnitEntity > units)

Deletes the given UnitEntity objects from `units`.

- Single< UnitEntity > getUnit (int id)

  Fetches the UnitEntity object with the given id from `units`.
- Single< List< UnitEntity > > getUnits (@NonNull RowType row)

  Fetches all UnitEntity objects from `units` in the given row.
- Flowable< List< UnitEntity > > getUnitsFlowable (@NonNull RowType row)

  Fetches a Flowable for all UnitEntity objects from `units` in the given row.
- Single< List< UnitEntity > > getUnits ()

  Fetches all UnitEntity objects from `units`.
- Single< Integer > countUnits (@NonNull RowType row)

  Counts UnitEntity objects in `units` in the given row.
- Single< Integer > countUnits ()

  Counts UnitEntity objects in `units`.
- Flowable< Boolean > hasNonEpicUnitsFlowable ()

  Fetches a Flowable of Booleans defining whether `units` contains units that are not epic.

5.61.1 Detailed Description

An interface defining data access operations related to `units` table.
The implementation is generated by the Room framework.
Definition at line 25 of file UnitDao.java.

5.61.2 Member Function Documentation

5.61.2.1 countUnits() [1/2]

```
Single< Integer > countUnits ( )
```

Counts UnitEntity objects in `units`.

Returns

A Single tracking operation status and returning the value.

5.61.2.2 countUnits() [2/2]

```
Single< Integer > countUnits (
            @NonNull RowType row )
```

Counts UnitEntity objects in `units` in the given row.

Parameters

row	RowType defining the UnitEntity::row foreign key.

Returns

A Single tracking operation status and returning the value.

Referenced by UnitRepository.countUnits(), and UnitRepository.countUnits().

5.61.2.3 deleteUnit()

```
Completable deleteUnit (
            int id )
```

Deletes the UnitEntity object with the given id from `units`.

Parameters

id	Integer representing the UnitEntity::id of the deleted UnitEntity.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.delete().

5.61.2.4 deleteUnits()

```
Completable deleteUnits (
            @NonNull Collection< UnitEntity > units )
```

Deletes the given UnitEntity objects from `units`.

Parameters

units	List of UnitEntity that are being inserted.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.delete().

5.61.2.5 getUnit()

```
Single< UnitEntity > getUnit (
            int id )
```

Fetches the UnitEntity object with the given id from `units`.

Parameters

id	Integer representing the UnitEntity::id of the requested UnitEntity.

Returns

A Single tracking operation status and returning the value.

Referenced by UnitRepository.getUnit().

5.61.2.6 getUnits() [1/2]

```
Single< List< UnitEntity > > getUnits ( )
```

Fetches all UnitEntity objects from `units`.

Returns

A Single tracking operation status and returning the value.

5.61.2.7 getUnits() [2/2]

```
Single< List< UnitEntity > > getUnits (
            @NonNull RowType row )
```

Fetches all UnitEntity objects from `units` in the given row.

See also

#getUnitsFlowable(RowType)

Parameters

row	RowType defining the UnitEntity::row foreign key.

Returns

A Single tracking operation status and returning the value.

Referenced by UnitRepository.getUnits(), and UnitRepository.getUnits().

5.61.2.8 getUnitsFlowable()

```
Flowable< List< UnitEntity > > getUnitsFlowable (
            @NonNull RowType row )
```

Fetches a Flowable for all UnitEntity objects from `units` in the given row.

Parameters

row	RowType defining the UnitEntity::row foreign key.

Returns

A Flowable emitting the values.

See also

#getUnits(RowType)

Referenced by UnitRepository.getUnitsFlowable().

5.61.2.9 hasNonEpicUnitsFlowable()

```
Flowable< Boolean > hasNonEpicUnitsFlowable ( )
```

Fetches a Flowable of Booleans defining whether `units` contains units that are not epic.

Returns

A Flowable emitting the values.

Referenced by UnitRepository.hasNonEpicUnitsFlowable().

5.61.2.10 insertUnit() [1/2]

```
Completable insertUnit (
            @NonNull UnitEntity unit )
```

Inserts the given UnitEntity into `units`.
If the same UnitEntity::id is already in the table, the new insert is ignored.

Parameters

unit	UnitEntity that is being inserted.

Returns

A Completable tracking operation status.

Referenced by UnitRepository.insertUnit(), and UnitRepository.reset().

5.61.2.11 insertUnit() [2/2]

```
Completable insertUnit (
            boolean epic,
            @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage,
            @NonNull Ability ability,
            @IntRange(from=1) @Nullable Integer squad,
            @NonNull RowType row )
```

Inserts a UnitEntity with the given values into `units`.

Parameters

epic	Boolean defining whether the unit is epic.
damage	Non-Negative value defining the base-damage of the unit.
ability	Ability defining the ability of the unit.
squad	Non-Negative value defining the squad of units, if UnitEntity::ability is Ability::BINDING, or `null`.
row	RowType defining the UnitEntity::row foreign key.

Returns

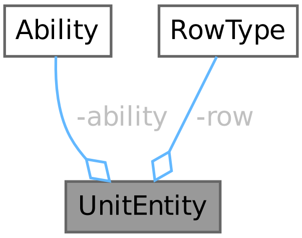
A Completable tracking operation status.

The documentation for this interface was generated from the following file:

- gwent/data/UnitDao.java

5.62 UnitEntity Class Reference

A class representing a card on the game board.
Collaboration diagram for UnitEntity:



Public Member Functions

- int calculateDamage (@NonNull DamageCalculator calculator)

  Calculates the damage of this unit when (de-)buffed.
- Color isBuffed (@NonNull DamageCalculator calculator)

  Returns whether this unit is (de-)buffed.
- String toString (@NonNull Context context)

  Returns a string representation of this unit.
- int getId ()

  Getter for #id.
- boolean isEpic ()

  Getter for #epic.

- int getDamage ()

  Getter for #damage.
- Ability getAbility ()

  Getter for #ability.
- Integer getSquad ()

  Getter for #squad.
- RowType getRow ()

  Getter for #row.

Static Public Member Functions

- static String collectionToString (@NonNull Context context, @NonNull Collection< UnitEntity > units)

  Creates a String containing the descriptions of all units in the given collection, separated by commas.

Static Public Attributes

- static final Integer[ ] EPIC_DAMAGE_VALUES = new Integer[ ]{0, 7, 8, 10, 11, 15}

  Integer array containing every damage value an epic unit may have.
- static final int NON_EPIC_DAMAGE_VALUES_UPPER_BOUND = 20

  Integer defining what the maximum damage of a non-epic UnitEntity can be.

Package Functions

- UnitEntity (boolean epic, @IntRange(from=0) int damage, @NonNull Ability ability, @IntRange(from=1) @Nullable Integer squad, @NonNull RowType row)

  Constructor of a UnitEntity.
- void setId (int id)

  Setter for #id.
- void setEpic (boolean epic)

  Setter for #epic.
- void setDamage (@IntRange(from=0) int damage)

  Setter for #damage.
- void setAbility (@NonNull Ability ability)

  Setter for #ability.
- void setSquad (@IntRange(from=1) @Nullable Integer squad)

  Setter for #squad.
- void setRow (@NonNull RowType row)

  Setter for #row.

Private Attributes

- int id

  Defines the primary key of the represented unit.
- boolean epic

  Defines whether the represented card is epic.
- int damage

  Defines the base-damage of the represented card.
- Ability ability

  Defines the Ability of the represented card.
- Integer squad

  Defines the squad the unit belongs to if #ability is Ability::BINDING.
- RowType row

  Defines the attack row the card lies in.

5.62.1 Detailed Description

A class representing a card on the game board.
Is a persistent Entity and is therefore saved in a database table named `units`.
Definition at line 36 of file UnitEntity.java.

5.62.2 Constructor & Destructor Documentation

5.62.2.1 UnitEntity()

```
UnitEntity (
            boolean epic,
            @IntRange(from=0) int damage,
            @NonNull Ability ability,
            @IntRange(from=1) @Nullable Integer squad,
            @NonNull RowType row )  [package]
```

Constructor of a UnitEntity.
#id may not be set here since the value is generated automatically.

Parameters

epic	Boolean representing whether card is #epic.
damage	Non-negative value representing the #damage of the card.
ability	Ability representing the #ability of the card.
squad	Integer representing the #squad of a card that has the Ability::BINDING #ability.
row	RowType representing the combat type of the card.

Exceptions

IllegalArgumentException	When damage is less than zero or if ability is Ability::BINDING and squad is null or less than one or if ability is not Ability::BINDING and squad is not null.

Definition at line 105 of file UnitEntity.java.
References Ability.BINDING.

5.62.3 Member Function Documentation

5.62.3.1 calculateDamage()

```
int calculateDamage (
            @NonNull DamageCalculator calculator )
```

Calculates the damage of this unit when (de-)buffed.
Returns #damage if #epic is true. Otherwise, the damage is calculated through the given DamageCalculator, which follows the visitor pattern.

Parameters

calculator	DamageCalculator visitor used for damage calculation.

Returns

An Integer representing the units (de-)buffed damage.

See also

#getDamage()

Definition at line 132 of file UnitEntity.java.

5.62.3.2 collectionToString()

```
static String collectionToString (
             @NonNull Context context,
             @NonNull Collection< UnitEntity > units ) [static]
```

Creates a String containing the descriptions of all units in the given collection, separated by commas.

If n units have the same description, instead of printing the same description n times, "n×" is printed in front of the description. Unit descriptions are created using #toString(Context). Ordering of units in the given Collection is not guaranteed to be kept.

Parameters

context	Context used to acquire String resources.
units	Collection of UnitEntity objects that should be in the created String.

Returns

A String containing the description of all units.

Exceptions

IllegalArgumentException	When units collection is empty.

See also

#toString(Context)

Definition at line 166 of file UnitEntity.java.

5.62.3.3 getAbility()

```
Ability getAbility ( )
```

Getter for #ability.

Returns

Ability representing the units' ability.

Definition at line 315 of file UnitEntity.java.

5.62.3.4 getDamage()

```
int getDamage ( )
```

Getter for #damage.

Returns

Integer representing the card's base-damage.

See also

#calculateDamage(DamageCalculator)

Definition at line 291 of file UnitEntity.java.

5.62.3.5 getId()

```
int getId ( )
```

Getter for #id.

Returns

Integer representing the units' id.

Definition at line 252 of file UnitEntity.java.
Referenced by UnitRepository.getUnitsFlowable().

5.62.3.6 getRow()

```
RowType getRow ( )
```

Getter for #row.

Returns

RowType representing the units combat row.

Definition at line 362 of file UnitEntity.java.

5.62.3.7 getSquad()

```
Integer getSquad ( )
```

Getter for #squad.

Returns

Integer representing the units' squad if #ability is Ability::BINDING or `null`.

Definition at line 335 of file UnitEntity.java.

5.62.3.8 isBuffed()

```
Color isBuffed (
            @NonNull DamageCalculator calculator )
```

Returns whether this unit is (de-)buffed.
Returns Color::DEFAULT if #epic is true. Otherwise, whether the unit is (de-)buffed is calculated through the given DamageCalculator, which follows the visitor pattern.

Parameters

calculator	DamageCalculator visitor used for damage calculation.

Returns

A Color representing the units (de-)buffed damage.

See also

#getDamage()

Definition at line 147 of file UnitEntity.java.

5.62.3.9 isEpic()

```
boolean isEpic ( )
```

Getter for #epic.

Returns

Boolean representing whether the card is epic.

Definition at line 271 of file UnitEntity.java.

5.62.3.10 setAbility()

```
void setAbility (
            @NonNull Ability ability )  [package]
```

Setter for #ability.
Only used by Room extension.

Parameters

ability	Ability representing the units' ability.

Definition at line 325 of file UnitEntity.java.

5.62.3.11 setDamage()

```
void setDamage (
            @IntRange(from=0) int damage )  [package]
```

Setter for #damage.
Only used by Room extension.

Parameters

damage	Integer representing the card's base-damage.

Exceptions

IllegalArgumentException	When damage is less than zero.

Definition at line 302 of file UnitEntity.java.

5.62.3.12 setEpic()

```
void setEpic (
            boolean epic )  [package]
```

Setter for #epic.
Only used by Room extension.

Parameters

epic	Boolean representing whether the card is epic.

Definition at line 281 of file UnitEntity.java.

5.62.3.13 setId()

```
void setId (
            int id )  [package]
```

Setter for #id.
Only used by Room extension.

Parameters

id	Integer representing the units' id.

Definition at line 262 of file UnitEntity.java.

5.62.3.14 setRow()

```
void setRow (
            @NonNull RowType row )  [package]
```

Setter for #row.
Only used by Room extension.

Parameters

row	RowType representing the units combat row.

Definition at line 372 of file UnitEntity.java.

5.62.3.15 setSquad()

```
void setSquad (
            @IntRange(from=1) @Nullable Integer squad )  [package]
```

Setter for #squad.
Only used by Room extension.

Parameters

squad	Integer representing the units' squad if #ability is Ability::BINDING or `null`.

Exceptions

IllegalArgumentException	When #ability is Ability::BINDING and squad is null or less than 1 or if #ability is not Ability::BINDING and squad is not null.

Definition at line 346 of file UnitEntity.java.
References Ability.BINDING.

5.62.3.16 toString()

```
String toString (
            @NonNull Context context )
```

Returns a string representation of this unit.
The representation contains information on each field of this class, i.e., #row, #epic, #damage, #ability, and #squad.

Parameters

context	Context used to acquire String resources.

Returns

A string representing the unit.

Definition at line 210 of file UnitEntity.java.

5.62.4 Member Data Documentation

5.62.4.1 ability

```
Ability ability  [private]
```

Defines the Ability of the represented card.
Is set to Ability::NONE by default. Must not be `null`.
Definition at line 75 of file UnitEntity.java.

5.62.4.2 damage

```
int damage [private]
```

Defines the base-damage of the represented card.
Must be non-negative.
Definition at line 55 of file UnitEntity.java.

5.62.4.3 epic

```
boolean epic [private]
```

Defines whether the represented card is epic.
Is set to `false` by default.
Definition at line 49 of file UnitEntity.java.

5.62.4.4 EPIC_DAMAGE_VALUES

```
final Integer [] EPIC_DAMAGE_VALUES = new Integer[]{0, 7, 8, 10, 11, 15} [static]
```

Integer array containing every damage value an epic unit may have.
Definition at line 61 of file UnitEntity.java.
Referenced by UnitRepository.insertUnit().

5.62.4.5 id

```
int id [private]
```

Defines the primary key of the represented unit.
Is generated automatically on insert.
Definition at line 42 of file UnitEntity.java.

5.62.4.6 NON_EPIC_DAMAGE_VALUES_UPPER_BOUND

```
final int NON_EPIC_DAMAGE_VALUES_UPPER_BOUND = 20 [static]
```

Integer defining what the maximum damage of a non-epic UnitEntity can be.
Definition at line 66 of file UnitEntity.java.
Referenced by UnitRepository.insertUnit().

5.62.4.7 row

```
RowType row [private]
```

Defines the attack row the card lies in.
Must not be `null`.
Definition at line 92 of file UnitEntity.java.

5.62.4.8 squad

```
Integer squad [private]
```

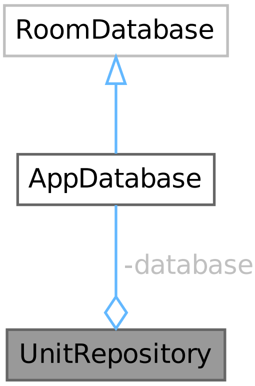
Defines the squad the unit belongs to if #ability is Ability::BINDING.
If #ability is anything else, this value is `null`.
Definition at line 84 of file UnitEntity.java.
The documentation for this class was generated from the following file:

- gwent/data/UnitEntity.java

5.63 UnitRepository Class Reference

A facade class managing public access to the data layer.

Collaboration diagram for UnitRepository:



Public Member Functions

- Completable reset ()

  Resets the board by removing all units and resetting row status.
- Completable reset (@Nullable UnitEntity keptUnit)

  Resets the board by removing all units but the given one and resetting row status.
- Completable insertUnit (boolean epic, @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability, @IntRange(from=1) @Nullable Integer squad, @NonNull RowType row, @IntRange(from=0) int number)

  Adds a number of units with the given stats to the given row.
- Completable switchWeather (@NonNull RowType row)

  Flips RowEntity::weather of the given attack row.
- Single< Boolean > isWeather (@NonNull RowType row)

  Returns the value of RowEntity::weather for the given attack row.
- Flowable< Boolean > isWeatherFlowable (@NonNull RowType row)

  Returns a Flowable emitting the latest value of RowEntity::weather for the given attack row.
- Completable clearWeather ()

  Sets RowEntity::weather to `false` for all attack rows.
- Completable switchHorn (@NonNull RowType row)

  Flips RowEntity::horn of the given attack row.
- Single< Boolean > isHorn (@NonNull RowType row)

  Returns the value of RowEntity::horn for the given attack row.
- Flowable< Boolean > isHornFlowable (@NonNull RowType row)

  Returns a Flowable emitting the values of RowEntity::horn for the given attack row.
- Completable delete (@NonNull Collection< UnitEntity > units)

  Removes the given units from the game.
- Completable delete (int id)

  Removes the unit with the given id from the game.
- Completable copy (int id)

  Copies the unit with the given id.
- Single< Integer > countUnits (@NonNull RowType row)

  Counts the units in the given attack row.

- Single< Integer > countUnits ()

  Counts the units in all attack rows.
- Single< UnitEntity > getUnit (int id)

  Returns the unit with the given id.
- Single< List< UnitEntity > > getUnits (@NonNull RowType row)

  Returns the units in the given attack row.
- Flowable< List< UnitEntity > > getUnitsFlowable (@NonNull RowType row)

  Returns a Flowable emitting the units in the given attack row.
- Single< List< UnitEntity > > getUnits ()

  Returns the units in the given attack row.
- Flowable< Boolean > hasNonEpicUnitsFlowable ()

  Returns a Flowable emitting Boolean values defining whether there are units in the game that are not epic.

Static Public Member Functions

- static Single< UnitRepository > getRepository (@NonNull AppDatabase database)

  Factory method for the UnitRepository class.

Private Member Functions

- UnitRepository (@NonNull AppDatabase database)

  Constructor of a UnitRepository.
- Completable initializeRows ()

  Adds one attack row for each RowType.
- Completable insertUnit (boolean epic, @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability, @IntRange(from=1) @Nullable Integer squad, @NonNull RowType row)

  Adds a unit with the given stats to the given row.

Private Attributes

- final AppDatabase database

  Defines the AppDatabase that is used as a data source by this repository.

5.63.1 Detailed Description

A facade class managing public access to the data layer.
The contained functions mostly redirect requests to package-private DAO methods in RowDao and UnitDao. Some functions implement slightly more complex behavior by chaining multiple DAO calls, e.g., #reset().
Definition at line 25 of file UnitRepository.java.

5.63.2 Constructor & Destructor Documentation

5.63.2.1 UnitRepository()

```
UnitRepository (
                @NonNull AppDatabase database )  [private]
```

Constructor of a UnitRepository.
Depends on the given AppDatabase as a data source. Should only be called by #getRepository().

Parameters

database	AppDatabase that is injected for the repository.

See also

#getRepository(AppDatabase)

Definition at line 41 of file UnitRepository.java.
References UnitRepository.database.

5.63.3 Member Function Documentation

5.63.3.1 clearWeather()

```
Completable clearWeather ( )
```

Sets RowEntity::weather to `false` for all attack rows.

Returns

A Completable tracking operation status.

Definition at line 208 of file UnitRepository.java.
References RowDao.clearWeather(), UnitRepository.database, and AppDatabase.rows().

5.63.3.2 copy()

```
Completable copy (
             int id )
```

Copies the unit with the given id.

Parameters

id	Integer representing the unit that should be copied.

Returns

A Completable tracking operation status.

Definition at line 277 of file UnitRepository.java.
References UnitRepository.getUnit(), and UnitRepository.insertUnit().

5.63.3.3 countUnits() [1/2]

```
Single< Integer > countUnits ( )
```

Counts the units in all attack rows.

Returns

A Single tracking operation status and returning the value.

See also

#countUnits(RowType)

Definition at line 302 of file UnitRepository.java.
References UnitDao.countUnits(), UnitRepository.database, and AppDatabase.units().

5.63.3.4 countUnits() [2/2]

```
Single< Integer > countUnits (
             @NonNull RowType row )
```

Counts the units in the given attack row.

Parameters

row	RowEntity::id where the units are counted.

Returns

A Single tracking operation status and returning the value.

See also

#countUnits()

Definition at line 291 of file UnitRepository.java.
References UnitDao.countUnits(), UnitRepository.database, and AppDatabase.units().

5.63.3.5 delete() [1/2]

```
Completable delete (
            @NonNull Collection< UnitEntity > units )
```

Removes the given units from the game.

Parameters

units	List of units to be removed.

Returns

A Completable tracking operation status.

Definition at line 256 of file UnitRepository.java.
References UnitRepository.database, UnitDao.deleteUnits(), and AppDatabase.units().

5.63.3.6 delete() [2/2]

```
Completable delete (
            int id )
```

Removes the unit with the given id from the game.

Parameters

id	Integer representing the unit that should be deleted.

Returns

A Completable tracking operation status.

Definition at line 267 of file UnitRepository.java.
References UnitRepository.database, UnitDao.deleteUnit(), and AppDatabase.units().

5.63.3.7 getRepository()

```
static Single< UnitRepository > getRepository (
            @NonNull AppDatabase database )  [static]
```

Factory method for the UnitRepository class.
Creates a new UnitRepository managing the given AppDatabase. Also initializes one RowEntity per RowType using #initializeRows().

See also

#initializeRows()

Parameters

database	AppDatabase managed and initialized by the returned UnitRepository.

Returns

A Single emitting the created UnitRepository when initialization is finished.

Definition at line 54 of file UnitRepository.java.
References UnitRepository.database, and UnitRepository.initializeRows().

5.63.3.8 getUnit()

```
Single< UnitEntity > getUnit (
                int id )
```

Returns the unit with the given id.

Parameters

id	Integer representing the queried unit.

Returns

A Single tracking operation status and returning the value.

Definition at line 313 of file UnitRepository.java.
References UnitRepository.database, UnitDao.getUnit(), and AppDatabase.units().
Referenced by UnitRepository.copy().

5.63.3.9 getUnits() [1/2]

```
Single< List< UnitEntity > > getUnits ( )
```

Returns the units in the given attack row.

Returns

A Single tracking operation status and returning the value.

See also

#getUnits(RowType)

Definition at line 361 of file UnitRepository.java.
References UnitRepository.database, UnitDao.getUnits(), and AppDatabase.units().

5.63.3.10 getUnits() [2/2]

```
Single< List< UnitEntity > > getUnits (
                @NonNull RowType row )
```

Returns the units in the given attack row.

See also

#getUnitsFlowable(RowType)

Parameters

row	RowEntity::id where the units have been placed.

Returns

A Single tracking operation status and returning the value.

See also

#getUnits()

Definition at line 325 of file UnitRepository.java.
References UnitRepository.database, UnitDao.getUnits(), and AppDatabase.units().

5.63.3.11 getUnitsFlowable()

```
Flowable< List< UnitEntity > > getUnitsFlowable (
            @NonNull RowType row )
```
Returns a Flowable emitting the units in the given attack row.

Parameters

row	RowEntity::id where the units have been placed.

Returns

A Flowable emitting the values.

See also

#getUnits(RowType)

#getUnits()

Definition at line 338 of file UnitRepository.java.
References UnitRepository.database, UnitEntity.getId(), UnitDao.getUnitsFlowable(), and AppDatabase.units().

5.63.3.12 hasNonEpicUnitsFlowable()

```
Flowable< Boolean > hasNonEpicUnitsFlowable ( )
```
Returns a Flowable emitting Boolean values defining whether there are units in the game that are not epic.

Returns

A Flowable emitting the values.

Definition at line 370 of file UnitRepository.java.
References UnitRepository.database, UnitDao.hasNonEpicUnitsFlowable(), and AppDatabase.units().

5.63.3.13 initializeRows()

```
Completable initializeRows ( ) [private]
```
Adds one attack row for each RowType.
If an attack row already exists, it is not inserted again.

Returns

A Completable tracking operation status.

Definition at line 69 of file UnitRepository.java.
References UnitRepository.database, RowDao.insertRow(), and AppDatabase.rows().
Referenced by UnitRepository.getRepository(), and UnitRepository.reset().

5.63.3.14 insertUnit() [1/2]

```
Completable insertUnit (
            boolean epic,
            @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage,
            @NonNull Ability ability,
            @IntRange(from=1) @Nullable Integer squad,
            @NonNull RowType row ) [private]
```
Adds a unit with the given stats to the given row.

Parameters

epic	Boolean representing whether card is #epic.
damage	Non-negative value representing the #damage of the card.
ability	Ability representing the #ability of the card.
squad	Integer representing the #squad of a card that has the Ability::BINDING #ability.
row	RowType representing the combat type of the card.

Returns

A Completable tracking operation status.

Exceptions

IllegalArgumentException	When damage is less than zero or if ability is Ability::BINDING and squad is null or less than zero or if ability is not Ability::BINDING and squad is not null.

Definition at line 119 of file UnitRepository.java.
References Ability.BINDING, UnitRepository.database, UnitEntity.EPIC_DAMAGE_VALUES, UnitDao.insertUnit(), UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND, and AppDatabase.units().
Referenced by UnitRepository.copy(), and UnitRepository.insertUnit().

5.63.3.15 insertUnit() [2/2]

```
Completable insertUnit (
            boolean epic,
            @IntRange(from=0, to=UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage,
            @NonNull Ability ability,
            @IntRange(from=1) @Nullable Integer squad,
            @NonNull RowType row,
            @IntRange(from=0) int number )
```

Adds a number of units with the given stats to the given row.
Essentially calls #insertUnit(boolean, int, Ability, Integer, RowType) multiple times.

Parameters

epic	Boolean representing whether card is #epic.
damage	Non-negative value representing the #damage of the card.
ability	Ability representing the #ability of the card.
squad	Integer representing the #squad of a card that has the Ability::BINDING #ability.
row	RowType representing the combat type of the card.
number	Integer representing the number of units to be added.

Returns

A Completable tracking operation status.

See also

#insertUnit(boolean, int, Ability, Integer, RowType)

Exceptions

IllegalArgumentException	When damage is less than zero or if ability is Ability::BINDING and squad is null or less than zero or if ability is not Ability::BINDING and squad is not null.

Definition at line 153 of file UnitRepository.java.
References UnitRepository.insertUnit().

5.63.3.16 isHorn()

```
Single< Boolean > isHorn (
              @NonNull RowType row )
```

Returns the value of RowEntity::horn for the given attack row.

See also

#isHornFlowable(RowType)

Parameters

row	RowEntity::id where the horn status is queried.

Returns

A Single tracking operation status and returning the value.

Definition at line 229 of file UnitRepository.java.
References UnitRepository.database, RowDao.isHorn(), and AppDatabase.rows().

5.63.3.17 isHornFlowable()

```
Flowable< Boolean > isHornFlowable (
              @NonNull RowType row )
```

Returns a Flowable emitting the values of RowEntity::horn for the given attack row.

Parameters

row	RowEntity::id where the horn status is queried.

Returns

A Flowable emitting the values.

See also

#isHorn(RowType)

Definition at line 241 of file UnitRepository.java.
References UnitRepository.database, RowDao.isHornFlowable(), and AppDatabase.rows().

5.63.3.18 isWeather()

```
Single< Boolean > isWeather (
              @NonNull RowType row )
```

Returns the value of RowEntity::weather for the given attack row.

See also

#isWeatherFlowable(RowType)

Parameters

row	RowEntity::id where the weather is queried.

Returns

A Single tracking operation status and returning the value.

Definition at line 181 of file UnitRepository.java.
References UnitRepository.database, RowDao.isWeather(), and AppDatabase.rows().

5.63.3.19 isWeatherFlowable()

```
Flowable< Boolean > isWeatherFlowable (
            @NonNull RowType row )
```

Returns a Flowable emitting the latest value of RowEntity::weather for the given attack row.

Parameters

row	RowEntity::id where the weather is queried.

Returns

A Flowable emitting the values.

See also

#isWeather(RowType)

Definition at line 193 of file UnitRepository.java.
References UnitRepository.database, RowDao.isWeatherFlowable(), and AppDatabase.rows().

5.63.3.20 reset() [1/2]

```
Completable reset ( )
```

Resets the board by removing all units and resetting row status.
Resetting row status is equivalent to removing the old rows and calling #initializeRows(). Method is a wrapper for #reset(UnitEntity).

Returns

A Completable tracking operation status.

See also

#reset(UnitEntity)

Definition at line 86 of file UnitRepository.java.
References UnitRepository.reset().
Referenced by UnitRepository.reset().

5.63.3.21 reset() [2/2]

```
Completable reset (
            @Nullable UnitEntity keptUnit )
```

Resets the board by removing all units but the given one and resetting row status.
Resetting row status is equivalent to removing the old rows and calling #initializeRows().

Parameters

keptUnit	UnitEntity that should be kept.

Returns

A Completable tracking operation status.

See also

#initializeRows()

Definition at line 99 of file UnitRepository.java.
References RowDao.clearRows(), UnitRepository.database, UnitRepository.initializeRows(), UnitDao.insertUnit(), AppDatabase.rows(), and AppDatabase.units().

5.63.3.22 switchHorn()

```
Completable switchHorn (
            @NonNull RowType row )
```

Flips RowEntity::horn of the given attack row.

Parameters

row	RowEntity::id where the horn status should be updated.

Returns

A Completable tracking operation status.

Definition at line 218 of file UnitRepository.java.
References UnitRepository.database, AppDatabase.rows(), and RowDao.updateHorn().

5.63.3.23 switchWeather()

```
Completable switchWeather (
            @NonNull RowType row )
```

Flips RowEntity::weather of the given attack row.

Parameters

row	RowEntity::id where the weather should be updated.

Returns

A Completable tracking operation status.

Definition at line 170 of file UnitRepository.java.
References UnitRepository.database, AppDatabase.rows(), and RowDao.updateWeather().

5.63.4 Member Data Documentation

5.63.4.1 database

```
final AppDatabase database [private]
```

Defines the AppDatabase that is used as a data source by this repository.
Is provided by dependency injection in #UnitRepository().
Definition at line 31 of file UnitRepository.java.

Referenced by UnitRepository.UnitRepository(), UnitRepository.clearWeather(), UnitRepository.countUnits(), UnitRepository.countUnits(), UnitRepository.delete(), UnitRepository.delete(), UnitRepository.getRepository(), UnitRepository.getUnit(), UnitRepository.getUnits(), UnitRepository.getUnits(), UnitRepository.getUnitsFlowable(), UnitRepository.hasNonEpicUnitsFlowable(), UnitRepository.initializeRows(), UnitRepository.insertUnit(), UnitRepository.isHorn(), UnitRepository.isHornFlowable(), UnitRepository.isWeather(), UnitRepository.isWeatherFlowable(), UnitRepository.reset(), UnitRepository.switchHorn(), and UnitRepository.switchWeather().

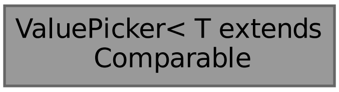
The documentation for this class was generated from the following file:

- gwent/data/UnitRepository.java

5.64 ValuePicker< T extends Comparable Class Template Reference

A generic abstract wrapper class around NumberPicker used to select values of arbitrary types extending Comparable.

Collaboration diagram for ValuePicker< T extends Comparable:



5.64.1 Detailed Description

A generic abstract wrapper class around NumberPicker used to select values of arbitrary types extending Comparable.

Must be subclassed to provide the displayed String values using #getDisplayString(). The default case, which is implemented in StringValuePicker, is that displayIntegers maps the selectableValues to a String resource id.

See also

NumberPicker

StringValuePicker

Definition at line 29 of file ValuePicker.java.

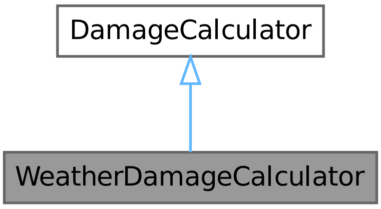
The documentation for this class was generated from the following file:

- gwent/ui/dialogs/addcard/ValuePicker.java

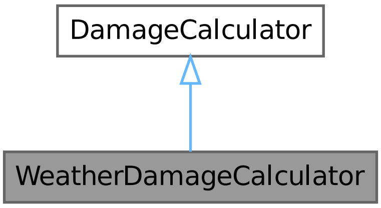
5.65 WeatherDamageCalculator Class Reference

A DamageCalculator class responsible for calculating the weather debuff if necessary, i.e., if #weather is true.

Inheritance diagram for WeatherDamageCalculator:



Collaboration diagram for WeatherDamageCalculator:



Public Member Functions

- int calculateDamage (int id, @IntRange(from=0) int damage)

  Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
- Color isBuffed (int id)

  Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.

Package Functions

- WeatherDamageCalculator (boolean weather)

  Constructor of a WeatherDamageCalculator.

Private Attributes

- final boolean weather

  Defines whether the weather debuff should be applied when calling #calculateDamage().

5.65.1 Detailed Description

A DamageCalculator class responsible for calculating the weather debuff if necessary, i.e., if #weather is true.
Acts as the ConcreteComponent in the implemented decorator pattern and is therefore always at the end of the decorator chain.
Definition at line 14 of file WeatherDamageCalculator.java.

5.65.2 Constructor & Destructor Documentation

5.65.2.1 WeatherDamageCalculator()

```
WeatherDamageCalculator (
            boolean weather )  [package]
```

Constructor of a WeatherDamageCalculator.
Should only be called by DamageCalculatorBuilder.

Parameters

weather	Boolean defining whether the weather debuff should be applied.

See also

DamageCalculatorBuilder

Definition at line 27 of file WeatherDamageCalculator.java.
References WeatherDamageCalculator.weather.

5.65.3 Member Function Documentation

5.65.3.1 calculateDamage()

```
int calculateDamage (
            int id,
            @IntRange(from=0) int damage )
```

Calculates the (de-)buffed damage of unit with the given id and the given base-damage.
Returns the given damage if #weather is false or 1.

Parameters

id	Integer representing the UnitEntity::id of the unit whose (de-)buff damage is calculated.
damage	Integer representing the base-damage of the unit whose (de-)buff damage is calculated.

Returns

Integer representing the (de-)buffed damage of the unit.

Exceptions

IllegalArgumentException	When damage is negative.

Implements DamageCalculator.
Definition at line 41 of file WeatherDamageCalculator.java.
References WeatherDamageCalculator.weather.

5.65.3.2 isBuffed()

```
Color isBuffed (
            int id )
```

Calculates whether the unit with the given id is shown as Color::BUFFED, Color::DEBUFFED, or Color::DEFAULT.
Returns Color::DEBUFFED if #weather is true or else Color::DEFAULT.

Parameters

id	Integer representing the UnitEntity::id of the unit buff status is calculated.

Returns

Color representing whether the unit is buffed, de-buffed or not affected.

See also

Color

Implements DamageCalculator.
Definition at line 61 of file WeatherDamageCalculator.java.
References WeatherDamageCalculator.weather.

5.65.4 Member Data Documentation

5.65.4.1 weather

```
final boolean weather  [private]
```

Defines whether the weather debuff should be applied when calling #calculateDamage().
Definition at line 18 of file WeatherDamageCalculator.java.
Referenced by WeatherDamageCalculator.WeatherDamageCalculator(), WeatherDamageCalculator.calculateDamage(), and WeatherDamageCalculator.isBuffed().
The documentation for this class was generated from the following file:

- gwent/domain/damage/WeatherDamageCalculator.java

Chapter 6

File Documentation

6.1 /home/runner/work/Gwent/Gwent/app/src/main/AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission
        android:name="com.google.android.gms.permission.AD_ID"
        tools:node="remove" />
    <application
        android:name=".GwentApplication"
        android:appCategory="game"
        android:description="@string/app_description"
        android:icon="@mipmap/icon_launcher"
        android:label="@string/app_name"
        android:resizeableActivity="false"
        android:roundIcon="@mipmap/icon_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/ScoiataelTheme"
        tools:targetApi="o">
        <activity android:name=".ui.introduction.IntroductionActivity" />
        <activity
            android:name=".ui.main.MainActivity"
            android:exported="true"
            android:screenOrientation="userLandscape">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ui.settings.SettingsActivity"
            android:exported="true"
            android:label="@string/settings_title">
            <intent-filter>
                <action android:name="android.intent.action.APPLICATION_PREFERENCES" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ui.settings.RuleActivity"
            android:label="@string/preference_rules_header" />
    </application>
</manifest>
```

6.2 Ability.java

```
package com.peternaggschga.gwent.data;

public enum Ability {
    NONE,

    HORN,

    REVENGE,

    BINDING,

    MORAL_BOOST
}
```

6.3 AppDatabase.java

```java
package com.peternaggschga.gwent.data;

import androidx.room.Database;
import androidx.room.RoomDatabase;

@Database(entities = {UnitEntity.class, RowEntity.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
    abstract UnitDao units();

    abstract RowDao rows();
}
```

6.4 RowDao.java

```java
package com.peternaggschga.gwent.data;

import androidx.annotation.NonNull;
import androidx.room.Dao;
import androidx.room.Insert;
import androidx.room.OnConflictStrategy;
import androidx.room.Query;

import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.Flowable;
import io.reactivex.rxjava3.core.Single;

@Dao
@SuppressWarnings("NullableProblems")
interface RowDao {
    @Insert(onConflict = OnConflictStrategy.IGNORE)
    Completable insertRow(@NonNull RowEntity row);

    @Query("DELETE FROM rows")
    Completable clearRows();

    @Query("UPDATE rows SET weather = NOT weather WHERE id = :row")
    Completable updateWeather(@NonNull RowType row);

    @Query("UPDATE rows SET weather = 0")
    Completable clearWeather();

    @Query("UPDATE rows SET horn = NOT horn WHERE id = :row")
    Completable updateHorn(@NonNull RowType row);

    @Query("SELECT weather FROM rows WHERE id = :row")
    Single<Boolean> isWeather(@NonNull RowType row);

    @Query("SELECT weather FROM rows WHERE id = :row")
    Flowable<Boolean> isWeatherFlowable(@NonNull RowType row);

    @Query("SELECT horn FROM rows WHERE id = :row")
    Single<Boolean> isHorn(@NonNull RowType row);

    @Query("SELECT horn FROM rows WHERE id = :row")
    Flowable<Boolean> isHornFlowable(@NonNull RowType row);
}
```

6.5 RowEntity.java

```java
package com.peternaggschga.gwent.data;

import androidx.annotation.NonNull;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;

@Entity(tableName = "rows")
@SuppressWarnings("unused")
class RowEntity {
    @PrimaryKey
    @NonNull
    private final RowType id;

    @ColumnInfo(defaultValue = "false")
    private boolean weather;

    @ColumnInfo(defaultValue = "false")
    private boolean horn;

    RowEntity(@NonNull RowType id) {
        this.id = id;
    }

    @NonNull
    RowType getId() {
        return id;
    }

    boolean isWeather() {
        return weather;
    }

    void setWeather(boolean weather) {
        this.weather = weather;
    }

    boolean isHorn() {
        return horn;
    }

    void setHorn(boolean horn) {
        this.horn = horn;
    }
}
```

6.6 RowType.java

```java
package com.peternaggschga.gwent.data;

public enum RowType {
    MELEE,

    RANGE,

    SIEGE
}
```

6.7 UnitDao.java

```java
package com.peternaggschga.gwent.data;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.room.Dao;
import androidx.room.Delete;
import androidx.room.Insert;
import androidx.room.OnConflictStrategy;
import androidx.room.Query;

import java.util.Collection;
import java.util.List;

import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.Flowable;
import io.reactivex.rxjava3.core.Single;

@Dao
@SuppressWarnings("NullableProblems")
interface UnitDao {
    @Insert(onConflict = OnConflictStrategy.IGNORE)
    Completable insertUnit(@NonNull UnitEntity unit);

    @Query("INSERT INTO units (epic, damage, ability, squad, `row`) VALUES (:epic, :damage, :ability, :squad, :row)")
    Completable insertUnit(boolean epic, @IntRange(from = 0, to = UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability, @IntRange(from = 1) @Nullable Integer squad, @NonNull RowType row);

    @Query("DELETE FROM units WHERE `id` = :id")
    Completable deleteUnit(int id);

    @Delete
    Completable deleteUnits(@NonNull Collection<UnitEntity> units);

    @Query("SELECT * FROM units WHERE `id` = :id")
    Single<UnitEntity> getUnit(int id);

    @Query("SELECT * FROM units WHERE `row` = :row")
    Single<List<UnitEntity>> getUnits(@NonNull RowType row);

    @Query("SELECT * FROM units WHERE `row` = :row")
    Flowable<List<UnitEntity» getUnitsFlowable(@NonNull RowType row);

    @Query("SELECT * FROM units")
    Single<List<UnitEntity» getUnits();

    @Query("SELECT COUNT(*) FROM units WHERE `row` = :row")
    Single<Integer> countUnits(@NonNull RowType row);

    @Query("SELECT COUNT(*) FROM units")
    Single<Integer> countUnits();

    @Query("SELECT COUNT(*) > 0 FROM units WHERE epic = 0")
    Flowable<Boolean> hasNonEpicUnitsFlowable();
}
```

6.8 UnitEntity.java

```
package com.peternaggschga.gwent.data;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color;
import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.DEFAULT;

import android.content.Context;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.ForeignKey;
import androidx.room.PrimaryKey;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;

@Entity(tableName = "units", foreignKeys = {
        @ForeignKey(entity = RowEntity.class,
                parentColumns = "id",
                childColumns = "row",
                onDelete = ForeignKey.CASCADE)})
@SuppressWarnings("unused")
public class UnitEntity {
    @PrimaryKey(autoGenerate = true)
    private int id;

    @ColumnInfo(defaultValue = "false")
    private boolean epic;

    private int damage;

    @NonNull
    public static final Integer[] EPIC_DAMAGE_VALUES = new Integer[]{0, 7, 8, 10, 11, 15};

    public static final int NON_EPIC_DAMAGE_VALUES_UPPER_BOUND = 20;

    @ColumnInfo(defaultValue = "NONE")
    @NonNull
    private Ability ability;

    @ColumnInfo(defaultValue = "NULL")
    @IntRange(from = 1)
    @Nullable
    private Integer squad;

    @ColumnInfo(index = true)
    @NonNull
    private RowType row;

    UnitEntity(boolean epic, @IntRange(from = 0) int damage, @NonNull Ability ability, @IntRange(from = 1) @Nullable Integer squad, @NonNull RowType row) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0 but is " + damage + ".");
        }
        if (ability != Ability.BINDING && squad != null) {
            throw new IllegalArgumentException("Squad must be null or ability must be BINDING but squad is " + squad + " and ability is " + ability + ".");
        }
```

```java
        if (ability == Ability.BINDING && (squad == null || squad < 1)) {
            throw new IllegalArgumentException("Squad must not be null or less than one but squad is " + squad + ".");
        }
        this.epic = epic;
        this.damage = damage;
        this.ability = ability;
        this.squad = squad;
        this.row = row;
    }

    public int calculateDamage(@NonNull DamageCalculator calculator) {
        return epic ? damage : calculator.calculateDamage(id, damage);
    }

    @NonNull
    public Color isBuffed(@NonNull DamageCalculator calculator) {
        return epic ? DEFAULT : calculator.isBuffed(id);
    }

    @NonNull
    public static String collectionToString(@NonNull Context context, @NonNull Collection<UnitEntity> units) {
        if (units.isEmpty()) {
            throw new IllegalArgumentException("Units collection must not be empty.");
        }

        Map<String, Integer> descriptionStrings = new HashMap<>(units.size());
        for (UnitEntity unit : units) {
            String description = unit.toString(context);
            descriptionStrings.merge(description, 1, Integer::sum);
        }

        List<String> descriptions = new ArrayList<>(descriptionStrings.size());
        if (descriptionStrings.size() < units.size()) {
            descriptionStrings.forEach((key, value) ->
                    descriptions.add(context.getString(R.string.unit_toString_multiplicity, value, key)));
        } else {
            descriptions.addAll(descriptionStrings.keySet());
        }

        Iterator<String> descriptionIterator = descriptions.iterator();
        String[] result = {descriptionIterator.next()};
        if (descriptionIterator.hasNext()) {
            result[0] = context.getString(R.string.unit_collection_toString_accumulation_word,
                            descriptionIterator.next(),
                            result[0])
                    .trim();
        }
        descriptionIterator.forEachRemaining(description ->
                result[0] = context.getString(R.string.unit_collection_toString_accumulation_symbol,
                                description,
                                result[0])
                        .trim());
        return result[0];
    }

    @NonNull
    public String toString(@NonNull Context context) {
        String row;
        switch (getRow()) {
            case RANGE:
                row = context.getString(R.string.unit_toString_range);
                break;
            case SIEGE:
                row = context.getString(R.string.unit_toString_siege);
                break;
            case MELEE:
            default:
                row = context.getString(R.string.unit_toString_melee);
        }
        String epic = isEpic() ? context.getString(R.string.unit_toString_epic) : context.getString(R.string.unit_toString_unit);
        String ability;
        String squad = "";
        switch (getAbility()) {
            case HORN:
                ability = context.getString(R.string.add_picker_ability_horn);
                break;
            case BINDING:
                ability = context.getString(R.string.add_picker_ability_binding);
                squad = context.getString(R.string.unit_toString_squad, getSquad());
                break;
            case MORAL_BOOST:
                ability = context.getString(R.string.add_picker_ability_moralBoost);
                break;
            case REVENGE:
                ability = context.getString(R.string.add_picker_ability_revenge);
                break;
            case NONE:
            default:
                ability = context.getString(R.string.unit_toString_ability_none);
        }
        return context.getString(R.string.unit_toString, row, epic, getDamage(), ability, squad).trim();
    }

    public int getId() {
        return id;
    }

    void setId(int id) {
        this.id = id;
    }

    public boolean isEpic() {
        return epic;
    }

    void setEpic(boolean epic) {
        this.epic = epic;
    }

    public int getDamage() {
        return damage;
    }

    void setDamage(@IntRange(from = 0) int damage) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must not be less than zero but is " + damage + ".");
        }
        this.damage = damage;
    }

    @NonNull
    public Ability getAbility() {
        return ability;
    }

    void setAbility(@NonNull Ability ability) {
        this.ability = ability;
    }

    @Nullable
    public Integer getSquad() {
        return squad;
    }

    void setSquad(@IntRange(from = 1) @Nullable Integer squad) {
        if (ability != Ability.BINDING && squad != null) {
            throw new IllegalArgumentException("Squad must be null or ability must be BINDING but squad is " + squad + " and ability is " + ability + ".");
        }
        if (ability == Ability.BINDING && (squad == null || squad < 1)) {
            throw new IllegalArgumentException("Squad must not be null or less than one but squad is " + squad + ".");
        }
        this.squad = squad;
    }

    @NonNull
    public RowType getRow() {
        return row;
    }

    void setRow(@NonNull RowType row) {
        this.row = row;
    }
}
```

6.9 UnitRepository.java

```
package com.peternaggschga.gwent.data;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import java.util.Arrays;
import java.util.Collection;
```

```java
import java.util.Iterator;
import java.util.List;
import java.util.concurrent.TimeUnit;

import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers;
import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.Flowable;
import io.reactivex.rxjava3.core.Single;
import io.reactivex.rxjava3.schedulers.Schedulers;

public class UnitRepository {
    @NonNull
    private final AppDatabase database;

    private UnitRepository(@NonNull AppDatabase database) {
        this.database = database;
    }

    @NonNull
    public static Single<UnitRepository> getRepository(@NonNull AppDatabase database) {
        UnitRepository repository = new UnitRepository(database);
        return repository.initializeRows()
                .andThen(Single.just(repository))
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    private Completable initializeRows() {
        Completable result = Completable.complete();
        for (RowType row : RowType.values()) {
            result = result.andThen(database.rows().insertRow(new RowEntity(row)));
        }
        return result;
    }

    @NonNull
    public Completable reset() {
        return reset(null);
    }

    @NonNull
    public Completable reset(@Nullable UnitEntity keptUnit) {
        Completable result = database.rows().clearRows().andThen(initializeRows());
        if (keptUnit != null) {
            result = result.andThen(database.units().insertUnit(keptUnit));
        }
        return result.subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    private Completable insertUnit(boolean epic, @IntRange(from = 0, to = UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability,
                                   @IntRange(from = 1) @Nullable Integer squad, @NonNull RowType row) {
        if (damage < 0 || damage > UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) {
            throw new IllegalArgumentException("Damage be between 0 and " + UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND + " but is " + damage + ".");
        }
        if (epic && Arrays.stream(UnitEntity.EPIC_DAMAGE_VALUES).noneMatch(integer -> integer == damage)) {
            throw new IllegalArgumentException("Damage of epic units must be in " + Arrays.toString(UnitEntity.EPIC_DAMAGE_VALUES) + " but is " + damage + ".");
        }
        if (ability != Ability.BINDING && squad != null) {
            throw new IllegalArgumentException("Squad must be null or ability must be BINDING but squad is " + squad + " and ability is " + ability + ".");
        }
        if (ability == Ability.BINDING && (squad == null || squad < 1)) {
            throw new IllegalArgumentException("Squad must not be null or less than one but squad is " + squad + ".");
        }
        return database.units().insertUnit(epic, damage, ability, squad, row)
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable insertUnit(boolean epic, @IntRange(from = 0, to = UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int damage, @NonNull Ability ability,
                                  @IntRange(from = 1) @Nullable Integer squad, @NonNull RowType row,
                                  @IntRange(from = 0) int number) {
        Completable result = Completable.complete();
        for (int i = 0; i < number; i++) {
            result = result.andThen(insertUnit(epic, damage, ability, squad, row));
        }
        return result.subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable switchWeather(@NonNull RowType row) {
        return database.rows().updateWeather(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<Boolean> isWeather(@NonNull RowType row) {
        return database.rows().isWeather(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Flowable<Boolean> isWeatherFlowable(@NonNull RowType row) {
        return database.rows()
                .isWeatherFlowable(row)
                .onBackpressureLatest()
                .distinctUntilChanged()
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable clearWeather() {
        return database.rows().clearWeather().subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable switchHorn(@NonNull RowType row) {
        return database.rows().updateHorn(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<Boolean> isHorn(@NonNull RowType row) {
        return database.rows().isHorn(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Flowable<Boolean> isHornFlowable(@NonNull RowType row) {
        return database.rows()
                .isHornFlowable(row)
                .onBackpressureLatest()
                .distinctUntilChanged()
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable delete(@NonNull Collection<UnitEntity> units) {
        return database.units().deleteUnits(units).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable delete(int id) {
        return database.units().deleteUnit(id).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Completable copy(int id) {
        return getUnit(id).flatMapCompletable(unit ->
                        insertUnit(unit.isEpic(), unit.getDamage(), unit.getAbility(), unit.getSquad(), unit.getRow()))
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<Integer> countUnits(@NonNull RowType row) {
        return database.units().countUnits(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<Integer> countUnits() {
        return database.units().countUnits().subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<UnitEntity> getUnit(int id) {
        return database.units().getUnit(id).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<List<UnitEntity» getUnits(@NonNull RowType row) {
        return database.units().getUnits(row).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Flowable<List<UnitEntity» getUnitsFlowable(@NonNull RowType row) {
        return database.units()
                .getUnitsFlowable(row)
                .onBackpressureLatest()
                .debounce(10, TimeUnit.MILLISECONDS)
                .distinctUntilChanged((list1, list2) -> {
                    if (list1.size() != list2.size()) {
                        return false;
                    }
                    Iterator<UnitEntity> list1Iterator = list1.iterator();
                    return list2.stream().map(UnitEntity::getId).allMatch(id -> id.equals(list1Iterator.next().getId()));
                })
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Single<List<UnitEntity» getUnits() {
        return database.units().getUnits().subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread());
    }

    @NonNull
    public Flowable<Boolean> hasNonEpicUnitsFlowable() {
        return database.units()
                .hasNonEpicUnitsFlowable()
                .onBackpressureLatest()
                .debounce(10, TimeUnit.MILLISECONDS)
                .distinctUntilChanged()
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());
    }
}
```

6.10 BurnDialogUseCase.java

```
package com.peternaggschga.gwent.domain.cases;

import android.content.Context;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;

import io.reactivex.rxjava3.core.Single;

public class BurnDialogUseCase {
    @NonNull
    private static Single<List<UnitEntity» getBurnUnits(@NonNull UnitRepository repository) {
        return repository.getUnits()
                .flatMap(units -> {
                    if (units.isEmpty()) {
                        return Single.just(units);
                    }

                    Single<Map<RowType, DamageCalculator» calculators = Single.just(new HashMap<>(RowType.values().length));
                    for (RowType row : RowType.values()) {
                        calculators =
calculators.zipWith(DamageCalculatorUseCase.getDamageCalculator(repository, row), (calculatorMap, damageCalculator) -> {
                            calculatorMap.put(row, damageCalculator);
                            return calculatorMap;
                        });
                    }

                    return calculators.map(damageCalculators -> {
                        List<UnitEntity> maxDamageUnits = new ArrayList<>(units.size());
                        final int[] maxDamage = {0};
                        units.stream()
                                .filter(unit -> !unit.isEpic())
                                .forEach(unit -> {
                                    int damage = unit.calculateDamage(Objects.requireNonNull(damageCalculators.get(unit.getRow())));
                                    if (damage > maxDamage[0]) {
                                        maxDamage[0] = damage;
                                        maxDamageUnits.clear();
                                        maxDamageUnits.add(unit);
                                    } else if (damage == maxDamage[0]) {
                                        maxDamageUnits.add(unit);
                                    }
                                });
                        return maxDamageUnits;
                    });
                });
    }

    @NonNull
    public static Single<Boolean> burn(@NonNull Context context, @NonNull SoundManager soundManager) {
        return GwentApplication.getRepository(context).flatMap(repository -> burn(context, repository, soundManager));
    }

    @NonNull
    protected static Single<Boolean> burn(@NonNull Context context, @NonNull UnitRepository repository, @NonNull SoundManager soundManager) {
        return getBurnUnits(repository).flatMap(units -> {
            if (units.isEmpty()) {
                return Single.just(false);
            }
            return Single.create(emitter -> new AlertDialog.Builder(context)
                    .setIconAttribute(android.R.attr.alertDialogIcon)
                    .setTitle(R.string.alertDialog_burn_title)
                    .setMessage(context.getString(R.string.alertDialog_burn_msg, UnitEntity.collectionToString(context, units)))
                    .setNegativeButton(R.string.alertDialog_burn_negative, (dialog, which) -> dialog.cancel())
                    .setPositiveButton(R.string.alertDialog_burn_positive, (dialog, which) -> {
                        // noinspection CheckResult, ResultOfMethodCallIgnored
                        RemoveUnitsUseCase.remove(context, repository, units, soundManager).subscribe(() -> emitter.onSuccess(true));
                    })
                    .setCancelable(true)
                    .setOnCancelListener(dialog -> emitter.onSuccess(false))
                    .create()
                    .show());
        });
    }
}
```

6.11 DamageCalculatorUseCase.java

```
package com.peternaggschga.gwent.domain.cases;

import android.util.Pair;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;
import com.peternaggschga.gwent.domain.damage.DamageCalculatorBuildDirector;

import java.util.Collection;

import io.reactivex.rxjava3.core.Single;

public class DamageCalculatorUseCase {
    @NonNull
    public static Single<DamageCalculator> getDamageCalculator(@NonNull UnitRepository repository, @NonNull RowType row) {
        return repository.isWeather(row)
                .zipWith(repository.isHorn(row), Pair::create)
                .zipWith(repository.getUnits(row), (weatherHorn, units) ->
                        getDamageCalculator(weatherHorn.first, weatherHorn.second, units));
    }

    @NonNull
    public static DamageCalculator getDamageCalculator(boolean weather, boolean horn, @NonNull Collection<UnitEntity> units) {
        return DamageCalculatorBuildDirector.getCalculator(weather, horn, units);
    }
}
```

6.12 RemoveUnitsUseCase.java

```
package com.peternaggschga.gwent.domain.cases;

import android.app.Dialog;
import android.content.Context;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.Collection;
import java.util.Collections;

import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.CompletableEmitter;

public class RemoveUnitsUseCase {
    @NonNull
    public static Completable remove(@NonNull Context context, @NonNull UnitRepository repository,
                                     @NonNull Collection<UnitEntity> units, @NonNull SoundManager soundManager) {
        long revengeUnits = units.stream()
                .filter(unit -> unit.getAbility() == Ability.REVENGE)
                .count();
        if (revengeUnits == 0) {
            return repository.delete(units);
        }
        return Completable.create(emitter ->
                getRevengeDialog(context, repository, emitter, units, (int) revengeUnits, soundManager).show()
        );
    }

    public static Completable remove(@NonNull Context context, @NonNull UnitRepository repository, int id, @NonNull SoundManager soundManager) {
        return repository.getUnit(id)
                .flatMapCompletable(unitEntity ->
                        remove(context, repository, Collections.singletonList(unitEntity), soundManager));
    }

    @NonNull
    private static Dialog getRevengeDialog(@NonNull Context context, @NonNull UnitRepository repository,
                                           @NonNull CompletableEmitter emitter, @NonNull Collection<UnitEntity> units,
                                           @IntRange(from = 1) int revengeUnits, @NonNull SoundManager soundManager) {
        return new RevengeAlertDialogBuilderAdapter(context)
                .setPositiveCallback((dialog, which) -> {
                    // noinspection CheckResult, ResultOfMethodCallIgnored
                    repository.delete(units)
                            .andThen(RevengeAlertDialogBuilderAdapter.insertAvengers(repository, revengeUnits, soundManager))
                            .subscribe(emitter::onComplete);
                })
                .setNegativeCallback(((dialog, which) -> {
                    // noinspection CheckResult, ResultOfMethodCallIgnored
                    repository.delete(units).subscribe(emitter::onComplete);
                }))
                .create();
    }
}
```

6.13 ResetAlertDialogBuilderAdapter.java

```java
package com.peternaggschga.gwent.domain.cases;

import android.content.Context;
import android.view.View;
import android.widget.CheckBox;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;

import com.peternaggschga.gwent.R;

class ResetAlertDialogBuilderAdapter {
    @NonNull
    private final AlertDialog.Builder adapteeBuilder;

    @NonNull
    private final Callback dialogCallback;

    ResetAlertDialogBuilderAdapter(@NonNull Context context, @NonNull Callback dialogCallback) {
        this.adapteeBuilder = new AlertDialog.Builder(context)
                .setIconAttribute(android.R.attr.alertDialogIcon)
                .setTitle(R.string.alertDialog_reset_title)
                .setOnCancelListener(dialog -> dialogCallback.reset(false))
                .setNegativeButton(R.string.alertDialog_reset_negative, (dialog, which) -> dialog.cancel());
        this.dialogCallback = dialogCallback;
    }

    @NonNull
    AlertDialog create() {
        return adapteeBuilder.create();
    }

    @NonNull
    ResetAlertDialogBuilderAdapter setTrigger(@NonNull ResetDialogUseCase.Trigger trigger) {
        adapteeBuilder.setMessage((trigger != ResetDialogUseCase.Trigger.FACTION_SWITCH) ?
                        R.string.alertDialog_reset_msg_default :
                        R.string.alertDialog_reset_msg_faction_switch)
                .setCancelable(trigger != ResetDialogUseCase.Trigger.FACTION_SWITCH);
        return this;
    }

    @NonNull
    ResetAlertDialogBuilderAdapter setMonsterDialog(boolean monsterDialog) {
        if (monsterDialog) {
            View checkBoxView = View.inflate(adapteeBuilder.getContext(), R.layout.alertdialog_checkbox, null);
            adapteeBuilder.setView(checkBoxView)
                    .setPositiveButton(R.string.alertDialog_reset_positive, (dialog, which) -> {
                        CheckBox checkBox = checkBoxView.findViewById(R.id.alertDialog_checkbox);
                        dialogCallback.reset(true, checkBox.isChecked());
                    });
        } else {
            adapteeBuilder.setPositiveButton(R.string.alertDialog_reset_positive, ((dialog, which) -> dialogCallback.reset(true)));
        }
        return this;
    }

    interface Callback {
        void reset(boolean resetDecision, boolean keepUnit);

        default void reset(boolean resetDecision) {
            reset(resetDecision, false);
        }
    }
}
```

6.14 ResetDialogUseCase.java

```java
package com.peternaggschga.gwent.domain.cases;

import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_MONSTER;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_PREFERENCE_KEY;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_SCOIATAEL;

import android.content.Context;
import android.content.SharedPreferences;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.preference.PreferenceManager;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.Arrays;
import java.util.stream.Collectors;

import io.reactivex.rxjava3.core.Single;

public class ResetDialogUseCase {
    @NonNull
    public static Single<Boolean> reset(@NonNull Context context, @NonNull Trigger trigger, @NonNull
SoundManager soundManager) {
        return GwentApplication.getRepository(context)
                .flatMap(repository -> reset(context, repository, trigger, soundManager));
    }

    @NonNull
    protected static Single<Boolean> reset(@NonNull Context context, @NonNull UnitRepository
repository,
                                           @NonNull Trigger trigger, @NonNull SoundManager
soundManager) {
        return getDialogType(context, repository, trigger).flatMap(dialogType -> {
            if (dialogType == DialogType.NONE) {
                return ResetRepositoryUseCase.reset(context, repository,
soundManager).andThen(Single.just(true));
            }
            return Single.create(emitter -> new ResetAlertDialogBuilderAdapter(context,
(resetDecision, keepUnit) -> {
                if (!resetDecision) {
                    emitter.onSuccess(false);
                    return;
                }
                // noinspection CheckResult, ResultOfMethodCallIgnored
                ResetRepositoryUseCase.reset(context, repository, keepUnit, soundManager)
                        .doAfterTerminate(() -> emitter.onSuccess(true))
                        .subscribe(unit ->
                                Toast.makeText(context,

context.getString(R.string.alertDialog_factionreset_monster_toast_keep, unit.toString(context)),
                                                Toast.LENGTH_LONG)
                                        .show());
            }).setTrigger(trigger)
                    .setMonsterDialog(dialogType == DialogType.MONSTER)
                    .create()
                    .show());
        });
    }

    @NonNull
    private static Single<DialogType> getDialogType(@NonNull Context context, @NonNull UnitRepository
repository,
                                                    @NonNull Trigger trigger) {
        return Single.concat(Arrays.stream(RowType.values()).map(row ->
                        repository.isWeather(row)
                                .concatWith(repository.isHorn(row))
                                .any(state -> state)
                ).collect(Collectors.toList())).any(state -> state)
                .zipWith(repository.getUnits(), (statusEffects, units) -> {
                    SharedPreferences preferences =
PreferenceManager.getDefaultSharedPreferences(context);
                    boolean monsterDialog = trigger != Trigger.FACTION_SWITCH;
                    monsterDialog &= preferences.getInt(THEME_PREFERENCE_KEY, THEME_SCOIATAEL) ==
THEME_MONSTER;
                    monsterDialog &= units.stream().anyMatch(unit -> !unit.isEpic());
                    if (monsterDialog) {
                        return DialogType.MONSTER;
                    }
                    boolean defaultWarning = statusEffects || !units.isEmpty();
                    defaultWarning &=
preferences.getBoolean(context.getString(R.string.preference_key_warning),
                            context.getResources().getBoolean(R.bool.warning_preference_default));
                    if (defaultWarning) {
                        return DialogType.DEFAULT;
                    }
                    return DialogType.NONE;
                });
    }

    public enum Trigger {
        BUTTON_CLICK,
        FACTION_SWITCH
    }

    private enum DialogType {
        NONE,
        DEFAULT,
        MONSTER
    }
}
```

6.15 ResetRepositoryUseCase.java

```
package com.peternaggschga.gwent.domain.cases;

import android.app.Dialog;
import android.content.Context;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.List;
import java.util.Optional;
import java.util.Random;
import java.util.stream.Collectors;

import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.CompletableEmitter;
import io.reactivex.rxjava3.core.Maybe;

class ResetRepositoryUseCase {
    @NonNull
    static Maybe<UnitEntity> reset(@NonNull Context context, @NonNull UnitRepository repository,
                                   boolean keepUnit, @NonNull SoundManager soundManager) {
        return repository.getUnits()
                .flatMapMaybe(units -> {
                    Optional<UnitEntity> keptUnit = keepUnit ? getRandomUnit(units) : Optional.empty();
                    long revengeUnits = units.stream()
                            .filter(unit -> unit.getAbility() == Ability.REVENGE)
                            .count() - (keptUnit.isPresent() && keptUnit.get().getAbility() == Ability.REVENGE ? 1 : 0);
                    Completable resultAction = (revengeUnits == 0) ?
                            repository.reset(keptUnit.orElse(null)) :
                            Completable.create(emitter ->
                                    getRevengeDialog(context, repository, emitter, keptUnit.orElse(null), (int) revengeUnits, soundManager).show()
                            );
                    return resultAction.andThen(Maybe.fromOptional(keptUnit));
                });
    }

    @NonNull
    private static Optional<UnitEntity> getRandomUnit(@NonNull List<UnitEntity> units) {
        units = units.stream()
                .filter(unit -> !unit.isEpic())
                .collect(Collectors.toList());
        return units.isEmpty() ? Optional.empty() : Optional.of(units.get(new Random().nextInt(units.size())));
    }

    @NonNull
    private static Dialog getRevengeDialog(@NonNull Context context, @NonNull UnitRepository repository,
                                           @NonNull CompletableEmitter emitter, @Nullable UnitEntity keptUnit,
                                           @IntRange(from = 1) int revengeUnits, @NonNull SoundManager soundManager) {
        return new RevengeAlertDialogBuilderAdapter(context)
                .setPositiveCallback((dialogInterface, which) -> {
                    // noinspection CheckResult, ResultOfMethodCallIgnored
                    repository.reset(keptUnit)
                            .andThen(RevengeAlertDialogBuilderAdapter.insertAvengers(repository, revengeUnits, soundManager))
                            .subscribe(emitter::onComplete);
                })
                .setNegativeCallback(((dialog, which) -> {
                    // noinspection CheckResult, ResultOfMethodCallIgnored
                    repository.reset(keptUnit).subscribe(emitter::onComplete);
                }))
                .create();
    }

```

```java
    @NonNull
    public static Completable reset(@NonNull Context context, @NonNull UnitRepository repository, @NonNull SoundManager soundManager) {
        return Completable.fromMaybe(reset(context, repository, false, soundManager));
    }
}
```

6.16 RevengeAlertDialogBuilderAdapter.java

```java
package com.peternaggschga.gwent.domain.cases;

import android.content.Context;
import android.content.DialogInterface;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import io.reactivex.rxjava3.core.Completable;

class RevengeAlertDialogBuilderAdapter {
    private static final boolean AVENGER_EPIC = false;

    private static final Ability AVENGER_ABILITY = Ability.NONE;

    private static final int AVENGER_DAMAGE = 8;

    private static final Integer AVENGER_SQUAD = null;

    private static final RowType AVENGER_ROW = RowType.MELEE;

    @NonNull
    private final AlertDialog.Builder adapteeBuilder;

    RevengeAlertDialogBuilderAdapter(@NonNull Context context) {
        this.adapteeBuilder = new AlertDialog.Builder(context)
                .setIconAttribute(android.R.attr.alertDialogIcon)
                .setTitle(R.string.alertDialog_revenge_title)
                .setMessage(R.string.alertDialog_revenge_msg)
                .setCancelable(false)
                .setPositiveButton(R.string.alertDialog_revenge_positive, (dialog, which) -> dialog.cancel())
                .setNegativeButton(R.string.alertDialog_revenge_negative, (dialog, which) -> dialog.cancel());
    }

    @NonNull
    public static Completable insertAvengers(@NonNull UnitRepository repository, @IntRange(from = 0) int numberOfAvengers, @NonNull SoundManager soundManager) {
        return repository.insertUnit(AVENGER_EPIC, AVENGER_DAMAGE, AVENGER_ABILITY, AVENGER_SQUAD, AVENGER_ROW, numberOfAvengers)
                .doOnComplete(() -> soundManager.playCardAddSound(AVENGER_ROW, AVENGER_EPIC));
    }

    @NonNull
    AlertDialog create() {
        return adapteeBuilder.create();
    }

    @NonNull
    RevengeAlertDialogBuilderAdapter setPositiveCallback(@NonNull DialogInterface.OnClickListener onPositiveButtonClick) {
        adapteeBuilder.setPositiveButton(R.string.alertDialog_revenge_positive, onPositiveButtonClick);
        return this;
    }

    @NonNull
    RevengeAlertDialogBuilderAdapter setNegativeCallback(@NonNull DialogInterface.OnClickListener onNegativeButtonClick) {
        adapteeBuilder.setNegativeButton(R.string.alertDialog_revenge_negative, onNegativeButtonClick);
        return this;
    }
}
```

6.17 BondDamageCalculatorDecorator.java

```java
package com.peternaggschga.gwent.domain.damage;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.BUFFED;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import java.util.Map;
import java.util.Objects;

class BondDamageCalculatorDecorator extends DamageCalculatorDecorator {
    private final Map<Integer, Integer> idToSquadSize;

    BondDamageCalculatorDecorator(@NonNull DamageCalculator component, @NonNull Map<Integer, Integer> idToSquadSize) {
        super(component);
        if (idToSquadSize.values().stream().anyMatch(integer -> integer == null || integer < 1)) {
            throw new IllegalArgumentException("Map idToSquadSize must not contain non-positive or null values.");
        }
        this.idToSquadSize = idToSquadSize;
    }

    @Override
    public int calculateDamage(int id, @IntRange(from = 0) int damage) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0.");
        }
        return Objects.requireNonNull(idToSquadSize.getOrDefault(id, 1)) * component.calculateDamage(id, damage);
    }

    @Override
    public Color isBuffed(int id) {
        return (Objects.requireNonNull(idToSquadSize.getOrDefault(id, 0)) > 1)
                ? BUFFED
                : component.isBuffed(id);
    }
}
```

6.18 DamageCalculator.java

```java
package com.peternaggschga.gwent.domain.damage;

import androidx.annotation.IntRange;

import java.util.Collection;

public interface DamageCalculator {
    Color isBuffed(int id);

    int calculateDamage(int id, @IntRange(from = 0) int damage);

    enum Color {
        DEFAULT,
        BUFFED,
        DEBUFFED
    }
}
```

6.19 DamageCalculatorBuildDirector.java

```java
package com.peternaggschga.gwent.domain.damage;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.UnitEntity;

import java.util.Collection;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;
import java.util.stream.Collectors;

public class DamageCalculatorBuildDirector {
    @NonNull
    public static DamageCalculator getCalculator(boolean weather, boolean horn, @NonNull Collection<UnitEntity> units) {
        DamageCalculatorBuilder builder = new DamageCalculatorBuilder();
        builder.setWeather(weather);

        if (units.stream().anyMatch(unit -> unit.getAbility() == Ability.BINDING)) {
            setSquads(units, builder);
        }

        if (units.stream().anyMatch(unit -> unit.getAbility() == Ability.MORAL_BOOST)) {
            setMoralBoosts(units, builder);
        }

        if (horn || units.stream().anyMatch(unit -> unit.getAbility() == Ability.HORN)) {
            setHorns(horn, units, builder);
        }

        return builder.getResult();
    }

    private static void setSquads(@NonNull Collection<UnitEntity> units, @NonNull DamageCalculatorBuilder builder) {
        List<UnitEntity> bindingUnits = units.stream().filter(unit -> unit.getAbility() == Ability.BINDING).collect(Collectors.toList());
        Map<Integer, Integer> squadToSquadSize = new HashMap<>();
        for (UnitEntity unit : bindingUnits) {
            squadToSquadSize.putIfAbsent(unit.getSquad(), 0);
            squadToSquadSize.put(unit.getSquad(), Objects.requireNonNull(squadToSquadSize.get(unit.getSquad())) + 1);
        }
        Map<Integer, Integer> idToSquadSize = new HashMap<>();
        for (UnitEntity unit : bindingUnits) {
            idToSquadSize.put(unit.getId(), squadToSquadSize.get(unit.getSquad()));
        }
        builder.setBond(idToSquadSize);
    }

    private static void setMoralBoosts(@NonNull Collection<UnitEntity> units, @NonNull DamageCalculatorBuilder builder) {
        List<Integer> unitIds = units.stream()
                .filter(unit -> unit.getAbility() == Ability.MORAL_BOOST)
                .map(UnitEntity::getId)
                .collect(Collectors.toList());
        builder.setMoral(unitIds);
    }

    private static void setHorns(boolean horn, @NonNull Collection<UnitEntity> units, @NonNull DamageCalculatorBuilder builder) {
        List<Integer> unitIds = units.stream()
                .filter(unit -> unit.getAbility() == Ability.HORN)
                .map(UnitEntity::getId)
                .collect(Collectors.toList());
        if (horn) {
            unitIds.add(null);
        }
        builder.setHorn(unitIds);
    }
}
```

6.20 DamageCalculatorBuilder.java

```
package com.peternaggschga.gwent.domain.damage;

import androidx.annotation.NonNull;

import java.util.List;
import java.util.Map;

class DamageCalculatorBuilder {
    @NonNull
    private DamageCalculator calculator = new WeatherDamageCalculator(false);

    void setWeather(boolean weather) {
        calculator = new WeatherDamageCalculator(weather);
    }

    void setBond(@NonNull Map<Integer, Integer> idToSquad) {
        calculator = new BondDamageCalculatorDecorator(calculator, idToSquad);
    }

    void setMoral(@NonNull List<Integer> unitIds) {
        calculator = new MoralDamageCalculatorDecorator(calculator, unitIds);
    }

    void setHorn(@NonNull List<Integer> unitIds) {
        calculator = new HornDamageCalculatorDecorator(calculator, unitIds);
    }

    @NonNull
    DamageCalculator getResult() {
        return calculator;
    }
}
```

6.21 DamageCalculatorDecorator.java

```
package com.peternaggschga.gwent.domain.damage;

import androidx.annotation.NonNull;

abstract class DamageCalculatorDecorator implements DamageCalculator {
    protected final DamageCalculator component;

    DamageCalculatorDecorator(@NonNull DamageCalculator component) {
        this.component = component;
    }
}
```

6.22 HornDamageCalculatorDecorator.java

```
package com.peternaggschga.gwent.domain.damage;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.BUFFED;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import java.util.List;

class HornDamageCalculatorDecorator extends DamageCalculatorDecorator {
    private final List<Integer> unitIds;

    HornDamageCalculatorDecorator(@NonNull DamageCalculator component, @NonNull List<Integer> unitIds) {
        super(component);
        this.unitIds = unitIds;
    }

    private boolean doubleDamage(int id) {
        return unitIds.contains(null) || !(unitIds.isEmpty() || unitIds.contains(id)) || unitIds.size() > 1;
    }

    @Override
    public int calculateDamage(int id, @IntRange(from = 0) int damage) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0 but is " + damage + ".");
        }
        return (doubleDamage(id) ? 2 : 1) * component.calculateDamage(id, damage);
    }

    @Override
    public Color isBuffed(int id) {
        return doubleDamage(id) ? BUFFED : component.isBuffed(id);
    }
}
```

6.23 MoralDamageCalculatorDecorator.java

```
package com.peternaggschga.gwent.domain.damage;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.BUFFED;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import java.util.List;

class MoralDamageCalculatorDecorator extends DamageCalculatorDecorator {
    private final List<Integer> unitIds;

    MoralDamageCalculatorDecorator(@NonNull DamageCalculator component, @NonNull List<Integer> unitIds) {
        super(component);
        if (unitIds.contains(null)) {
            throw new IllegalArgumentException("List<Integer> unitIds must not contain null values.");
        }
        this.unitIds = unitIds;
    }

    @Override
    public int calculateDamage(int id, @IntRange(from = 0) int damage) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0 but is " + damage + ".");
        }
        int componentDamage = component.calculateDamage(id, damage) + unitIds.size();
        return unitIds.contains(id) ? componentDamage - 1 : componentDamage;
    }

    @Override
    public Color isBuffed(int id) {
        return (!unitIds.isEmpty() && (!unitIds.contains(id) || (unitIds.size() > 1)))
                ? BUFFED
                : component.isBuffed(id);
    }
}
```

6.24 WeatherDamageCalculator.java

```
package com.peternaggschga.gwent.domain.damage;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.DEBUFFED;
import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.DEFAULT;

import androidx.annotation.IntRange;

class WeatherDamageCalculator implements DamageCalculator {
    private final boolean weather;

    WeatherDamageCalculator(boolean weather) {
        this.weather = weather;
    }

    @Override
    public int calculateDamage(int id, @IntRange(from = 0) int damage) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must not be less than zero but is " + damage + ".");
        }
        if (damage == 0) {
            return damage;
        }
        return weather ? 1 : damage;
    }

    @Override
    public Color isBuffed(int id) {
        return weather ? DEBUFFED : DEFAULT;
    }
}
```

6.25 GwentApplication.java

```
package com.peternaggschga.gwent;

import android.app.Application;
import android.content.Context;

import androidx.annotation.NonNull;
import androidx.room.Room;

import com.peternaggschga.gwent.data.AppDatabase;
import com.peternaggschga.gwent.data.UnitRepository;

import io.reactivex.rxjava3.core.Single;

public class GwentApplication extends Application {
    private static UnitRepository repository = null;
    private AppDatabase database;

    public static Single<UnitRepository> getRepository(@NonNull Context context) {
        return ((GwentApplication) context.getApplicationContext()).getRepository();
    }

    @Override
    public void onCreate() {
        super.onCreate();
        database = Room.databaseBuilder(this, AppDatabase.class, "database").build();
    }

    public Single<UnitRepository> getRepository() {
        if (repository != null) {
            return Single.just(repository);
        }
        return UnitRepository.getRepository(database).doOnSuccess(unitRepository -> repository = unitRepository);
    }
}
```

6.26 AddCardDialog.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;

import android.content.Context;
import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.ui.dialogs.OverlayDialog;
import com.peternaggschga.gwent.ui.dialogs.cards.ShowUnitsDialog;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import io.reactivex.rxjava3.disposables.CompositeDisposable;

public class AddCardDialog extends OverlayDialog {
    @NonNull
    private final RowType row;

    @NonNull
    private final SoundManager soundManager;

    @NonNull
    private final CompositeDisposable disposables = new CompositeDisposable();

    @Nullable
    private CardNumberPickerAdapter pickerAdapter = null;

    public AddCardDialog(@NonNull ShowUnitsDialog caller, @NonNull SoundManager soundManager) {
        this(caller.getContext(), caller.getRow(), soundManager);

        setOnDismissListener(dialog -> {
            disposables.dispose();
            caller.show();
        });
    }

    public AddCardDialog(@NonNull Context context, @NonNull RowType row, @NonNull SoundManager soundManager) {
        super(context, R.layout.popup_add_card, R.id.popup_add_card_cancel_button);
        this.row = row;
        this.soundManager = soundManager;

        setOnDismissListener(dialog -> disposables.dispose());
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if (pickerAdapter == null) {
            disposables.add(
                    GwentApplication.getRepository(getContext())
                            .flatMap(UnitRepository::getUnits)
                            .map(SquadManager::new)
                            .map(squadManager ->
                                    new CardNumberPickerAdapter(findViewById(R.id.card_layout), squadManager))
                            .subscribe(cardNumberPickerAdapter -> pickerAdapter = cardNumberPickerAdapter)
            );
        }

        findViewById(R.id.popup_add_card_save_button).setOnClickListener(v -> {
            if (pickerAdapter != null) {
                disposables.add(pickerAdapter.addSelectedUnits(row).subscribe(epic -> {
                    soundManager.playCardAddSound(row, epic);
                    dismiss();
                }));
            }
        });
    }
}
```

6.27 CardNumberPickerAdapter.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;

import android.os.Handler;
import android.os.Looper;
import android.view.View;
import android.view.ViewGroup;
import android.widget.NumberPicker;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitEntity;

import org.jetbrains.annotations.Contract;

import java.util.Arrays;
import java.util.List;
import java.util.Objects;
import java.util.SortedMap;
import java.util.TreeMap;

import io.reactivex.rxjava3.core.Single;

class CardNumberPickerAdapter {
    @NonNull
    private static final List<Ability> EPIC_UNIT_ABILITIES = Arrays.asList(Ability.NONE, Ability.HORN, Ability.MORAL_BOOST);

    @NonNull
    private final ValuePicker<Boolean> epicPicker;

    @NonNull
    private final DamageValuePicker damagePicker;

    @NonNull
    private final ValuePicker<Ability> abilityPicker;

    @NonNull
    private final NumberPicker squadPicker;

    @NonNull
    private final SquadManager squadManager;

    @NonNull
    private final NumberPicker numberPicker;

    CardNumberPickerAdapter(@NonNull ViewGroup pickerGroup, @NonNull SquadManager squadManager) {
        this.squadManager = squadManager;

        SortedMap<Boolean, Integer> epicStringResources = new TreeMap<>();
        epicStringResources.put(false, R.string.add_picker_epic_normal);
        epicStringResources.put(true, R.string.add_picker_epic_epic);
        epicPicker = new StringValuePicker<>(pickerGroup.findViewById(R.id.popup_add_card_epic_picker),
                epicStringResources,
                false);

        damagePicker = new DamageValuePicker(pickerGroup.findViewById(R.id.popup_add_card_dmg_picker));

        SortedMap<Ability, Integer> abilityStringResources = new TreeMap<>();
        abilityStringResources.put(Ability.NONE, R.string.add_picker_ability_default);
        abilityStringResources.put(Ability.HORN, R.string.add_picker_ability_horn);
        abilityStringResources.put(Ability.REVENGE, R.string.add_picker_ability_revenge);
        abilityStringResources.put(Ability.BINDING, R.string.add_picker_ability_binding);
        abilityStringResources.put(Ability.MORAL_BOOST, R.string.add_picker_ability_moralBoost);
        abilityPicker = new StringValuePicker<>(pickerGroup.findViewById(R.id.popup_add_card_ability_picker),
                abilityStringResources,
                Ability.NONE);

        squadPicker = pickerGroup.findViewById(R.id.popup_add_card_binding_picker);
        squadPicker.setMinValue(1);
        squadPicker.setMaxValue(SquadManager.MAX_NR_SQUADS);

        numberPicker = pickerGroup.findViewById(R.id.popup_add_card_number_picker);
        numberPicker.setMinValue(1);
        numberPicker.setMaxValue(10);

        epicPicker.setOnValueChangedListener((picker, oldVal, newVal) -> {
            damagePicker.setEpicValues(newVal);
            abilityPicker.setSelectableValues(newVal ? EPIC_UNIT_ABILITIES : Arrays.asList(Ability.values()));
        });

        abilityPicker.setOnValueChangedListener((picker, oldVal, newVal) -> {
            if (newVal == Ability.BINDING) {
                squadPicker.setVisibility(View.VISIBLE);
                squadPicker.setValue(squadManager.getFirstSquadWithMembers());
                squadManager.onSquadChanged(squadManager.getFirstSquadWithMembers(), damagePicker);
            } else {
                squadPicker.setVisibility(View.GONE);
            }
        });

        squadPicker.setOnValueChangedListener(
                getDelayedOnValueChangeListener((picker, oldVal, newVal) -> {
                    if (newVal == picker.getValue()) {
                        squadManager.onSquadChanged(newVal, damagePicker);
                    }
                })
        );
    }

    @NonNull
    @Contract(pure = true)
    static NumberPicker.OnValueChangeListener getDelayedOnValueChangeListener(@NonNull NumberPicker.OnValueChangeListener originalListener) {
        return (picker, oldVal, newVal) -> new Handler(Looper.getMainLooper()).postDelayed(() -> {
            if (newVal == picker.getValue()) {
                originalListener.onValueChange(picker, oldVal, newVal);
            }
        }, 500);
    }

    @NonNull
    Single<Boolean> addSelectedUnits(@NonNull RowType row) {
        // handle delayed events due to delayed OnValueChangedListeners
        // delayed changes in damagePicker after hero selection
        int damage;
        if (epicPicker.getValue()) {
            damage = Arrays.stream(UnitEntity.EPIC_DAMAGE_VALUES).anyMatch(integer -> Objects.equals(integer, damagePicker.getValue())) ? damagePicker.getValue() : UnitEntity.EPIC_DAMAGE_VALUES[3];
        } else {
            damage = damagePicker.getValue();
        }

        // delayed changes in squadPicker after squad selection
        Integer squad;
        if (abilityPicker.getValue() == Ability.BINDING) {
            squad = squadPicker.getVisibility() == View.VISIBLE ? squadPicker.getValue() : squadManager.getFirstSquadWithMembers();
        } else {
            squad = null;
        }
        return GwentApplication.getRepository(numberPicker.getContext())
                .flatMapCompletable(repository ->
                        repository.insertUnit(epicPicker.getValue(),
                                damage,
                                abilityPicker.getValue(),
                                squad,
                                row,
                                numberPicker.getValue()))
                .andThen(Single.just(epicPicker.getValue()));
    }
}
```

6.28 DamageValuePicker.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;


import static com.peternaggschga.gwent.data.UnitEntity.EPIC_DAMAGE_VALUES;
import static com.peternaggschga.gwent.data.UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND;

import android.widget.NumberPicker;

```

```java
import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import java.util.Arrays;
import java.util.Collection;
import java.util.Map;
import java.util.SortedMap;
import java.util.TreeMap;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

class DamageValuePicker extends ValuePicker<Integer> {
    private boolean epicValues = false;

    DamageValuePicker(@NonNull NumberPicker picker) {
        super(picker, Arrays.stream(EPIC_DAMAGE_VALUES).collect(
                TreeMap::new,
                (map, integer) -> map.put(integer, integer),
                Map::putAll)
        );
        setEpicValues(false);
    }

    @Override
    @NonNull
    protected String getDisplayString(@NonNull Integer value) {
        if (epicValues && !getDisplayIntegers().containsKey(value)) {
            throw new IllegalStateException("When epicValues is true, displayIntegers must contain the
given key but does not contain " + value + ".");
        }
        return epicValues ? String.valueOf(getDisplayIntegers().get(value)) : String.valueOf(value);
    }

    @Override
    void setValue(@NonNull @IntRange(from = 0, to = NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) Integer value)
{
        if (epicValues) {
            super.setValue(value);
        } else if (value < 0 || value > NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) {
            throw new IllegalStateException("Value must be in [0, " +
NON_EPIC_DAMAGE_VALUES_UPPER_BOUND + "] but is " + value + ".");
        } else {
            getPicker().setValue(value);
        }
    }

    @NonNull
    @Override
    Integer getValue() {
        return epicValues ? super.getValue() : getPicker().getValue();
    }

    void setEpicValues(boolean epicValues) {
        this.epicValues = epicValues;

        if (epicValues) {
            super.setSelectableValues(Arrays.asList(EPIC_DAMAGE_VALUES), EPIC_DAMAGE_VALUES[3]);
            return;
        }
        getSelectableValues().clear();
        getSelectableValues().addAll(IntStream.rangeClosed(0,
NON_EPIC_DAMAGE_VALUES_UPPER_BOUND).boxed().collect(Collectors.toList()));
        getPicker().setDisplayedValues(null);
        getPicker().setMaxValue(NON_EPIC_DAMAGE_VALUES_UPPER_BOUND);
        getPicker().setValue(5);
    }

    @Override
    void setSelectableValues(@NonNull Collection<Integer> values, @Nullable Integer defaultValue) {
    }
}
```

6.29 SquadManager.java

```java
package com.peternaggschga.gwent.ui.dialogs.addcard;



import android.content.Context;
import android.widget.Toast;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.Ability;
import com.peternaggschga.gwent.data.UnitEntity;

import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

class SquadManager {
    public static final int MAX_NR_SQUADS = 3;

    @NonNull
    private final SquadState[] states = new SquadState[MAX_NR_SQUADS];

    SquadManager(@NonNull List<UnitEntity> units) {
        units = units.stream()
                .filter(unit -> unit.getAbility() == Ability.BINDING)
                .collect(Collectors.toList());
        for (int i = 0; i < MAX_NR_SQUADS; i++) {
            states[i] = SquadState.getState(i + 1, units);
        }
    }

    int getFirstSquadWithMembers() {
        return Arrays.stream(states)
                .filter(SquadState::hasMembers)
                .findFirst()
                .map(SquadState::getSquadNumber)
                .orElse(1);
    }

    void onSquadChanged(@IntRange(from = 1, to = MAX_NR_SQUADS) int newVal, @NonNull DamageValuePicker picker) {
        if (newVal < 1 || newVal > MAX_NR_SQUADS) {
            throw new IllegalArgumentException("NewVal must be in [1, " + MAX_NR_SQUADS + "] but is " + newVal + ".");
        }
        SquadState squad = states[newVal - 1];

        Context context = picker.getContext();
        Toast.makeText(context,
                context.getString(R.string.popUp_add_card_binding_count, squad.getSquadNumber(), squad.getSquadMembers()),
                Toast.LENGTH_SHORT).show();

        if (squad.hasMembers()) {
            picker.setValue(squad.getMemberBaseDamage());
        }
    }
}
```

6.30 SquadState.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;

import static com.peternaggschga.gwent.data.UnitEntity.NON_EPIC_DAMAGE_VALUES_UPPER_BOUND;
import static com.peternaggschga.gwent.ui.dialogs.addcard.SquadManager.MAX_NR_SQUADS;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.data.UnitEntity;

import java.util.List;
import java.util.stream.Collectors;

class SquadState {
    @IntRange(from = 1, to = MAX_NR_SQUADS)
    private final int squadNumber;

    @IntRange(from = 0)
    private final int squadMembers;

    @IntRange(from = 0, to = NON_EPIC_DAMAGE_VALUES_UPPER_BOUND)
    private final int memberBaseDamage;

    private SquadState(@IntRange(from = 1, to = MAX_NR_SQUADS) int squadNumber, @IntRange(from = 0) int squadMembers,
                       @IntRange(from = 0, to = NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) int memberBaseDamage) {
        if (squadNumber < 1 || squadNumber > MAX_NR_SQUADS) {
            throw new IllegalArgumentException("SquadNumber must be in [1, " + MAX_NR_SQUADS + "] but is " + squadNumber + ".");
        }
        if (squadMembers < 0) {
            throw new IllegalArgumentException("SquadMembers must be non-negative but is " + squadMembers + ".");
        }
        if (memberBaseDamage < 0 || memberBaseDamage > NON_EPIC_DAMAGE_VALUES_UPPER_BOUND) {
            throw new IllegalArgumentException("MemberBaseDamage must be in [0, " + NON_EPIC_DAMAGE_VALUES_UPPER_BOUND + "] but is " + memberBaseDamage + ".");
        }
        this.squadNumber = squadNumber;
        this.squadMembers = squadMembers;
        this.memberBaseDamage = memberBaseDamage;
    }

    @NonNull
    static SquadState getState(@IntRange(from = 1, to = MAX_NR_SQUADS) int squadNumber,
                               @NonNull List<UnitEntity> units) {
        if (squadNumber < 1 || squadNumber > MAX_NR_SQUADS) {
            throw new IllegalArgumentException("SquadNumber must be in [1, " + MAX_NR_SQUADS + "] but is " + squadNumber + ".");
        }
        units = units.stream()
                .filter(unit -> unit.getSquad() != null && unit.getSquad() == squadNumber)
                .collect(Collectors.toList());
        return new SquadState(squadNumber,
                units.size(),
                units.stream().findAny().map(UnitEntity::getDamage).orElse(5));
    }

    boolean hasMembers() {
        return squadMembers > 0;
    }

    @IntRange(from = 1, to = MAX_NR_SQUADS)
    int getSquadNumber() {
        return squadNumber;
    }

    @IntRange(from = 0)
    int getSquadMembers() {
        return squadMembers;
    }

    @IntRange(from = 0, to = NON_EPIC_DAMAGE_VALUES_UPPER_BOUND)
    int getMemberBaseDamage() {
        return memberBaseDamage;
    }
}
```

6.31 StringValuePicker.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;



import android.widget.NumberPicker;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;

import java.util.SortedMap;

class StringValuePicker<T extends Comparable<T» extends ValuePicker<T> {
    @SuppressWarnings("unused")
    StringValuePicker(@NonNull NumberPicker picker, @NonNull SortedMap<T, Integer> valueToStringRes) {
        this(picker, valueToStringRes, null);
    }

    StringValuePicker(@NonNull NumberPicker picker, @NonNull SortedMap<T, Integer> valueToStringRes, @Nullable T defaultValue) {
        super(picker, valueToStringRes, defaultValue);
    }

    @Override
    @NonNull
    protected String getDisplayString(@NonNull T value) {
        Integer resId;
        if ((resId = getDisplayIntegers().get(value)) == null) {
            throw new IllegalStateException("Value must be key in displayIntegers but is " + value + ".");
        }
        return getContext().getString(resId);
    }
}
```

6.32 ValuePicker.java

```
package com.peternaggschga.gwent.ui.dialogs.addcard;



import android.content.Context;
import android.widget.NumberPicker;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.annotation.UiContext;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.SortedMap;

abstract class ValuePicker<T extends Comparable<T» {
    @NonNull
    private final NumberPicker picker;

    @NonNull
    private final Map<T, Integer> displayIntegers;

    @NonNull
    private final List<T> selectableValues;

    ValuePicker(@NonNull NumberPicker picker, @NonNull SortedMap<T, Integer> valueToStringRes) {
        this(picker, valueToStringRes, null);
    }

    ValuePicker(@NonNull NumberPicker picker, @NonNull SortedMap<T, Integer> valueToStringRes,
                @Nullable T defaultValue) {
        if (valueToStringRes.isEmpty()) {
            throw new IllegalArgumentException("SortedMap<T, Integer> valueToStringRes must not be empty.");
        }
        if (defaultValue != null && !valueToStringRes.containsKey(defaultValue)) {
            throw new IllegalArgumentException("DefaultValue must be null or a key in the given SortedMap.");
        }
        this.picker = picker;
        displayIntegers = new HashMap<>(valueToStringRes);
        selectableValues = new ArrayList<>(valueToStringRes.size());

        picker.setMinValue(0);
        setSelectableValues(valueToStringRes.keySet(), defaultValue);
    }

    void setSelectableValues(@NonNull Collection<T> values, @Nullable T defaultValue) {
        if (values.isEmpty()) {
            throw new IllegalArgumentException("Collection<T> values must not be empty.");
        }
        if (!displayIntegers.keySet().containsAll(values)) {
            throw new IllegalStateException("DisplayIntegers must contain all given values.");
        }
        if (defaultValue != null && !displayIntegers.containsKey(defaultValue)) {
            throw new IllegalArgumentException("DefaultValue must be null or a key in the given SortedMap.");
        }

        picker.setDisplayedValues(null);
        picker.setValue(0);
        picker.setMaxValue(values.size() - 1);

        selectableValues.clear();
        String[] displayValues = new String[values.size()];
        int i = 0;
        for (T value : values) {
            selectableValues.add(value);
            displayValues[i++] = getDisplayString(value);
        }

        if (defaultValue != null) {
            setValue(defaultValue);
        }
        picker.setDisplayedValues(displayValues);
    }

    @NonNull
    protected NumberPicker getPicker() {
        return picker;
    }

    void setOnValueChangedListener(@NonNull OnValueChangeListener<T> onValueChangedListener) {
        picker.setOnValueChangedListener(
                CardNumberPickerAdapter.getDelayedOnValueChangeListener((picker, oldVal, newVal) ->
                        onValueChangedListener.onValueChange(ValuePicker.this,
                                selectableValues.get(oldVal),
                                selectableValues.get(newVal)))
        );
    }

    @NonNull
    T getValue() {
        return selectableValues.get(picker.getValue());
    }

    void setValue(@NonNull T value) {
        int position;
        if ((position = selectableValues.indexOf(value)) < 0) {
            throw new IllegalStateException("SelectableValues must contain the given value.");
        }
        picker.setValue(position);
    }

    @NonNull
    protected abstract String getDisplayString(@NonNull T value);

    @UiContext
    public Context getContext() {
        return picker.getContext();
    }

    interface OnValueChangeListener<T extends Comparable<T» {
        void onValueChange(@NonNull ValuePicker<T> picker, T oldVal, T newVal);
    }

    @NonNull
    protected Map<T, Integer> getDisplayIntegers() {
        return displayIntegers;
    }

    @NonNull
    protected List<T> getSelectableValues() {
        return selectableValues;
    }

    void setSelectableValues(@NonNull Collection<T> values) {
        setSelectableValues(values, null);
    }
}
```

6.33 CardListAdapter.java

```
package com.peternaggschga.gwent.ui.dialogs.cards;

import static androidx.recyclerview.widget.RecyclerView.NO_ID;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.core.util.Consumer;
import androidx.recyclerview.widget.DiffUtil;
import androidx.recyclerview.widget.ListAdapter;
import androidx.recyclerview.widget.RecyclerView;

import com.peternaggschga.gwent.R;

class CardListAdapter extends ListAdapter<CardUiState, CardListAdapter.CardViewHolder> {
    @NonNull
    private final Consumer<Integer> onCopy;

    @NonNull
    private final Consumer<Integer> onRemove;

    CardListAdapter(@NonNull Consumer<Integer> onCopy, @NonNull Consumer<Integer> onRemove) {
        super(CardUiState.DIFF_CALLBACK);
        this.onCopy = onCopy;
        this.onRemove = onRemove;
        setHasStableIds(true);
    }

    @NonNull
```

```java
    @Override
    public CardViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View cardView = LayoutInflater.from(parent.getContext()).inflate(R.layout.card, parent, false);
        return new CardViewHolder(cardView, onCopy, onRemove);
    }

    @Override
    public void onBindViewHolder(@NonNull CardViewHolder holder, @IntRange(from = 0) int position) {
        CardUiState item = getItem(position);
        holder.getDamageView().setText(item.getDamageString());
        holder.getDamageView().setBackgroundResource(item.getDamageBackgroundImageId());
        holder.getDamageView().setTextColor(item.getDamageTextColor());

        if (item.showAbility()) {
            holder.getAbilityView().setImageResource(item.getAbilityImageId());
            holder.getAbilityView().setVisibility(View.VISIBLE);
        } else {
            holder.getAbilityView().setVisibility(View.GONE);
        }

        if (item.showSquad()) {
            holder.getBindingView().setText(item.getSquadString());
            holder.getBindingView().setVisibility(View.VISIBLE);
        } else {
            holder.getBindingView().setVisibility(View.GONE);
        }

        holder.setItemId(item.getUnitId());
    }

    @Override
    public long getItemId(@IntRange(from = 0) int position) {
        return getItem(position).getUnitId();
    }

    static class CardViewHolder extends RecyclerView.ViewHolder {
        @NonNull
        private final TextView damageView;
        @NonNull
        private final ImageView abilityView;
        @NonNull
        private final TextView bindingView;
        private int itemId = (int) NO_ID;

        CardViewHolder(@NonNull View itemView,
                       @NonNull Consumer<Integer> onCopy, @NonNull Consumer<Integer> onRemove) {
            super(itemView);

            damageView = itemView.findViewById(R.id.damageView);
            abilityView = itemView.findViewById(R.id.abilityView);
            bindingView = itemView.findViewById(R.id.bindingView);

            itemView.findViewById(R.id.copyButton).setOnClickListener(v -> {
                if (itemId <= NO_ID) {
                    throw new IllegalArgumentException("ItemId must be greater than NO_ID (" + NO_ID + ") but is " + itemId + ".");
                }
                onCopy.accept(itemId);
            });
            itemView.findViewById(R.id.deleteButton).setOnClickListener(v -> {
                if (itemId <= NO_ID) {
                    throw new IllegalArgumentException("ItemId must be greater than NO_ID (" + NO_ID + ") but is " + itemId + ".");
                }
                onRemove.accept(itemId);
            });
        }

        @NonNull
        TextView getDamageView() {
            return damageView;
        }

        @NonNull
        ImageView getAbilityView() {
            return abilityView;
        }

        @NonNull
        TextView getBindingView() {
            return bindingView;
        }

        void setItemId(int itemId) {
            if (itemId <= NO_ID) {
                throw new IllegalArgumentException("ItemId must be greater than NO_ID (-1) but is " +
itemId + ".");
            }
            this.itemId = itemId;
        }
    }
}
```

6.34 CardUiState.java

```
package com.peternaggschga.gwent.ui.dialogs.cards;



import androidx.annotation.ColorInt;
import androidx.annotation.DrawableRes;
import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.recyclerview.widget.DiffUtil;

import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;

import java.util.Objects;

public class CardUiState {
    @NonNull
    public static final DiffUtil.ItemCallback<CardUiState> DIFF_CALLBACK = new DiffUtil.ItemCallback<CardUiState>() {
        @Override
        public boolean areItemsTheSame(@NonNull CardUiState oldItem, @NonNull CardUiState newItem) {
            return oldItem.unitId == newItem.unitId;
        }

        @Override
        public boolean areContentsTheSame(@NonNull CardUiState oldItem, @NonNull CardUiState newItem) {
            return Objects.equals(oldItem, newItem);
        }
    };

    public static final int UNUSED = -1;

    private final int unitId;

    @DrawableRes
    private final int damageBackgroundImageId;

    @NonNull
    private final String damageString;

    @ColorInt
    private final int damageTextColor;

    @DrawableRes
    private final int abilityImageId;

    @NonNull
    private final String squadString;

    public CardUiState(int unitId, @DrawableRes int damageBackgroundImageId, @IntRange(from = UNUSED) int damage,
                       @ColorInt int damageTextColor, @DrawableRes int abilityImageId,
                       @Nullable @IntRange(from = 1) Integer squad) {
        if (damage < UNUSED) {
            throw new IllegalArgumentException("Damage must be greater or equal to UNUSED (" + UNUSED + ") but is " + damage + ".");
        }
        if (squad != null && squad < 1) {
            throw new IllegalArgumentException("Squad must not be null or less than 1 but is " + squad + ".");
        }
        this.unitId = unitId;
        this.damageBackgroundImageId = damageBackgroundImageId;
        this.damageString = damage == UNUSED ? "" : String.valueOf(damage);
        this.damageTextColor = damageTextColor;
        this.abilityImageId = abilityImageId;
        this.squadString = squad == null ? "" : String.valueOf(squad);
    }

    public boolean showAbility() {
        return abilityImageId != UNUSED;
    }

    public boolean showSquad() {
```

```java
        return showAbility() && !squadString.isEmpty();
    }

    public int getUnitId() {
        return unitId;
    }

    @DrawableRes
    public int getDamageBackgroundImageId() {
        return damageBackgroundImageId;
    }

    @NonNull
    public String getDamageString() {
        return damageString;
    }

    @ColorInt
    public int getDamageTextColor() {
        return damageTextColor;
    }

    @DrawableRes
    public int getAbilityImageId() {
        return abilityImageId;
    }

    @NonNull
    public String getSquadString() {
        return squadString;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof CardUiState)) return false;
        CardUiState that = (CardUiState) o;
        return damageBackgroundImageId == that.damageBackgroundImageId
                && damageTextColor == that.damageTextColor
                && abilityImageId == that.abilityImageId
                && Objects.equals(damageString, that.damageString)
                && Objects.equals(squadString, that.squadString);
    }
}
```

6.35 CardUiStateFactory.java

```java
package com.peternaggschga.gwent.ui.dialogs.cards;

import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color;
import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.BUFFED;
import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.DEBUFFED;
import static com.peternaggschga.gwent.domain.damage.DamageCalculator.Color.DEFAULT;

import android.content.Context;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.domain.cases.DamageCalculatorUseCase;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;

import java.util.Collection;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;
import java.util.stream.Collectors;

public class CardUiStateFactory {
    private final boolean weather;

    private final boolean horn;

    @NonNull
    private final Map<Color, Integer> damageTextColors = new HashMap<>(Color.values().length);

    public CardUiStateFactory(@NonNull Context context, boolean weather, boolean horn) {
        this.weather = weather;
        this.horn = horn;
        damageTextColors.put(DEFAULT, context.getColor(R.color.color_damage_textColor));
        damageTextColors.put(BUFFED, context.getColor(R.color.color_damage_textColor_buffed));
        damageTextColors.put(DEBUFFED, context.getColor(R.color.color_damage_textColor_debuffed));
    }

    @NonNull
    public List<CardUiState> createCardUiState(@NonNull Collection<UnitEntity> units) {
        DamageCalculator calculator = DamageCalculatorUseCase.getDamageCalculator(weather, horn, units);
        return units.stream()
                .map(unit -> createCardUiState(unit, calculator))
                .collect(Collectors.toList());
    }

    @NonNull
    public CardUiState createCardUiState(@NonNull UnitEntity unit, @NonNull DamageCalculator calculator) {
        int damageBackgroundId = R.drawable.icon_damage_background;
        int damage = unit.calculateDamage(calculator);

        if (unit.isEpic()) {
            switch (damage) {
                case 7:
                    damageBackgroundId = R.drawable.icon_epic_damage_7;
                    break;
                case 8:
                    damageBackgroundId = R.drawable.icon_epic_damage_8;
                    break;
                case 10:
                    damageBackgroundId = R.drawable.icon_epic_damage_10;
                    break;
                case 11:
                    damageBackgroundId = R.drawable.icon_epic_damage_11;
                    break;
                case 15:
                    damageBackgroundId = R.drawable.icon_epic_damage_15;
                    break;
                case 0:
                default:
                    damageBackgroundId = R.drawable.icon_epic_damage_0;
            }
            damage = CardUiState.UNUSED;
        }

        int damageColor = Objects.requireNonNull(damageTextColors.get(unit.isBuffed(calculator)));

        int abilityImage;
        switch (unit.getAbility()) {
            case HORN:
                abilityImage = R.drawable.icon_horn;
                break;
            case REVENGE:
                abilityImage = R.drawable.icon_revenge;
                break;
            case BINDING:
                abilityImage = R.drawable.icon_binding;
                break;
            case MORAL_BOOST:
                abilityImage = R.drawable.icon_moral_boost;
                break;
            case NONE:
            default:
                abilityImage = CardUiState.UNUSED;
        }

        return new CardUiState(unit.getId(), damageBackgroundId, damage, damageColor, abilityImage, unit.getSquad());
    }
}
```

6.36 ShowUnitsDialog.java

```
package com.peternaggschga.gwent.ui.dialogs.cards;

import android.content.Context;
import android.os.Bundle;
import android.util.Log;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.DefaultItemAnimator;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitEntity;
import com.peternaggschga.gwent.domain.cases.RemoveUnitsUseCase;
import com.peternaggschga.gwent.ui.dialogs.OverlayDialog;
import com.peternaggschga.gwent.ui.dialogs.addcard.AddCardDialog;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.Objects;

import io.reactivex.rxjava3.core.Single;
import io.reactivex.rxjava3.disposables.CompositeDisposable;

public class ShowUnitsDialog extends OverlayDialog {
    @NonNull
    private final RowType row;

    @NonNull
    private final CardListAdapter cardListAdapter;

    @NonNull
    private final SoundManager soundManager;

    @NonNull
    private final CompositeDisposable disposables = new CompositeDisposable();

    private RecyclerView unitRecyclerView;

    private ShowUnitsDialog(@NonNull Context context, @NonNull RowType row,
                            @NonNull CardListAdapter cardListAdapter, @NonNull SoundManager soundManager) {
        super(context, R.layout.popup_cards, R.id.popup_cards_cancel_button);
        this.row = row;
        this.cardListAdapter = cardListAdapter;
        this.soundManager = soundManager;
    }

    @NonNull
    public static Single<ShowUnitsDialog> getDialog(@NonNull Context context, @NonNull RowType row, @NonNull SoundManager soundManager) {
        return GwentApplication.getRepository(context)
                .flatMap(repository -> repository.isWeather(row)
                        .zipWith(repository.isHorn(row), (weather, horn) ->
                                new CardUiStateFactory(context, weather, horn))
                        .map(factory -> {
                            CompositeDisposable initialDisposables = new CompositeDisposable();
                            CardListAdapter adapter = new CardListAdapter(
                                    id -> initialDisposables.add(
                                            repository
                                                    .copy(id)
                                                    .andThen(repository.getUnit(id))
                                                    .map(UnitEntity::isEpic)
                                                    .subscribe(epic -> soundManager.playCardAddSound(row, epic))
                                    ),
                                    id -> initialDisposables.add(
                                            RemoveUnitsUseCase.remove(context, repository, id, soundManager)
                                                    .subscribe(soundManager::playCardRemovedSound,
                                                            throwable -> Log.e(ShowUnitsDialog.class.getSimpleName(),
                                                                    "There has been an error with the removal of a unit. " +
                                                                            "A reason might be tapping delete buttons too fast!",
                                                                    throwable))
                                    )
                            );
                            initialDisposables.add(
                                    repository.getUnitsFlowable(row)
                                            .map(factory::createCardUiState)
                                            .subscribe(adapter::submitList)
                            );
                            ShowUnitsDialog result = new ShowUnitsDialog(context, row, adapter, soundManager);
                            result.disposables.add(initialDisposables);
                            return result;
                        })
                );
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if (unitRecyclerView == null) {
            LinearLayoutManager llm = new LinearLayoutManager(getContext());
            llm.setOrientation(RecyclerView.HORIZONTAL);
            unitRecyclerView = findViewById(R.id.cards_list);
            ((DefaultItemAnimator) Objects.requireNonNull(unitRecyclerView.getItemAnimator())).setSupportsChangeAnimations(false);
            unitRecyclerView.setLayoutManager(llm);
            unitRecyclerView.setAdapter(cardListAdapter);

            cardListAdapter.registerAdapterDataObserver(new RecyclerView.AdapterDataObserver() {
                @Override
                public void onItemRangeInserted(int positionStart, int itemCount) {
                    super.onItemRangeInserted(positionStart, itemCount);
                    int scrollPosition = positionStart + itemCount - 1;
                    if (scrollPosition >= 0) {
                        unitRecyclerView.smoothScrollToPosition(scrollPosition);
                    }
                }
            });
        }

        findViewById(R.id.popup_cards_add_button).setOnClickListener(v -> {
            hide();
            new AddCardDialog(ShowUnitsDialog.this, soundManager).show();
        });

        setOnDismissListener(dialog -> disposables.dispose());
    }

    @NonNull
    public RowType getRow() {
        return row;
    }
}
```

6.37 ChangeFactionDialog.java

```
package com.peternaggschga.gwent.ui.dialogs;

import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_MONSTER;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_NILFGAARD;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_NORTHERN_KINGDOMS;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_SCOIATAEL;

import android.content.Context;
import android.os.Bundle;
import android.view.View;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;

public class ChangeFactionDialog extends OverlayDialog {
    private final Callback callback;

    public ChangeFactionDialog(@NonNull Context context, @NonNull Callback callback) {
        super(context, R.layout.popup_faction, R.id.factionBackground);
        this.callback = callback;
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        findViewById(R.id.monsterCardView).setOnClickListener(getOnThemeClickListener(THEME_MONSTER));
        findViewById(R.id.monsterButton).setOnClickListener(getOnThemeClickListener(THEME_MONSTER));

findViewById(R.id.nilfgaardCardView).setOnClickListener(getOnThemeClickListener(THEME_NILFGAARD));
findViewById(R.id.nilfgaardButton).setOnClickListener(getOnThemeClickListener(THEME_NILFGAARD));

findViewById(R.id.northernKingdomsCardView).setOnClickListener(getOnThemeClickListener(THEME_NORTHERN_KINGDOMS));
findViewById(R.id.northernKingdomsButton).setOnClickListener(getOnThemeClickListener(THEME_NORTHERN_KINGDOMS));

findViewById(R.id.scoiataelCardView).setOnClickListener(getOnThemeClickListener(THEME_SCOIATAEL));
findViewById(R.id.scoiataelButton).setOnClickListener(getOnThemeClickListener(THEME_SCOIATAEL));
    }

    @NonNull
    private View.OnClickListener getOnThemeClickListener(@IntRange(from = THEME_MONSTER, to =
THEME_SCOIATAEL) int theme) {
        return v -> {
            cancel();
            callback.onThemeSelect(theme);
        };
    }

    public interface Callback {
        void onThemeSelect(@IntRange(from = THEME_MONSTER, to = THEME_SCOIATAEL) int theme);
    }
}
```

6.38 CoinFlipDialog.java

```
package com.peternaggschga.gwent.ui.dialogs;

import android.content.Context;
import android.os.Bundle;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;

import java.util.Random;

public class CoinFlipDialog extends OverlayDialog {
    private static final Random RANDOM = new Random();

    public CoinFlipDialog(@NonNull Context context) {
        super(context, R.layout.popup_coin_normal, R.id.coinflipBackground);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        switch (RANDOM.nextInt(100)) {
            case 0:
                setContentView(R.layout.popup_coin_stewie);
                break;
            case 1:
                setContentView(R.layout.popup_coin_terry);
                break;
            case 2:
                setContentView(R.layout.popup_coin_vin);
                break;
            default:
                if (RANDOM.nextBoolean()) {
                    ((ImageView)
findViewById(R.id.popup_coin_normal_coinView)).setImageResource(R.drawable.coin_lose);
                    ((TextView)
findViewById(R.id.popup_coin_normal_textView)).setText(R.string.popUp_coin_normal_lose);
                }
        }
    }
}
```

6.39 OverlayDialog.java

```
package com.peternaggschga.gwent.ui.dialogs;

import android.app.Dialog;
import android.content.Context;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.view.ViewGroup;
import android.view.Window;

import androidx.annotation.IdRes;
import androidx.annotation.LayoutRes;
import androidx.annotation.NonNull;

import java.util.Objects;

public class OverlayDialog extends Dialog {
    static final int NO_DISMISS_VIEW = -1;
    @NonNull
    private static final ColorDrawable BACKGROUND = new ColorDrawable(Color.TRANSPARENT);
    @LayoutRes
    private final int layout;
    @IdRes
    private final int dismissViewId;

    protected OverlayDialog(@NonNull Context context, @LayoutRes int layout, @IdRes int dismissViewId)
{
        super(context);
        this.layout = layout;
        this.dismissViewId = dismissViewId;
    }

    @SuppressWarnings("unused")
    protected OverlayDialog(@NonNull Context context, @LayoutRes int layout) {
        this(context, layout, NO_DISMISS_VIEW);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(layout);

        Window window = Objects.requireNonNull(getWindow());
        window.setLayout(ViewGroup.LayoutParams.MATCH_PARENT, ViewGroup.LayoutParams.MATCH_PARENT);
        window.setBackgroundDrawable(BACKGROUND);

        if (dismissViewId != NO_DISMISS_VIEW) {
            findViewById(dismissViewId).setOnClickListener(v -> dismiss());
        }
        setCancelable(true);
    }
}
```

6.40 IndicatorManager.java

```
package com.peternaggschga.gwent.ui.introduction;

import android.view.Window;
import android.widget.ImageView;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;

class IndicatorManager {
    @NonNull
    private final ImageView[] indicators = new ImageView[5];

    @IntRange(from = 0, to = 4)
    private int currentView = 0;

    IndicatorManager(@NonNull Window window) {
        indicators[0] = window.findViewById(R.id.introduction_indicator_0);
        indicators[1] = window.findViewById(R.id.introduction_indicator_1);
        indicators[2] = window.findViewById(R.id.introduction_indicator_2);
        indicators[3] = window.findViewById(R.id.introduction_indicator_3);
        indicators[4] = window.findViewById(R.id.introduction_indicator_4);
    }

    void updateIndicators(@IntRange(from = 0, to = 4) int position) {
        indicators[currentView].setImageResource(R.drawable.indicator_unselected);
        indicators[position].setImageResource(R.drawable.indicator_selected);
        currentView = position;
    }
}
```

6.41 IntroductionActivity.java

```
package com.peternaggschga.gwent.ui.introduction;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;

import androidx.activity.OnBackPressedCallback;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;
import androidx.viewpager2.widget.ViewPager2;

import com.peternaggschga.gwent.R;

public class IntroductionActivity extends AppCompatActivity {
    private IndicatorManager indicatorManager;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_introduction);

        if (indicatorManager == null) {
            indicatorManager = new IndicatorManager(getWindow());
        }

        final ViewPager2 viewPager = findViewById(R.id.introduction_viewPager);
        OnBackPressedCallback callback = new OnBackPressedCallback(false) {
            @Override
            public void handleOnBackPressed() {
                viewPager.setCurrentItem(viewPager.getCurrentItem() - 1);
            }
        };
        getOnBackPressedDispatcher().addCallback(callback);

        View.OnClickListener onFinish = view -> {
            PreferenceManager.getDefaultSharedPreferences(getApplicationContext())
                    .edit()
                    .putBoolean(getString(R.string.preference_first_use_key), false)
                    .apply();
            callback.setEnabled(false);
            getOnBackPressedDispatcher().onBackPressed();
        };

        final Button skipButton = findViewById(R.id.introduction_button_skip);
        final ImageButton nextButton = findViewById(R.id.introduction_button_next);
        final Button finishButton = findViewById(R.id.introduction_button_finish);

        skipButton.setOnClickListener(onFinish);
        nextButton.setOnClickListener(v -> viewPager.setCurrentItem(viewPager.getCurrentItem() + 1));
        finishButton.setOnClickListener(onFinish);

        viewPager.setAdapter(new SectionsPagerAdapter(this));
        viewPager.registerOnPageChangeCallback(new ViewPager2.OnPageChangeCallback() {
            @Override
            public void onPageSelected(int position) {
                indicatorManager.updateIndicators(position);
                skipButton.setVisibility(position == IntroductionFragment.PAGES_COUNT - 1 ? View.GONE : View.VISIBLE);
                nextButton.setVisibility(position == IntroductionFragment.PAGES_COUNT - 1 ? View.GONE : View.VISIBLE);
                finishButton.setVisibility(position == IntroductionFragment.PAGES_COUNT - 1 ? View.VISIBLE : View.GONE);
                callback.setEnabled(position != 0);
            }
        });
    }
}
```

6.42 IntroductionFragment.java

```
package com.peternaggschga.gwent.ui.introduction;

import android.os.Bundle;
import android.text.Html;
import android.text.method.LinkMovementMethod;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.peternaggschga.gwent.R;

public class IntroductionFragment extends Fragment {
    public static final int PAGES_COUNT = 5;
    private static final String ARG_SECTION_NUMBER = "section_number";
    private static final int[] imageIds = {R.drawable.introduction_1, R.drawable.introduction_2,
            R.drawable.introduction_3, R.drawable.introduction_4, R.drawable.introduction_5};
    private static final int[] stringIds = {R.string.introduction_page1, R.string.introduction_page2,
            R.string.introduction_page3, R.string.introduction_page4, R.string.introduction_page5};

    @NonNull
    public static IntroductionFragment newInstance(@IntRange(from = 0, to = PAGES_COUNT - 1) int index) {
        if (index < 0 || index >= PAGES_COUNT) {
            throw new IllegalArgumentException("Index must be in [0, PAGES_COUNT - 1] but is " + index + ".");
        }
        Bundle arguments = new Bundle();
        arguments.putInt(ARG_SECTION_NUMBER, index);
        IntroductionFragment fragment = new IntroductionFragment();
        fragment.setArguments(arguments);
        return fragment;
    }

    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        int page = getArguments() != null ? getArguments().getInt(ARG_SECTION_NUMBER) : 0;

        View root = inflater.inflate(R.layout.fragment_introduction, container, false);

        TextView textView = root.findViewById(R.id.introduction_textView);
        textView.setText(Html.fromHtml(getString(stringIds[page]), Html.FROM_HTML_MODE_LEGACY));
        textView.setMovementMethod(LinkMovementMethod.getInstance());

        ((ImageView) root.findViewById(R.id.introduction_imageView)).setImageResource(imageIds[page]);
        return root;
    }
}
```

6.43 SectionsPagerAdapter.java

```
package com.peternaggschga.gwent.ui.introduction;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentActivity;
import androidx.viewpager2.adapter.FragmentStateAdapter;

public class SectionsPagerAdapter extends FragmentStateAdapter {
    public SectionsPagerAdapter(@NonNull FragmentActivity fragmentActivity) {
        super(fragmentActivity);
    }

    @Override
    public int getItemCount() {
        return IntroductionFragment.PAGES_COUNT;
    }

    @NonNull
    @Override
    public Fragment createFragment(int position) {
        return IntroductionFragment.newInstance(position);
    }
}
```

6.44 FactionSwitchListener.java

```
package com.peternaggschga.gwent.ui.main;

import android.content.Context;
import android.content.SharedPreferences;
import android.content.res.TypedArray;
import android.os.Build;
import android.view.Window;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.preference.PreferenceManager;

import com.peternaggschga.gwent.R;

import org.jetbrains.annotations.Contract;

import java.util.ArrayList;
import java.util.List;

public class FactionSwitchListener implements SharedPreferences.OnSharedPreferenceChangeListener {
    @NonNull
    public static final String THEME_PREFERENCE_KEY = "theme";

    public static final int THEME_MONSTER = 0;

    public static final int THEME_NILFGAARD = 1;

    public static final int THEME_NORTHERN_KINGDOMS = 2;

    public static final int THEME_SCOIATAEL = 3;

    @NonNull
    private final List<ImageView> ballViews;

    @NonNull
    private final List<ImageView> cardViews;

    @NonNull
    private final List<TextView> unitViews;

    @NonNull
    private final ImageButton factionButton;

    private FactionSwitchListener(@NonNull List<ImageView> ballViews, @NonNull List<ImageView> cardViews,
                                  @NonNull List<TextView> unitViews, @NonNull ImageButton factionButton) {
        this.ballViews = ballViews;
        this.cardViews = cardViews;
        this.unitViews = unitViews;
        this.factionButton = factionButton;
    }

    @NonNull
    @Contract("_ -> new")
    public static FactionSwitchListener getListener(@NonNull Window mainWindow) {
        List<ImageView> ballViews = new ArrayList<>(4);
        List<ImageView> cardViews = new ArrayList<>(3);
        List<TextView> unitViews = new ArrayList<>(3);

        for (int id : new int[]{R.id.firstRow, R.id.secondRow, R.id.thirdRow}) {
            ConstraintLayout rowLayout = mainWindow.findViewById(id);

            ballViews.add(rowLayout.findViewById(R.id.pointBall));
            cardViews.add(rowLayout.findViewById(R.id.cardsImage));
            unitViews.add(rowLayout.findViewById(R.id.cardCountView));
        }

        ballViews.add(mainWindow.findViewById(R.id.overallPointBall));

        return new FactionSwitchListener(ballViews, cardViews, unitViews, mainWindow.findViewById(R.id.factionButton));
    }

    public static void setTheme(@NonNull Context context) {
        switch (PreferenceManager.getDefaultSharedPreferences(context)
                .getInt(THEME_PREFERENCE_KEY, THEME_SCOIATAEL)) {
            case THEME_MONSTER:
                context.setTheme(R.style.MonsterTheme);
                break;
            case THEME_NILFGAARD:
                context.setTheme(R.style.NilfgaardTheme);
                break;
            case THEME_NORTHERN_KINGDOMS:
                context.setTheme(R.style.NorthernKingdomsTheme);
                break;
            case THEME_SCOIATAEL:
            default:
                context.setTheme(R.style.ScoiataelTheme);
        }
    }

    @Override
    public void onSharedPreferenceChanged(SharedPreferences sharedPreferences, @Nullable String key) {
        if (!THEME_PREFERENCE_KEY.equals(key)) {
            return;
        }

        Context context = factionButton.getContext();

        switch (sharedPreferences.getInt(key, THEME_SCOIATAEL)) {
            case THEME_MONSTER:
                context.setTheme(R.style.MonsterTheme);
                break;
            case THEME_NILFGAARD:
                context.setTheme(R.style.NilfgaardTheme);
                break;
            case THEME_NORTHERN_KINGDOMS:
                context.setTheme(R.style.NorthernKingdomsTheme);
                break;
            case THEME_SCOIATAEL:
            default:
                context.setTheme(R.style.ScoiataelTheme);
        }

        int ballImageRes;
        int cardImageRes;
        int unitNumberTextColor;
        int factionButtonImageRes;
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.S) {
            try (TypedArray attributeValues = context.getTheme().obtainStyledAttributes(R.styleable.theme)) {
                ballImageRes = attributeValues.getResourceId(R.styleable.theme_point_ball_mipmap, R.drawable.ball_green);
                cardImageRes = attributeValues.getResourceId(R.styleable.theme_card_view_mipmap, R.drawable.card_scoiatael_landscape_free);
                unitNumberTextColor = attributeValues.getColor(R.styleable.theme_colorPrimary, context.getColor(R.color.color_text_scoiatael));
                factionButtonImageRes = attributeValues.getResourceId(R.styleable.theme_android_alertDialogIcon, R.drawable.icon_round_scoiatael);
            }
        } else {
            //noinspection resource
            TypedArray attributeValues = context.getTheme().obtainStyledAttributes(R.styleable.theme);
            ballImageRes = attributeValues.getResourceId(R.styleable.theme_point_ball_mipmap, R.drawable.ball_green);
            cardImageRes = attributeValues.getResourceId(R.styleable.theme_card_view_mipmap, R.drawable.card_scoiatael_landscape_free);
            unitNumberTextColor = attributeValues.getColor(R.styleable.theme_colorPrimary, context.getColor(R.color.color_text_scoiatael));
            factionButtonImageRes = attributeValues.getResourceId(R.styleable.theme_android_alertDialogIcon, R.drawable.icon_round_scoiatael);
            attributeValues.recycle();
        }

        ballViews.forEach(view -> ImageViewSwitchAnimator.animatedSwitch(view, ballImageRes).subscribe());
        cardViews.forEach(view -> ImageViewSwitchAnimator.animatedSwitch(view, cardImageRes).subscribe());
        unitViews.forEach(textView -> textView.setTextColor(unitNumberTextColor));
        ImageViewSwitchAnimator.animatedSwitch(factionButton, factionButtonImageRes).subscribe();
    }
}
```

6.45 GameBoardViewModel.java

```
package com.peternaggschga.gwent.ui.main;

import static androidx.lifecycle.ViewModelProvider.AndroidViewModelFactory.APPLICATION_KEY;

import android.content.Context;

import androidx.annotation.NonNull;
import androidx.lifecycle.AndroidViewModel;
import androidx.lifecycle.ViewModelProvider;
import androidx.lifecycle.ViewModelStoreOwner;
import androidx.lifecycle.viewmodel.ViewModelInitializer;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.data.UnitRepository;
import com.peternaggschga.gwent.domain.cases.BurnDialogUseCase;
import com.peternaggschga.gwent.domain.cases.DamageCalculatorUseCase;
import com.peternaggschga.gwent.domain.cases.ResetDialogUseCase;
import com.peternaggschga.gwent.domain.damage.DamageCalculator;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import java.util.HashMap;
import java.util.Map;
import java.util.Objects;
import java.util.concurrent.TimeUnit;

import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers;
import io.reactivex.rxjava3.core.Completable;
import io.reactivex.rxjava3.core.Flowable;
import io.reactivex.rxjava3.core.Single;
import io.reactivex.rxjava3.schedulers.Schedulers;

public class GameBoardViewModel extends AndroidViewModel {
    @NonNull
    private static final ViewModelInitializer<GameBoardViewModel> INITIALIZER = new ViewModelInitializer<>(
            GameBoardViewModel.class,
            creationExtras -> {
                GwentApplication app = (GwentApplication) creationExtras.get(APPLICATION_KEY);
                assert app != null;
                return new GameBoardViewModel(app);
            });

    private GameBoardViewModel(@NonNull GwentApplication application) {
        super(application);
    }

    @NonNull
    private final Map<RowType, Flowable<RowUiState» rowUiStates = new HashMap<>(RowType.values().length);

    @NonNull
    private Flowable<MenuUiState> menuUiState = Flowable.empty();

    private SoundManager soundManager;

    @NonNull
    public static GameBoardViewModel getModel(@NonNull ViewModelStoreOwner owner,
                                              @NonNull UnitRepository repository,
                                              @NonNull SoundManager soundManager) {
        GameBoardViewModel result = new ViewModelProvider(owner, ViewModelProvider.Factory.from(INITIALIZER))
                .get(GameBoardViewModel.class);

        for (RowType row : RowType.values()) {
            result.rowUiStates.put(row,
                    Flowable.combineLatest(repository.isWeatherFlowable(row),
                            repository.isHornFlowable(row),
                            repository.getUnitsFlowable(row),
                            (weather, horn, units) -> {
                                DamageCalculator calculator = DamageCalculatorUseCase.getDamageCalculator(weather, horn, units);
                                int damage = units.stream()
                                        .map(unitEntity -> unitEntity.calculateDamage(calculator))
                                        .reduce(0, Integer::sum);
                                return new RowUiState(damage, weather, horn, units.size());
                            })
                            .distinctUntilChanged()
                            .onBackpressureLatest()
                            .debounce(10, TimeUnit.MILLISECONDS)
                            .subscribeOn(Schedulers.io())
                            .observeOn(AndroidSchedulers.mainThread())
            );
        }

        Flowable<MenuUiState> combinedRowStates = Flowable.combineLatest(result.rowUiStates.values(), (Object[] rowUiStates) -> {
            int damage = 0;
            boolean reset = false;
            boolean weather = false;
            for (Object state : rowUiStates) {
                RowUiState rowUiState = (RowUiState) state;
                damage += rowUiState.getDamage();
                reset |= rowUiState.isHorn() || rowUiState.getUnits() != 0;
                weather |= rowUiState.isWeather();
            }
            reset |= weather;
            return new MenuUiState(damage, reset, weather, false);
        }).distinctUntilChanged().onBackpressureLatest();

        result.menuUiState = Flowable.combineLatest(combinedRowStates,
                        repository.hasNonEpicUnitsFlowable(),
                        (menuUiState, hasNonEpicUnits) -> new MenuUiState(menuUiState.getDamage(),
                                menuUiState.isReset(),
                                menuUiState.isWeather(),
                                hasNonEpicUnits))
                .distinctUntilChanged()
                .onBackpressureLatest()
                .debounce(10, TimeUnit.MILLISECONDS)
                .subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread());

        result.soundManager = soundManager;
        return result;
    }

    private Single<UnitRepository> getRepository() {
        return GwentApplication.getRepository(getApplication());
    }

    @NonNull
    public Flowable<RowUiState> getRowUiState(@NonNull RowType row) {
        return Objects.requireNonNull(rowUiStates.get(row));
    }

    @NonNull
    public Flowable<MenuUiState> getMenuUiState() {
        return menuUiState;
    }

    @NonNull
    public SoundManager getSoundManager() {
        return soundManager;
    }

    @NonNull
    public Completable onWeatherViewPressed(@NonNull RowType row) {
        return getRepository()
                .flatMap(repository -> repository
                        .switchWeather(row)
                        .andThen(repository.isWeather(row)))
                .doOnSuccess(weather -> {
                    if (weather) {
                        soundManager.playWeatherSound(row);
                    }
                })
                .ignoreElement();
    }

    public Completable onHornViewPressed(@NonNull RowType row) {
        return getRepository()
                .flatMap(repository -> repository
                        .switchHorn(row)
                        .andThen(repository.isHorn(row)))
                .doOnSuccess(horn -> {
                    if (horn) {
                        soundManager.playHornSound();
                    }
                })
                .ignoreElement();
    }

    @NonNull
    public Completable onResetButtonPressed(@NonNull Context context) {
        return reset(context, ResetDialogUseCase.Trigger.BUTTON_CLICK);
    }

    @NonNull
    public Completable onFactionSwitchReset(@NonNull Context context) {
        return reset(context, ResetDialogUseCase.Trigger.FACTION_SWITCH);
    }

    @NonNull
    private Completable reset(@NonNull Context context, @NonNull ResetDialogUseCase.Trigger trigger) {
        return ResetDialogUseCase
                .reset(context, trigger, soundManager)
                .doOnSuccess(playSound -> {
                    if (playSound) {
                        soundManager.playResetSound();
                    }
                })
                .ignoreElement();
    }

    @NonNull
    public Completable onWeatherButtonPressed() {
        return
getRepository().flatMapCompletable(UnitRepository::clearWeather).doOnComplete(soundManager::playClearWeatherSound);
    }

    @NonNull
    public Completable onBurnButtonPressed(@NonNull Context context) {
        return BurnDialogUseCase
                .burn(context, soundManager)
                .doOnSuccess(playSound -> {
                    if (playSound) {
                        soundManager.playBurnSound();
                    }
                })
                .ignoreElement();
    }
}
```

6.46 ImageViewSwitchAnimator.java

```
package com.peternaggschga.gwent.ui.main;

import android.view.ViewGroup;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;

import androidx.annotation.DrawableRes;
```

```java
import androidx.annotation.NonNull;
import androidx.constraintlayout.widget.ConstraintLayout;

import io.reactivex.rxjava3.core.Completable;

public class ImageViewSwitchAnimator {
    @NonNull
    private static ImageView getOverlayView(@NonNull ImageView template) {
        ImageView result = new ImageView(template.getContext());
        ViewGroup.LayoutParams oldParams = template.getLayoutParams();
        ConstraintLayout.LayoutParams newParams = new ConstraintLayout.LayoutParams(oldParams.width, oldParams.height);
        newParams.topToTop = template.getId();
        newParams.endToEnd = template.getId();
        newParams.startToStart = template.getId();
        newParams.bottomToBottom = template.getId();
        result.setLayoutParams(newParams);
        result.setImageDrawable(template.getDrawable());
        result.setScaleType(template.getScaleType());
        result.setTranslationZ(template.getTranslationZ() + 1);
        return result;
    }

    @NonNull
    public static Completable animatedSwitch(@NonNull ImageView view, @DrawableRes int resId) {
        return Completable.create(emitter -> {
            Animation fadeAnimation = AnimationUtils.loadAnimation(view.getContext(), android.R.anim.fade_out);
            ImageView foregroundView = getOverlayView(view);
            ViewGroup parent = (ViewGroup) view.getParent();

            fadeAnimation.setAnimationListener(new Animation.AnimationListener() {
                @Override
                public void onAnimationStart(Animation animation) {
                    view.setImageResource(resId);
                }

                @Override
                public void onAnimationEnd(Animation animation) {
                    parent.removeView(foregroundView);
                    emitter.onComplete();
                }

                @Override
                public void onAnimationRepeat(Animation animation) {
                }
            });

            parent.addView(foregroundView);
            foregroundView.startAnimation(fadeAnimation);
        });
    }
}
```

6.47 MainActivity.java

```java
package com.peternaggschga.gwent.ui.main;

import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_PREFERENCE_KEY;
import static com.peternaggschga.gwent.ui.main.FactionSwitchListener.THEME_SCOIATAEL;

import android.app.Dialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.view.View;
import android.view.WindowManager;
import android.widget.ImageButton;
import android.widget.ImageView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.core.view.WindowCompat;
import androidx.core.view.WindowInsetsCompat;
import androidx.core.view.WindowInsetsControllerCompat;
import androidx.preference.PreferenceManager;

import com.peternaggschga.gwent.GwentApplication;
import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;
import com.peternaggschga.gwent.ui.dialogs.ChangeFactionDialog;
import com.peternaggschga.gwent.ui.dialogs.CoinFlipDialog;
import com.peternaggschga.gwent.ui.dialogs.OverlayDialog;
import com.peternaggschga.gwent.ui.dialogs.addcard.AddCardDialog;
import com.peternaggschga.gwent.ui.dialogs.cards.ShowUnitsDialog;
import com.peternaggschga.gwent.ui.introduction.IntroductionActivity;
import com.peternaggschga.gwent.ui.settings.SettingsActivity;
import com.peternaggschga.gwent.ui.sounds.SoundManager;

import io.reactivex.rxjava3.core.Single;
import io.reactivex.rxjava3.disposables.CompositeDisposable;
import io.reactivex.rxjava3.functions.Function;

public class MainActivity extends AppCompatActivity {
    @NonNull
    private final CompositeDisposable disposables = new CompositeDisposable();

    private SharedPreferences.OnSharedPreferenceChangeListener factionSwitchListener;

    private GameBoardViewModel gameBoardViewModel;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(this);

        if (preferences.getBoolean(getString(R.string.preference_first_use_key), true)) {
            startActivity(new Intent(this, IntroductionActivity.class));
        }

        FactionSwitchListener.setTheme(this);
        setContentView(R.layout.activity_main);

        if (gameBoardViewModel == null) {
            disposables.add(
                    GwentApplication.getRepository(this)
                            .map(repository ->
                                    GameBoardViewModel.getModel(MainActivity.this,
                                            repository,
                                            new SoundManager(this)))
                            .subscribe(gameBoardViewModel -> {
                                this.gameBoardViewModel = gameBoardViewModel;
                                initializeViewModel();
                            })
            );
        }

        if (factionSwitchListener == null) {
            factionSwitchListener = FactionSwitchListener.getListener(getWindow());
            preferences.registerOnSharedPreferenceChangeListener(factionSwitchListener);
        }

        findViewById(R.id.factionButton).setOnClickListener(v -> inflateFactionPopup());
        findViewById(R.id.coinButton).setOnClickListener(v -> inflateCoinFlipPopup());
        findViewById(R.id.settingsButton).setOnClickListener(v -> startActivity(new Intent(MainActivity.this, SettingsActivity.class)));
    }

    @Override
    protected void onResume() {
        super.onResume();
        // set background image according to preferences
        int backgroundImageKey = Integer.parseInt(
                PreferenceManager.getDefaultSharedPreferences(this)
                        .getString(getString(R.string.preference_key_design),
                                getString(R.string.design_preference_default))
        );
        ImageView backgroundImage = findViewById(R.id.backgroundImageView);
        if (backgroundImageKey != 0) {
            backgroundImage.setVisibility(View.VISIBLE);
            backgroundImage.setImageResource(new int[]{
                    R.drawable.background_geralt,
                    R.drawable.background_ciri,
                    R.drawable.background_jaskier,
                    R.drawable.background_yennefer,
                    R.drawable.background_eredin
            }[backgroundImageKey - 1]);
        } else {
            backgroundImage.setVisibility(View.INVISIBLE);
        }
    }

    @Override
    public void onWindowFocusChanged(boolean hasFocus) {
        super.onWindowFocusChanged(hasFocus);

        if (hasFocus) {
            new Handler(Looper.getMainLooper()).postDelayed(() -> {
                if (!hasWindowFocus()) {
                    return;
                }

                // hide system UI
                WindowInsetsControllerCompat windowController = WindowCompat.getInsetsController(getWindow(), getWindow().getDecorView());
                windowController.hide(WindowInsetsCompat.Type.systemBars());
                windowController.hide(WindowInsetsCompat.Type.tappableElement());
                windowController.setSystemBarsBehavior(WindowInsetsControllerCompat.BEHAVIOR_SHOW_TRANSIENT_BARS_BY_SWIPE);

                // keep screen on
                getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
            }, 250);
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        disposables.dispose();
        disposables.clear();
    }

    private void initializeViewModel() {
        int[] rowIds = {R.id.firstRow, R.id.secondRow, R.id.thirdRow};
        for (int rowId = 0; rowId < rowIds.length; rowId++) {
            RowType row = RowType.values()[rowId];
            ConstraintLayout rowLayout = findViewById(rowIds[rowId]);

            ImageView weather = rowLayout.findViewById(R.id.weatherView);
            ImageView horn = rowLayout.findViewById(R.id.hornView);
            ConstraintLayout cards = rowLayout.findViewById(R.id.cardView);

            weather.setOnClickListener(v ->
                    disposables.add(gameBoardViewModel.onWeatherViewPressed(row).subscribe()));
            horn.setOnClickListener(v ->
                    disposables.add(gameBoardViewModel.onHornViewPressed(row).subscribe()));

            cards.setOnClickListener(v -> disposables.add(
                    GwentApplication.getRepository(this)
                            .flatMap(repository -> repository.countUnits(row))
                            .map(count -> count == 0)
                            .flatMap((Function<Boolean, Single<? extends OverlayDialog») rowEmpty ->
                                    rowEmpty
                                            ? Single.just(new AddCardDialog(MainActivity.this, row, gameBoardViewModel.getSoundManager()))
                                            : ShowUnitsDialog.getDialog(MainActivity.this, row, gameBoardViewModel.getSoundManager()))
                            .subscribe(Dialog::show)
            ));

            final RowUiStateObserver observer = RowUiStateObserver.getObserver(row,
                    rowLayout.findViewById(R.id.pointView),
                    weather,
                    horn,
                    rowLayout.findViewById(R.id.cardCountView));
            disposables.add(gameBoardViewModel.getRowUiState(row).subscribe(observer));
        }

        ImageButton reset = findViewById(R.id.resetButton);
        ImageButton weather = findViewById(R.id.weatherButton);
        ImageButton burn = findViewById(R.id.burnButton);

        final MenuUiStateObserver observer = new MenuUiStateObserver(findViewById(R.id.overallPointView),
                reset,
                weather,
                burn);
        disposables.add(gameBoardViewModel.getMenuUiState().subscribe(observer));

        reset.setOnClickListener(v ->
                disposables.add(gameBoardViewModel.onResetButtonPressed(this).subscribe()));
        weather.setOnClickListener(v ->
                disposables.add(gameBoardViewModel.onWeatherButtonPressed().subscribe()));
        burn.setOnClickListener(v ->
                disposables.add(gameBoardViewModel.onBurnButtonPressed(MainActivity.this).subscribe()));
    }

    private void inflateFactionPopup() {
        new ChangeFactionDialog(this, theme -> {
            SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(this);

            if (preferences.getInt(THEME_PREFERENCE_KEY, THEME_SCOIATAEL) == theme) {
                return;
            }

            boolean resetOnFactionSwitch = preferences.getBoolean(
                    getString(R.string.preference_key_faction_reset),
                    getResources().getBoolean(R.bool.faction_reset_preference_default)
            );
            if (resetOnFactionSwitch) {
                disposables.add(gameBoardViewModel.onFactionSwitchReset(this).subscribe());
            }
            preferences.edit()
                    .putInt(THEME_PREFERENCE_KEY, theme)
                    .apply();
        }).show();
    }

    private void inflateCoinFlipPopup() {
        new CoinFlipDialog(this).show();
        gameBoardViewModel.getSoundManager().playCoinSound();
    }
}
```

6.48 MenuUiState.java

```
package com.peternaggschga.gwent.ui.main;



import androidx.annotation.IntRange;

public class MenuUiState {
    @IntRange(from = 0)
    private final int damage;

    private final boolean reset;

    private final boolean weather;

    private final boolean burn;

    public MenuUiState(@IntRange(from = 0) int damage, boolean reset, boolean weather, boolean burn) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0 but is " + damage + ".");
        }
        this.damage = damage;
        this.reset = reset;
        this.weather = weather;
        this.burn = burn;
    }

    @IntRange(from = 0)
    public int getDamage() {
        return damage;
    }

    public boolean isReset() {
        return reset;
    }

    public boolean isWeather() {
        return weather;
    }

    public boolean isBurn() {
        return burn;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof MenuUiState)) return false;
        MenuUiState that = (MenuUiState) o;
        return damage == that.damage && reset == that.reset && weather == that.weather && burn == that.burn;
    }
}
```

6.49 MenuUiStateObserver.java

```
package com.peternaggschga.gwent.ui.main;
```

```java

import android.widget.ImageButton;
import android.widget.TextView;

import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;

import io.reactivex.rxjava3.functions.Consumer;

public class MenuUiStateObserver implements Consumer<MenuUiState> {
    @NonNull
    private final TextView damageView;

    @NonNull
    private final ImageButton resetButton;

    @NonNull
    private final ImageButton weatherButton;

    @NonNull
    private final ImageButton burnButton;

    public MenuUiStateObserver(@NonNull TextView damageView, @NonNull ImageButton resetButton,
                               @NonNull ImageButton weatherButton, @NonNull ImageButton burnButton) {
        this.damageView = damageView;
        this.resetButton = resetButton;
        this.weatherButton = weatherButton;
        this.burnButton = burnButton;
    }

    @Override
    public void accept(@NonNull MenuUiState menuUiState) {
        damageView.setText(String.valueOf(menuUiState.getDamage()));
        resetButton.setClickable(menuUiState.isReset());
        ImageViewSwitchAnimator.animatedSwitch(resetButton, menuUiState.isReset() ? R.drawable.icon_reset : R.drawable.icon_reset_grey)
                .subscribe();
        weatherButton.setClickable(menuUiState.isWeather());
        ImageViewSwitchAnimator.animatedSwitch(weatherButton, menuUiState.isWeather() ? R.drawable.icon_weather : R.drawable.icon_weather_grey)
                .subscribe();
        burnButton.setClickable(menuUiState.isBurn());
        ImageViewSwitchAnimator.animatedSwitch(burnButton, menuUiState.isBurn() ? R.drawable.icon_burn : R.drawable.icon_burn_grey)
                .subscribe();
    }
}
```

6.50 RowUiState.java

```java
package com.peternaggschga.gwent.ui.main;



import androidx.annotation.IntRange;

public class RowUiState {
    @IntRange(from = 0)
    private final int damage;

    private final boolean weather;

    private final boolean horn;

    @IntRange(from = 0)
    private final int units;

    public RowUiState(@IntRange(from = 0) int damage, boolean weather, boolean horn, @IntRange(from = 0) int units) {
        if (damage < 0) {
            throw new IllegalArgumentException("Damage must be greater or equal to 0 but is " + damage + ".");
        }
        if (units < 0) {
            throw new IllegalArgumentException("Units must be greater or equal to 0 but is " + units + ".");
        }
        this.damage = damage;
        this.weather = weather;
        this.horn = horn;
        this.units = units;
    }

    @IntRange(from = 0)
    public int getDamage() {
        return damage;
    }

    public boolean isWeather() {
        return weather;
    }

    public boolean isHorn() {
        return horn;
    }

    @IntRange(from = 0)
    public int getUnits() {
        return units;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof RowUiState)) return false;
        RowUiState state = (RowUiState) o;
        return damage == state.damage && weather == state.weather && horn == state.horn && units == state.units;
    }
}
```

6.51 RowUiStateObserver.java

```
package com.peternaggschga.gwent.ui.main;

import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.DrawableRes;
import androidx.annotation.NonNull;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;

import org.jetbrains.annotations.Contract;

import io.reactivex.rxjava3.functions.Consumer;

public class RowUiStateObserver implements Consumer<RowUiState> {
    @NonNull
    private final TextView damageView;

    @NonNull
    private final ImageView weatherView;

    @DrawableRes
    private final int weatherResource;

    @NonNull
    private final ImageView hornView;

    @NonNull
    private final TextView unitView;

    private RowUiStateObserver(@NonNull TextView damageView, @NonNull ImageView weatherView,
                               @DrawableRes int weatherResource, @NonNull ImageView hornView,
                               @NonNull TextView unitView) {
        this.damageView = damageView;
        this.weatherView = weatherView;
        this.weatherResource = weatherResource;
        this.hornView = hornView;
        this.unitView = unitView;
    }

    @NonNull
    @Contract("_, _, _, _, _ -> new")
    public static RowUiStateObserver getObserver(@NonNull RowType row, @NonNull TextView damageView,
                                                 @NonNull ImageView weatherView, @NonNull ImageView hornView,
                                                 @NonNull TextView unitView) {
        int weatherResource;
        switch (row) {
            case MELEE:
                weatherResource = R.drawable.frost_weather;
                break;
            case RANGE:
                weatherResource = R.drawable.fog_weather;
                break;
            case SIEGE:
                weatherResource = R.drawable.rain_weather;
                break;
            default:
                weatherResource = R.drawable.good_weather;
        }
        return new RowUiStateObserver(damageView, weatherView, weatherResource, hornView, unitView);
    }

    @Override
    public void accept(@NonNull RowUiState rowUiState) {
        damageView.setText(String.valueOf(rowUiState.getDamage()));
        ImageViewSwitchAnimator.animatedSwitch(weatherView, rowUiState.isWeather() ? weatherResource : R.drawable.good_weather)
                .subscribe();
        ImageViewSwitchAnimator.animatedSwitch(hornView, rowUiState.isHorn() ? R.drawable.horn : R.drawable.horn_grey)
                .subscribe();
        unitView.setText(String.valueOf(rowUiState.getUnits()));
    }
}
```

6.52 RuleActivity.java

```
package com.peternaggschga.gwent.ui.settings;

import android.os.Bundle;
import android.text.Html;
import android.view.MenuItem;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;

import com.peternaggschga.gwent.R;

import java.util.Objects;

public class RuleActivity extends AppCompatActivity {
    public static final String INTENT_EXTRA_TAG = "rule_section";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_rules);

        RuleSection section = Objects.requireNonNull((RuleSection) getIntent().getSerializableExtra(INTENT_EXTRA_TAG));

        setSupportActionBar(findViewById(R.id.rulesToolbar));

        TextView textView = findViewById(R.id.rulesTextView);

        ActionBar actionBar = Objects.requireNonNull(getSupportActionBar());
        actionBar.setDisplayHomeAsUpEnabled(true);

        switch (section) {
            case GENERAL:
                textView.setText(Html.fromHtml(getString(R.string.rules_general_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_general_title);
                break;
            case COURSE:
                textView.setText(Html.fromHtml(getString(R.string.rules_course_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_course_title);
                break;
            case FACTIONS:
                textView.setText(Html.fromHtml(getString(R.string.rules_factions_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_factions_title);
                break;
            case COMMANDER:
                textView.setText(Html.fromHtml(getString(R.string.rules_commander_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_commander_title);
                break;
            case CARDS:
                textView.setText(Html.fromHtml(getString(R.string.rules_cards_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_cards_title);
                break;
            case CARD_ABILITIES:
                textView.setText(Html.fromHtml(getString(R.string.rules_card_abilities_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_card_abilities_title);
                break;
            case SPECIAL_CARDS:
                textView.setText(Html.fromHtml(getString(R.string.rules_special_cards_text),
                        Html.FROM_HTML_MODE_LEGACY));
                actionBar.setTitle(R.string.preference_rules_special_cards_title);
        }
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        if (item.getItemId() == android.R.id.home) {
            getOnBackPressedDispatcher().onBackPressed();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

6.53 RuleSection.java

```
package com.peternaggschga.gwent.ui.settings;

enum RuleSection {
    GENERAL,

    COURSE,

    FACTIONS,

    COMMANDER,

    CARDS,

    CARD_ABILITIES,

    SPECIAL_CARDS
}
```

6.54 SettingsActivity.java

```
package com.peternaggschga.gwent.ui.settings;

import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.MenuItem;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.ui.main.FactionSwitchListener;

import java.util.Objects;

public class SettingsActivity extends AppCompatActivity implements
PreferenceFragmentCompat.OnPreferenceStartFragmentCallback {
    private static final String CURRENT_FRAGMENT_KEY = "currentFragment";

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        FactionSwitchListener.setTheme(this);

        setContentView(R.layout.activity_settings);

        setSupportActionBar(findViewById(R.id.settingsToolbar));
        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null) {
            actionBar.setDisplayHomeAsUpEnabled(true);
        }

        Fragment currentFragment = null;
        if (savedInstanceState != null) {
            currentFragment = getSupportFragmentManager().getFragment(savedInstanceState,
CURRENT_FRAGMENT_KEY);
```

```
        }
        currentFragment = currentFragment == null ? new SettingsHeaderFragment() : currentFragment;
        getSupportFragmentManager().beginTransaction()
                .replace(R.id.settingsFrameLayout, currentFragment)
                .commit();
    }

    @Override
    protected void onSaveInstanceState(@NonNull Bundle outState) {
        super.onSaveInstanceState(outState);

        getSupportFragmentManager().getFragments()
                .stream()
                .filter(Fragment::isVisible)
                .findAny()
                .ifPresent(fragment ->
                        getSupportFragmentManager().putFragment(outState, CURRENT_FRAGMENT_KEY, fragment));
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        if (item.getItemId() == android.R.id.home) {
            getOnBackPressedDispatcher().onBackPressed();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    public boolean onPreferenceStartFragment(@NonNull PreferenceFragmentCompat caller, @NonNull Preference pref) {
        // Instantiate the new Fragment
        final Bundle args = pref.getExtras();

        Fragment fragment = getSupportFragmentManager()
                .getFragmentFactory()
                .instantiate(getClassLoader(), Objects.requireNonNull(pref.getFragment()));
        fragment.setArguments(args);

        // Replace the existing Fragment with the new Fragment
        getSupportFragmentManager().beginTransaction()
                .replace(R.id.settingsFrameLayout, fragment)
                .addToBackStack(null)
                .commit();
        setTitle(pref.getTitle());
        return true;
    }
}
```

6.55 SettingsHeaderFragment.java

```
package com.peternaggschga.gwent.ui.settings;

import android.content.Intent;
import android.os.Bundle;

import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.ui.introduction.IntroductionActivity;

import java.util.Objects;

public class SettingsHeaderFragment extends PreferenceFragmentCompat {
    @Override
    public void onCreatePreferences(Bundle savedInstanceState, String rootKey) {
        setPreferencesFromResource(R.xml.header_preferences, rootKey);

        Preference introductionPreference = Objects.requireNonNull(findPreference(getString(R.string.preference_key_introduction)));
        introductionPreference.setOnPreferenceClickListener(preference -> {
            startActivity(new Intent(getContext(), IntroductionActivity.class));
            return true;
        });
    }
}
```

6.56 SettingsRuleFragment.java

```
package com.peternaggschga.gwent.ui.settings;
```

```

import static com.peternaggschga.gwent.ui.settings.RuleSection.CARDS;
import static com.peternaggschga.gwent.ui.settings.RuleSection.CARD_ABILITIES;
import static com.peternaggschga.gwent.ui.settings.RuleSection.COMMANDER;
import static com.peternaggschga.gwent.ui.settings.RuleSection.COURSE;
import static com.peternaggschga.gwent.ui.settings.RuleSection.FACTIONS;
import static com.peternaggschga.gwent.ui.settings.RuleSection.GENERAL;
import static com.peternaggschga.gwent.ui.settings.RuleSection.SPECIAL_CARDS;

import android.content.Intent;
import android.os.Bundle;

import androidx.annotation.Keep;
import androidx.annotation.NonNull;
import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;

import com.peternaggschga.gwent.R;

import org.jetbrains.annotations.Contract;

import java.io.Serializable;
import java.util.Objects;

@Keep
public class SettingsRuleFragment extends PreferenceFragmentCompat {
    @NonNull
    @Contract(pure = true)
    private Preference.OnPreferenceClickListener getSectionClickListener(@NonNull RuleSection section)
{
        return preference -> {
            startActivity(
                    new Intent(getContext(), RuleActivity.class)
                            .putExtra(RuleActivity.INTENT_EXTRA_TAG, section)
            );
            return true;
        };
    }

    @Override
    public void onCreatePreferences(Bundle savedInstanceState, String rootKey) {
        setPreferencesFromResource(R.xml.rule_preferences, rootKey);

        Preference rulesGeneral =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_general_key)));
        rulesGeneral.setOnPreferenceClickListener(getSectionClickListener(GENERAL));

        Preference rulesCourse =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_course_key)));
        rulesCourse.setOnPreferenceClickListener(getSectionClickListener(COURSE));

        Preference ruleFactions =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_factions_key)));
        ruleFactions.setOnPreferenceClickListener(getSectionClickListener(FACTIONS));

        Preference rulesCommander =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_commander_key)));
        rulesCommander.setOnPreferenceClickListener(getSectionClickListener(COMMANDER));

        Preference rulesCards =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_cards_key)));
        rulesCards.setOnPreferenceClickListener(getSectionClickListener(CARDS));

        Preference ruleCardAbilities =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_card_abilities_key)));
        ruleCardAbilities.setOnPreferenceClickListener(getSectionClickListener(CARD_ABILITIES));

        Preference rulesSpecialCards =
Objects.requireNonNull(findPreference(getString(R.string.preference_rules_special_cards_key)));
        rulesSpecialCards.setOnPreferenceClickListener(getSectionClickListener(SPECIAL_CARDS));
    }
}
```

6.57 SettingsSoundFragment.java

```
package com.peternaggschga.gwent.ui.settings;

import android.os.Bundle;

import androidx.annotation.Keep;
import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;

import com.peternaggschga.gwent.R;

```

```
@Keep
public class SettingsSoundFragment extends PreferenceFragmentCompat {
    @Override
    public void onCreatePreferences(Bundle savedInstanceState, String rootKey) {
        setPreferencesFromResource(R.xml.sound_preferences, rootKey);
    }
}
```

6.58 Sound.java

```
package com.peternaggschga.gwent.ui.sounds;

import android.content.Context;
import android.content.SharedPreferences;
import android.media.SoundPool;

import androidx.annotation.NonNull;
import androidx.annotation.RawRes;
import androidx.annotation.StringRes;
import androidx.preference.PreferenceManager;

import com.peternaggschga.gwent.R;

import org.jetbrains.annotations.Contract;

class Sound {
    @NonNull
    private final String preferenceKey;

    private final int soundId;

    private boolean activated;

    private Sound(@NonNull String preferenceKey, int soundId, boolean activated) {
        this.preferenceKey = preferenceKey;
        this.soundId = soundId;
        this.activated = activated;
    }

    @NonNull
    @Contract("_, _, _, _ -> new")
    static Sound createSound(@NonNull Context context, @StringRes int preferenceRes,
                             @NonNull SoundPool soundPool, @RawRes int resId) {
        String preferenceKey = context.getString(preferenceRes);
        int soundId = soundPool.load(context, resId, 1);
        SharedPreferences pref = PreferenceManager.getDefaultSharedPreferences(context);
        boolean activated = context.getResources().getBoolean(R.bool.sound_preference_default);
        activated = pref.getBoolean(context.getString(R.string.preference_key_sounds_all), activated)
                && pref.getBoolean(preferenceKey, activated);
        return new Sound(preferenceKey, soundId, activated);
    }

    @NonNull
    String getPreferenceKey() {
        return preferenceKey;
    }

    int getSoundId() {
        return soundId;
    }

    boolean isActivated() {
        return activated;
    }

    void setActivated(boolean activated) {
        this.activated = activated;
    }

    void setActivated(@NonNull SharedPreferences sharedPreferences) {
        setActivated(sharedPreferences.getBoolean(preferenceKey, activated));
    }
}
```

6.59 SoundManager.java

```
package com.peternaggschga.gwent.ui.sounds;

import android.content.Context;
import android.content.SharedPreferences;
import android.media.AudioAttributes;
import android.media.SoundPool;
```

```java
import android.os.Build;

import androidx.annotation.IntRange;
import androidx.annotation.NonNull;
import androidx.preference.PreferenceManager;

import com.peternaggschga.gwent.R;
import com.peternaggschga.gwent.data.RowType;

import java.util.Arrays;

public class SoundManager {
    public static final int SOUND_WEATHER_GOOD = 0;

    public static final int SOUND_WEATHER_FROST = 1;

    public static final int SOUND_WEATHER_FOG = 2;

    public static final int SOUND_WEATHER_RAIN = 3;

    public static final int SOUND_HORN = 4;

    public static final int SOUND_CARDS_EPIC = 5;

    public static final int SOUND_CARDS_MELEE = 6;

    public static final int SOUND_CARDS_RANGE = 7;

    public static final int SOUND_CARDS_SIEGE = 8;

    public static final int SOUND_RESET = 9;

    public static final int SOUND_BURN = 10;

    public static final int SOUND_COIN = 11;

    @NonNull
    private final Sound[] sounds = new Sound[SOUND_COIN + 1];

    @NonNull
    private final SoundPool soundPool;

    @NonNull
    @SuppressWarnings("FieldCanBeLocal")
    private final SharedPreferences.OnSharedPreferenceChangeListener changeListener;

    public SoundManager(@NonNull Context context) {
        AudioAttributes attributes = new AudioAttributes.Builder()
                .setUsage(AudioAttributes.USAGE_GAME)
                .setContentType(AudioAttributes.CONTENT_TYPE_SONIFICATION)
                .build();

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.UPSIDE_DOWN_CAKE) {
            soundPool = new
SoundPool.Builder().setMaxStreams(5).setAudioAttributes(attributes).setContext(context).build();
        } else {
            soundPool = new
SoundPool.Builder().setMaxStreams(5).setAudioAttributes(attributes).build();
        }

        sounds[SOUND_WEATHER_GOOD] = Sound.createSound(context,
R.string.preference_key_sounds_weather, soundPool, R.raw.weather_good);
        sounds[SOUND_WEATHER_FROST] = Sound.createSound(context,
R.string.preference_key_sounds_weather, soundPool, R.raw.weather_frost);
        sounds[SOUND_WEATHER_FOG] = Sound.createSound(context, R.string.preference_key_sounds_weather,
soundPool, R.raw.weather_fog);
        sounds[SOUND_WEATHER_RAIN] = Sound.createSound(context,
R.string.preference_key_sounds_weather, soundPool, R.raw.weather_rain);
        sounds[SOUND_HORN] = Sound.createSound(context, R.string.preference_key_sounds_horn,
soundPool, R.raw.horn);
        sounds[SOUND_CARDS_EPIC] = Sound.createSound(context, R.string.preference_key_sounds_cards,
soundPool, R.raw.card_epic);
        sounds[SOUND_CARDS_MELEE] = Sound.createSound(context, R.string.preference_key_sounds_cards,
soundPool, R.raw.card_melee);
        sounds[SOUND_CARDS_RANGE] = Sound.createSound(context, R.string.preference_key_sounds_cards,
soundPool, R.raw.card_range);
        sounds[SOUND_CARDS_SIEGE] = Sound.createSound(context, R.string.preference_key_sounds_cards,
soundPool, R.raw.card_siege);
        sounds[SOUND_RESET] = Sound.createSound(context, R.string.preference_key_sounds_reset,
soundPool, R.raw.reset);
        sounds[SOUND_BURN] = Sound.createSound(context, R.string.preference_key_sounds_burn,
soundPool, R.raw.burn);
        sounds[SOUND_COIN] = Sound.createSound(context, R.string.preference_key_sounds_coin,
soundPool, R.raw.coin);

        SharedPreferences pref = PreferenceManager.getDefaultSharedPreferences(context);
        changeListener = (sharedPreferences, key) -> {
            if (context.getResources().getString(R.string.preference_key_sounds_all).equals(key)) {
                if (sharedPreferences.getBoolean(key,
                        context.getResources().getBoolean(R.bool.sound_preference_default))) {
                    Arrays.stream(sounds).forEach(sound -> sound.setActivated(sharedPreferences));
                } else {
                    Arrays.stream(sounds).forEach(sound -> sound.setActivated(false));
                }
            } else {
                Arrays.stream(sounds)
                        .filter(sound -> sound.getPreferenceKey().equals(key))
                        .forEach(sound -> sound.setActivated(sharedPreferences.getBoolean(key,
                                context.getResources().getBoolean(R.bool.sound_preference_default))));
            }
        };
        pref.registerOnSharedPreferenceChangeListener(changeListener);
    }

    public void playSound(@IntRange(from = SOUND_WEATHER_GOOD, to = SOUND_COIN) int soundId) {
        if (sounds[soundId].isActivated()) {
            soundPool.play(sounds[soundId].getSoundId(), 1, 1, 0, 0, 1);
        }
    }

    public void playClearWeatherSound() {
        playSound(SOUND_WEATHER_GOOD);
    }

    public void playWeatherSound(@NonNull RowType row) {
        switch (row) {
            case MELEE:
                playSound(SOUND_WEATHER_FROST);
                break;
            case RANGE:
                playSound(SOUND_WEATHER_FOG);
                break;
            case SIEGE:
                playSound(SOUND_WEATHER_RAIN);
        }
    }

    public void playHornSound() {
        playSound(SOUND_HORN);
    }

    public void playCardAddSound(@NonNull RowType row, boolean epic) {
        if (epic) {
            playSound(SOUND_CARDS_EPIC);
            return;
        }
        switch (row) {
            case MELEE:
                playSound(SOUND_CARDS_MELEE);
                break;
            case RANGE:
                playSound(SOUND_CARDS_RANGE);
                break;
            case SIEGE:
                playSound(SOUND_CARDS_SIEGE);
        }
    }

    public void playCardRemovedSound() {
        playSound(SOUND_RESET);
    }

    public void playResetSound() {
        playSound(SOUND_RESET);
    }

    public void playBurnSound() {
        playSound(SOUND_BURN);
    }

    public void playCoinSound() {
        playSound(SOUND_COIN);
    }
}
```

6.60 background_drawable.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:width="200dp"
    android:height="117.63864dp"
    android:viewportWidth="510.03275"
    android:viewportHeight="299.9978">

    <path
        android:fillColor="#5f4f2d"
        android:pathData="M-0,-0h510.033v299.998h-510.033z"
        android:strokeWidth="0.264583"
        android:strokeColor="#00000000" />

    <path
        android:fillColor="#35270a"
        android:pathData="m495.074,15h15v-14.997zM15.004,15L0,15v-15zM465.07,15h30.004l-15,-15zM435.065,15h30.004l-15,-
        android:strokeWidth="0.264583"
        android:strokeColor="#00000000"
        android:strokeLineCap="butt"
        android:strokeLineJoin="miter"
        tools:ignore="VectorPath" />
</vector>
```

6.61 icon_copy.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="50dp"
    android:height="50dp"
    android:autoMirrored="true"
    android:viewportWidth="112.7125"
    android:viewportHeight="112.7125">

    <path
        android:fillColor="?attr/card_view_button_background"
        android:pathData="M56.356,56.356m-56.356,0a56.356,56.356 0,1 1,112.713 0a56.356,56.356 0,1
1,-112.713 0"
        android:strokeWidth="0.907308" />

    <path
        android:fillColor="#ffffff"
        android:pathData="M63.681,91.414L38.472,91.414c-5.822,0 -10.559,-4.914
-10.559,-10.955L27.914,43.347c0,-6.041 4.736,-10.955 10.559,-10.955h25.209c5.822,0 10.559,4.914
10.559,10.955v37.111c0,6.041 -4.736,10.955 -10.559,10.955zM38.472,37.869c-2.911,0 -5.279,2.457
-5.279,5.478v37.111c0,3.02 2.369,5.478 5.279,5.478h25.209c2.911,0 5.279,-2.457
5.279,-5.478L68.961,43.347c0,-3.02 -2.369,-5.478 -5.279,-5.478zM84.799,73.611L84.799,32.254c0,-6.041
-4.736,-10.955 -10.559,-10.955L44.94,21.299c-1.458,0 -2.64,1.226 -2.64,2.739 0,1.513 1.182,2.739
2.64,2.739h29.301c2.911,0 5.279,2.457 5.279,5.478v41.357c0,1.513 1.182,2.739 2.64,2.739 1.458,0
2.64,-1.226 2.64,-2.739zM84.799,73.611"
        android:strokeWidth="0.13444" />
</vector>
```

6.62 icon_delete.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="50dp"
    android:height="50dp"
    android:viewportWidth="150.28333"
    android:viewportHeight="150.28333">

    <path
        android:fillColor="?attr/card_view_button_background"
        android:pathData="M75.142,75.142m-75.142,0a75.142,75.142 0,1 1,150.283 0a75.142,75.142 0,1
1,-150.283 0"
        android:strokeWidth="0.264583" />

    <path
        android:fillColor="#ffffff"
        android:pathData="m112.311,48.066l-63.967,63.847l-10.372,-10.393l63.967,-63.847zM48.414,37.623
L112.261,101.59 101.868,111.962 38.021,47.996z"
        android:strokeWidth="0.264583" />
</vector>
```

6.63 icon_design.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">

    <path
        android:fillColor="@android:color/white"
        android:pathData="M21,19V5c0,-1.1 -0.9,-2 -2,-2H5c-1.1,0 -2,0.9 -2,2v14c0,1.1 0.9,2 2,2h14c1.1,0 2,-0.9 2,-2zM8.5,13.5l2.5,3.01L14.5,12l4.5,6H5l3.5,-4.5z" />
</vector>
```

6.64 icon_explore.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:autoMirrored="true"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M12,10.9c-0.61,0 -1.1,0.49 -1.1,1.1s0.49,1.1 1.1,1.1c0.61,0 1.1,-0.49 1.1,-1.1s-0.49,-1.1 -1.1,-1.1zM12,2C6.48,2 2,6.48 2,12s4.48,10 10,10 10,-4.48 10,-10S17.52,2 12,2zM14.19,14.19L6,18l3.81,-8.19L18,6l-3.81,8.19z" />
</vector>
```

6.65 icon_factionreset.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">

    <path
        android:fillColor="@android:color/white"
        android:pathData="M12,6v3l4,-4 -4,-4v3c-4.42,0 -8,3.58 -8,8 0,1.57 0.46,3.03 1.24,4.26L6.7,14.8c-0.45,-0.83 -0.7,-1.79 -0.7,-2.8 0,-3.31 2.69,-6 6,-6zM18.76,7.74L17.3,9.2c0.44,0.84 0.7,1.79 0.7,2.8 0,3.31 -2.69,6 -6,6v-3l-4,4 4,4v-3c4.42,0 8,-3.58 8,-8 0,-1.57 -0.46,-3.03 -1.24,-4.26z" />
</vector>
```

6.66 icon_next.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:autoMirrored="true"
    android:tint="#FFFFFF"
    android:viewportWidth="24"
    android:viewportHeight="24">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M10,6L8.59,7.41 13.17,12l-4.58,4.59L10,18l6,-6z" />
</vector>
```

6.67 icon_rules.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">

    <path
        android:fillColor="@android:color/white"
        android:pathData="M18,2H6c-1.1,0 -2,0.9 -2,2v16c0,1.1 0.9,2 2,2h12c1.1,0 2,-0.9 2,-2V4c0,-1.1 -0.9,-2 -2,-2zM6,4h5v8l-2.5,-1.5L6,12V4z" />
</vector>
```

6.68 icon_sounds.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">

    <path
        android:fillColor="@android:color/white"
        android:pathData="M12,3v10.55c-0.59,-0.34 -1.27,-0.55 -2,-0.55 -2.21,0 -4,1.79 -4,4s1.79,4 4,4 4,-1.79 4,-4V7h4V3h-6z" />
</vector>
```

6.69 icon_warnings.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:tint="?android:attr/colorControlNormal"
    android:viewportWidth="24"
    android:viewportHeight="24">

    <path
        android:fillColor="@android:color/white"
        android:pathData="M1,21h22L12,2 1,21zM13,18h-2v-2h2v2zM13,14h-2v-4h2v4z" />
</vector>
```

6.70 indicator_selected.xml

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="oval">
    <corners android:radius="100dp" />
    <solid android:color="@android:color/primary_text_dark" />
</shape>
```

6.71 indicator_unselected.xml

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="oval">
    <corners android:radius="100dp" />
    <solid android:color="@android:color/tab_indicator_text" />
</shape>
```

6.72 rectangle.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <stroke
        android:width="@dimen/rectangle_stroke_width"
        android:color="?attr/colorPrimary" />

    <corners android:radius="@dimen/rectangle_corner_radius" />

    <gradient
        android:angle="90"
        android:centerColor="?attr/card_background_color"
        android:endColor="?attr/card_background_gradient"
        android:startColor="?attr/card_background_gradient" />
</shape>
```

6.73 rectangle_black_transparent.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <solid android:color="@color/color_popup_background" />

    <corners android:radius="@dimen/rectangle_corner_radius" />
</shape>
```

6.74 rectangle_cancel_popup.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:shape="rectangle">

    <gradient
        android:angle="90"
        android:centerColor="@color/color_cancel_popup_background"
        android:endColor="@color/color_cancel_popup_background_gradient"
        android:startColor="@color/color_cancel_popup_background_gradient" />

    <corners android:radius="@dimen/rectangle_corner_radius" />

    <stroke
        android:width="@dimen/rectangle_stroke_width"
        android:color="@color/color_popup_cancel_text" />
</shape>
```

6.75 fragment_introduction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.introduction.IntroductionFragment">

    <ImageView
        android:id="@+id/introduction_imageView"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="@dimen/activity_horizontal_margin"
        android:contentDescription="@string/content_description_introduction_image"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/introduction_scrollView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintWidth_percent=".5" />

    <ScrollView
        android:id="@+id/introduction_scrollView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@id/introduction_imageView"
        app:layout_constraintTop_toTopOf="parent">

        <TextView
            android:id="@+id/introduction_textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:justificationMode="inter_word"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

6.76 fragment_introduction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.introduction.IntroductionFragment">

    <ImageView
        android:id="@+id/introduction_imageView"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="@dimen/activity_horizontal_margin"
        android:contentDescription="@string/content_description_introduction_image"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/introduction_scrollView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintWidth_percent=".5" />

    <ScrollView
```

```
        android:id="@+id/introduction_scrollView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@id/introduction_imageView"
        app:layout_constraintTop_toTopOf="parent">

        <TextView
            android:id="@+id/introduction_textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

6.77 fragment_introduction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.introduction.IntroductionFragment">

    <ImageView
        android:id="@+id/introduction_imageView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_margin="@dimen/activity_horizontal_margin"
        android:contentDescription="@string/content_description_introduction_image"
        app:layout_constraintBottom_toTopOf="@id/introduction_scrollView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_weight="1" />

    <ScrollView
        android:id="@+id/introduction_scrollView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/introduction_imageView"
        app:layout_constraintVertical_weight="2">

        <TextView
            android:id="@+id/introduction_textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:justificationMode="inter_word"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

6.78 fragment_introduction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.introduction.IntroductionFragment">

    <ImageView
        android:id="@+id/introduction_imageView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_margin="@dimen/activity_horizontal_margin"
        android:contentDescription="@string/content_description_introduction_image"
        app:layout_constraintBottom_toTopOf="@id/introduction_scrollView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_weight="1" />

    <ScrollView
        android:id="@+id/introduction_scrollView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/introduction_imageView"
        app:layout_constraintVertical_weight="2">

        <TextView
            android:id="@+id/introduction_textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

6.79 activity_rules.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/rulesToolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        app:layout_constraintTop_toTopOf="parent" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginBottom="@dimen/margin_row_buttons"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@id/rulesToolbar">

        <TextView
            android:id="@+id/rulesTextView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:justificationMode="inter_word"
            android:textColor="@android:color/secondary_text_dark"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.80 activity_rules.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/rulesToolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        app:layout_constraintTop_toTopOf="parent" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginBottom="@dimen/margin_row_buttons"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@id/rulesToolbar">

        <TextView
            android:id="@+id/rulesTextView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/activity_horizontal_margin"
            android:textColor="@android:color/secondary_text_dark"
            android:textSize="@dimen/rules_textSize" />
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.81 activity_introduction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".ui.introduction.IntroductionActivity">

    <androidx.viewpager2.widget.ViewPager2
        android:id="@+id/introduction_viewPager"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginTop="@dimen/activity_vertical_margin"
        app:layout_behavior="@string/appbar_scrolling_view_behavior"
        app:layout_constraintBottom_toTopOf="@id/introduction_bottomLayout"
        app:layout_constraintTop_toTopOf="parent" />

    <View
        android:layout_width="match_parent"
        android:layout_height="@dimen/divider_strength"
        android:alpha="0.12"
        android:background="@android:color/white"
        app:layout_constraintBottom_toTopOf="@id/introduction_bottomLayout" />

    <FrameLayout
        android:id="@+id/introduction_bottomLayout"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:layout_marginStart="@dimen/activity_horizontal_margin"
        android:layout_marginEnd="@dimen/activity_horizontal_margin"
        app:layout_constraintBottom_toBottomOf="parent">

        <Button
            android:id="@+id/introduction_button_skip"
            style="@style/Widget.AppCompat.Button.Borderless"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="start|center_vertical"
            android:text="@string/introduction_button_skip" />

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/introduction_indicator_0"
                android:layout_width="@dimen/indicator_radius"
                android:layout_height="@dimen/indicator_radius"
                android:layout_marginEnd="@dimen/border_margin"
                app:srcCompat="@drawable/indicator_selected"
                android:contentDescription="@string/content_description_introduction_indicator" />

            <ImageView
                android:id="@+id/introduction_indicator_1"
                android:layout_width="@dimen/indicator_radius"
                android:layout_height="@dimen/indicator_radius"
                android:layout_marginEnd="@dimen/border_margin"
                app:srcCompat="@drawable/indicator_unselected"
                android:contentDescription="@string/content_description_introduction_indicator" />

            <ImageView
                android:id="@+id/introduction_indicator_2"
                android:layout_width="@dimen/indicator_radius"
                android:layout_height="@dimen/indicator_radius"
                android:layout_marginEnd="@dimen/border_margin"
                app:srcCompat="@drawable/indicator_unselected"
                android:contentDescription="@string/content_description_introduction_indicator" />

            <ImageView
                android:id="@+id/introduction_indicator_3"
                android:layout_width="@dimen/indicator_radius"
                android:layout_height="@dimen/indicator_radius"
                android:layout_marginEnd="@dimen/border_margin"
                app:srcCompat="@drawable/indicator_unselected"
                android:contentDescription="@string/content_description_introduction_indicator" />

            <ImageView
                android:id="@+id/introduction_indicator_4"
                android:layout_width="@dimen/indicator_radius"
                android:layout_height="@dimen/indicator_radius"
                app:srcCompat="@drawable/indicator_unselected"
                android:contentDescription="@string/content_description_introduction_indicator" />
        </LinearLayout>

        <Button
            android:id="@+id/introduction_button_finish"
            style="@style/Widget.AppCompat.Button.Borderless"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="end|center_vertical"
            android:text="@string/introduction_button_finish"
            android:visibility="gone" />

        <ImageButton
            android:id="@+id/introduction_button_next"
            style="@style/Widget.AppCompat.Button.Borderless"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="end|center_vertical"
            android:contentDescription="@string/content_description_introduction_next"
            android:padding="@dimen/activity_horizontal_margin"
            app:srcCompat="@drawable/icon_next" />
    </FrameLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

6.82 activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/background_drawable"
    tools:context=".ui.main.MainActivity">

    <ImageView
        android:id="@+id/backgroundImageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:contentDescription="@string/content_description_activity_main_background"
        android:scaleType="centerCrop"
        android:visibility="gone" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:baselineAligned="false"
        android:orientation="horizontal"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <include
            android:id="@+id/firstRow"
            layout="@layout/row"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginVertical="@dimen/activity_vertical_margin"
            android:layout_weight="1" />

        <include
            android:id="@+id/secondRow"
            layout="@layout/row"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginVertical="@dimen/activity_vertical_margin"
            android:layout_weight="1" />

        <include
            android:id="@+id/thirdRow"
            layout="@layout/row"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginVertical="@dimen/activity_vertical_margin"
            android:layout_marginStart="@dimen/border_margin"
            android:layout_marginEnd="@dimen/margin_row_buttons"
            android:layout_weight="1" />

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_gravity="center"
            android:layout_marginVertical="@dimen/activity_vertical_margin"
            android:layout_marginStart="@dimen/margin_row_buttons"
            android:layout_marginEnd="@dimen/border_margin"
            android:layout_weight="1">

            <androidx.constraintlayout.widget.ConstraintLayout
                android:id="@+id/overallPointLayout"
                android:layout_width="match_parent"
                android:layout_height="0dp"
                android:layout_margin="@dimen/margin_menu_buttons"
                app:layout_constraintBottom_toTopOf="@id/menuView"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_weight="1">

                <ImageView
                    android:id="@+id/overallPointBall"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:background="@null"
                    android:contentDescription="@string/content_description_activity_main_point_ball"
                    android:scaleType="fitCenter"
                    android:src="?attr/point_ball_mipmap"
                    android:translationZ="5dp" />

                <TextView
                    android:id="@+id/overallPointView"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:textColor="@color/circle_text"
                    android:textSize="@dimen/overall_point_circle_textSize"
                    android:translationZ="10dp"
                    app:layout_constraintBottom_toBottomOf="parent"
                    app:layout_constraintEnd_toEndOf="parent"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintTop_toTopOf="parent" />
            </androidx.constraintlayout.widget.ConstraintLayout>

            <androidx.constraintlayout.widget.ConstraintLayout
                android:id="@+id/menuView"
                android:layout_width="match_parent"
                android:layout_height="0dp"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintTop_toBottomOf="@id/overallPointLayout"
                app:layout_constraintVertical_weight="3">

                <ImageButton
                    android:id="@+id/factionButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
android:contentDescription="@string/content_description_activity_main_faction_button"
                    android:scaleType="fitCenter"
                    android:src="?android:attr/alertDialogIcon"
                    app:layout_constraintBottom_toTopOf="@id/weatherButton"
                    app:layout_constraintEnd_toStartOf="@id/resetButton"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintTop_toTopOf="parent" />

                <ImageButton
                    android:id="@+id/resetButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
                    android:clickable="false"
android:contentDescription="@string/content_description_activity_main_reset_button"
                    android:scaleType="fitCenter"
                    android:src="@drawable/icon_reset_grey"
                    app:layout_constraintBottom_toTopOf="@id/burnButton"
                    app:layout_constraintEnd_toEndOf="parent"
                    app:layout_constraintStart_toEndOf="@id/factionButton"
                    app:layout_constraintTop_toTopOf="parent" />

                <ImageButton
                    android:id="@+id/weatherButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
                    android:clickable="false"
                    android:contentDescription="@string/content_description_activity_main_weather_button"
                    android:scaleType="fitCenter"
                    android:src="@drawable/icon_weather_grey"
                    app:layout_constraintBottom_toTopOf="@id/coinButton"
                    app:layout_constraintEnd_toStartOf="@id/burnButton"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintTop_toBottomOf="@id/factionButton" />

                <ImageButton
                    android:id="@+id/burnButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
                    android:clickable="false"
                    android:contentDescription="@string/content_description_activity_main_burn_button"
                    android:scaleType="fitCenter"
                    android:src="@drawable/icon_burn_grey"
                    app:layout_constraintBottom_toTopOf="@id/settingsButton"
                    app:layout_constraintEnd_toEndOf="parent"
                    app:layout_constraintStart_toEndOf="@+id/weatherButton"
                    app:layout_constraintTop_toBottomOf="@id/resetButton" />

                <ImageButton
                    android:id="@+id/coinButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
                    android:contentDescription="@string/content_description_activity_main_coin_button"
                    android:scaleType="fitCenter"
                    android:src="@drawable/icon_coin"
                    app:layout_constraintBottom_toBottomOf="parent"
                    app:layout_constraintEnd_toStartOf="@id/settingsButton"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintTop_toBottomOf="@id/weatherButton" />

                <ImageButton
                    android:id="@+id/settingsButton"
                    android:layout_width="0dp"
                    android:layout_height="0dp"
                    android:layout_margin="@dimen/margin_menu_buttons"
                    android:background="@null"
                    android:contentDescription="@string/content_description_activity_main_settings_button"
                    android:scaleType="fitCenter"
                    android:src="@drawable/icon_settings"
                    app:layout_constraintBottom_toBottomOf="parent"
                    app:layout_constraintEnd_toEndOf="parent"
                    app:layout_constraintStart_toEndOf="@+id/coinButton"
                    app:layout_constraintTop_toBottomOf="@id/burnButton" />
            </androidx.constraintlayout.widget.ConstraintLayout>
        </androidx.constraintlayout.widget.ConstraintLayout>
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.83 activity_settings.xml

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/settingsToolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        app:layout_constraintBottom_toTopOf="@id/settingsFrameLayout"
        app:layout_constraintTop_toTopOf="parent"
        app:title="@string/settings_title" />

    <FrameLayout
        android:id="@+id/settingsFrameLayout"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@id/settingsToolbar" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.84 alertdialog_checkbox.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <CheckBox
        android:id="@+id/alertDialog_checkbox"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="@dimen/alertDialog_checkBox_marginStart"
        android:layout_marginEnd="@dimen/alertDialog_checkBox_marginEnd"
        android:checked="true"
        android:layoutDirection="rtl"
        android:minHeight="@dimen/alertDialog_checkBox_minHeight"
        android:text="@string/alertDialog_reset_checkbox" />
</FrameLayout>
```

6.85 card.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="@dimen/popUp_card_width"
    android:layout_height="match_parent"
    android:layout_margin="@dimen/border_margin"
    android:background="@drawable/rectangle">

    <TextView
        android:id="@+id/damageView"
        android:layout_width="@dimen/popUp_card_image_diameter"
        android:layout_height="@dimen/popUp_card_image_diameter"
        android:background="@drawable/icon_damage_background"
        android:gravity="center"
        android:textColor="@color/color_damage_textColor"
        android:textSize="@dimen/popUp_card_textSize"
        android:textStyle="bold"
        app:layout_constraintBottom_toTopOf="@+id/abilityView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/abilityView"
        android:layout_width="@dimen/popUp_card_image_diameter"
        android:layout_height="@dimen/popUp_card_image_diameter"
        android:contentDescription="@string/content_description_card_ability"
        android:visibility="gone"
        app:layout_constraintBottom_toTopOf="@id/copyButton"
        app:layout_constraintEnd_toStartOf="@id/bindingView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/damageView"
        app:srcCompat="@drawable/icon_binding" />

    <TextView
        android:id="@+id/bindingView"
        android:layout_width="@dimen/popUp_card_image_diameter"
        android:layout_height="@dimen/popUp_card_image_diameter"
        android:background="@drawable/icon_damage_background"
        android:gravity="center"
        android:textColor="@color/color_damage_textColor"
        android:textSize="@dimen/popUp_card_textSize"
        android:textStyle="bold"
        android:visibility="gone"
        app:layout_constraintBottom_toBottomOf="@id/abilityView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@id/abilityView"
        app:layout_constraintTop_toTopOf="@+id/abilityView" />

    <ImageButton
        android:id="@+id/copyButton"
        android:layout_width="@dimen/popUp_card_image_diameter"
        android:layout_height="@dimen/popUp_card_image_diameter"
        android:background="@null"
        android:contentDescription="@string/content_description_card_copy"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/deleteButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/abilityView"
        app:srcCompat="@drawable/icon_copy" />

    <ImageButton
        android:id="@+id/deleteButton"
        android:layout_width="@dimen/popUp_card_image_diameter"
        android:layout_height="@dimen/popUp_card_image_diameter"
        android:background="@null"
        android:contentDescription="@string/content_description_card_delete"
        app:layout_constraintBottom_toBottomOf="@id/copyButton"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@id/copyButton"
        app:layout_constraintTop_toTopOf="@id/copyButton"
        app:srcCompat="@drawable/icon_delete" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.86 popup_add_card.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/card_layout"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="@dimen/border_margin"
        android:layout_marginTop="@dimen/border_margin"
        android:background="@drawable/rectangle"
        app:layout_constraintBottom_toTopOf="@id/button_layout"
        app:layout_constraintEnd_toEndOf="@id/button_layout"
        app:layout_constraintHeight_max="@dimen/popUp_add_card_max_card_height"
        app:layout_constraintStart_toStartOf="@id/button_layout"
        app:layout_constraintTop_toTopOf="parent">

        <TextView
            android:id="@+id/popup_add_card_epic_picker_label"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="@dimen/popUp_add_card_label_marginBottom"
            android:gravity="center"
            android:labelFor="@id/popup_add_card_epic_picker"
            android:text="@string/popUp_add_card_label_epic"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_add_card_label_textSize"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="@id/popup_add_card_epic_picker"
            app:layout_constraintStart_toStartOf="@id/popup_add_card_epic_picker"
            app:layout_constraintTop_toBottomOf="@id/popup_add_card_epic_picker" />

        <NumberPicker
            android:id="@+id/popup_add_card_epic_picker"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginTop="@dimen/border_margin"
            android:descendantFocusability="blocksDescendants"
            app:layout_constraintBottom_toTopOf="@id/popup_add_card_epic_picker_label"
            app:layout_constraintEnd_toStartOf="@id/popup_add_card_dmg_picker"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/popup_add_card_dmg_picker_label"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="@dimen/popUp_add_card_label_marginBottom"
            android:gravity="center"
            android:labelFor="@id/popup_add_card_dmg_picker"
            android:text="@string/popUp_add_card_label_dmg"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_add_card_label_textSize"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="@id/popup_add_card_dmg_picker"
            app:layout_constraintStart_toStartOf="@id/popup_add_card_dmg_picker"
            app:layout_constraintTop_toBottomOf="@id/popup_add_card_dmg_picker" />

        <NumberPicker
            android:id="@+id/popup_add_card_dmg_picker"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginTop="@dimen/border_margin"
            android:descendantFocusability="blocksDescendants"
            app:layout_constraintBottom_toTopOf="@id/popup_add_card_dmg_picker_label"
            app:layout_constraintEnd_toStartOf="@id/popup_add_card_ability_picker"
            app:layout_constraintStart_toEndOf="@id/popup_add_card_epic_picker"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/popup_add_card_ability_picker_label"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="@dimen/popUp_add_card_label_marginBottom"
            android:gravity="center"
            android:labelFor="@id/popup_add_card_ability_picker"
            android:text="@string/popUp_add_card_label_ability"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_add_card_label_textSize"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="@id/popup_add_card_ability_picker"
            app:layout_constraintStart_toStartOf="@id/popup_add_card_ability_picker"
            app:layout_constraintTop_toBottomOf="@id/popup_add_card_ability_picker" />

        <NumberPicker
            android:id="@+id/popup_add_card_ability_picker"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginTop="@dimen/border_margin"
            android:descendantFocusability="blocksDescendants"
            app:layout_constraintBottom_toTopOf="@id/popup_add_card_ability_picker_label"
            app:layout_constraintEnd_toStartOf="@id/popup_add_card_binding_picker"
            app:layout_constraintStart_toEndOf="@id/popup_add_card_dmg_picker"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/popup_add_card_binding_picker_label"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="@dimen/popUp_add_card_label_marginBottom"
            android:gravity="center"
            android:labelFor="@id/popup_add_card_binding_picker"
            android:text="@string/popUp_add_card_label_binding"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_add_card_label_textSize"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="@id/popup_add_card_binding_picker"
            app:layout_constraintStart_toStartOf="@id/popup_add_card_binding_picker"
            app:layout_constraintTop_toBottomOf="@id/popup_add_card_binding_picker" />

        <NumberPicker
            android:id="@+id/popup_add_card_binding_picker"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginTop="@dimen/border_margin"
            android:descendantFocusability="blocksDescendants"
            android:visibility="gone"
            app:layout_constraintBottom_toTopOf="@id/popup_add_card_binding_picker_label"
            app:layout_constraintEnd_toStartOf="@id/popup_add_card_number_picker"
            app:layout_constraintStart_toEndOf="@id/popup_add_card_ability_picker"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/popup_add_card_number_picker_label"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="@dimen/popUp_add_card_label_marginBottom"
            android:gravity="center"
            android:labelFor="@id/popup_add_card_number_picker"
            android:text="@string/popUp_add_card_label_number"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_add_card_label_textSize"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="@id/popup_add_card_number_picker"
            app:layout_constraintStart_toStartOf="@id/popup_add_card_number_picker"
            app:layout_constraintTop_toBottomOf="@id/popup_add_card_number_picker" />

        <NumberPicker
            android:id="@+id/popup_add_card_number_picker"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="@dimen/border_margin"
            android:layout_marginTop="@dimen/border_margin"
            android:descendantFocusability="blocksDescendants"
            app:layout_constraintBottom_toTopOf="@id/popup_add_card_number_picker_label"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/popup_add_card_binding_picker"
            app:layout_constraintTop_toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/button_layout"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_margin="@dimen/border_margin"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_max="@dimen/popUp_card_button_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/card_layout">

        <Button
            android:id="@+id/popup_add_card_cancel_button"
            android:layout_width="@dimen/popUp_card_button_width"
            android:layout_height="match_parent"
            android:layout_marginEnd="@dimen/popUp_card_button_margin"
            android:background="@drawable/rectangle_cancel_popup"
            android:text="@string/alertDialog_cancel"
            android:textAllCaps="false"
            android:textColor="@color/color_popup_cancel_text"
            android:textSize="@dimen/popUp_card_textSize"
            app:layout_constraintEnd_toStartOf="@id/popup_add_card_save_button"
            app:layout_constraintHeight_max="@dimen/popUp_card_button_height"
            app:layout_constraintStart_toStartOf="parent" />

        <Button
            android:id="@+id/popup_add_card_save_button"
            android:layout_width="@dimen/popUp_card_button_width"
            android:layout_height="match_parent"
            android:layout_marginStart="@dimen/popUp_card_button_margin"
            android:background="@drawable/rectangle"
            android:text="@string/popUp_cards_save"
            android:textAllCaps="false"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_card_textSize"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHeight_max="@dimen/popUp_card_button_height"
            app:layout_constraintStart_toEndOf="@id/popup_add_card_cancel_button" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.87 popup_cards.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/cards_list"
        android:layout_width="0dp"
        android:layout_height="@dimen/popUp_card_list_height"
        android:layout_margin="@dimen/activity_horizontal_margin"
        app:layout_constraintBottom_toTopOf="@id/button_layout"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/button_layout"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_margin="@dimen/border_margin"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_max="@dimen/popUp_card_button_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/cards_list">

        <Button
            android:id="@+id/popup_cards_cancel_button"
            android:layout_width="@dimen/popUp_card_button_width"
            android:layout_height="match_parent"
            android:layout_marginEnd="@dimen/popUp_card_button_margin"
            android:background="@drawable/rectangle_cancel_popup"
            android:text="@string/alertDialog_cancel"
            android:textAllCaps="false"
            android:textColor="@color/color_popup_cancel_text"
            android:textSize="@dimen/popUp_card_textSize"
            app:layout_constraintEnd_toStartOf="@id/popup_cards_add_button"
            app:layout_constraintStart_toStartOf="parent" />

        <Button
            android:id="@+id/popup_cards_add_button"
            android:layout_width="@dimen/popUp_card_button_width"
            android:layout_height="match_parent"
            android:layout_marginStart="@dimen/popUp_card_button_margin"
            android:background="@drawable/rectangle"
            android:text="@string/popUp_cards_add"
            android:textAllCaps="false"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/popUp_card_textSize"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/popup_cards_cancel_button" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.88 popup_coin_normal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/coinflipBackground"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/rectangle_black_transparent"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <ImageView
            android:id="@+id/popup_coin_normal_coinView"
            android:layout_width="@dimen/popUp_coin_diameter"
            android:layout_height="@dimen/popUp_coin_diameter"
            android:layout_gravity="center"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:contentDescription="@string/content_description_coinpopup_coin"
            android:src="@drawable/coin_win" />

        <TextView
            android:id="@+id/popup_coin_normal_textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:text="@string/popUp_coin_normal_win"
            android:textColor="@color/color_coin_text"
            android:textSize="@dimen/popUp_coin_title_textSize" />
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.89 popup_coin_stewie.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/coinflipBackground"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_margin="@dimen/activity_vertical_margin"
        android:background="@drawable/rectangle_black_transparent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_max="@dimen/popUp_coin_window_max_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:id="@+id/imageLayout"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:orientation="vertical"
            app:layout_constraintEnd_toStartOf="@id/textLayout"
            app:layout_constraintStart_toStartOf="parent">

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_stewie_1"
                android:src="@drawable/stewie_1" />

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_stewie_2"
                android:src="@drawable/stewie_2" />

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_stewie_3"
                android:src="@drawable/stewie_3" />
        </LinearLayout>

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/textLayout"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_horizontal_margin"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/imageLayout"
            app:layout_constraintWidth_max="@dimen/popUp_coin_textview_max_width">

            <TextView
                android:id="@+id/text1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_stewie_1"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_title_textSize"
                app:layout_constraintTop_toTopOf="parent" />

            <TextView
                android:id="@+id/text2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_stewie_2"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_dialog_textSize"
                app:layout_constraintTop_toBottomOf="@id/text1" />

            <TextView
                android:id="@+id/text3"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_stewie_3"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_dialog_textSize"
                app:layout_constraintBottom_toTopOf="@id/text4"
                app:layout_constraintTop_toBottomOf="@id/text2" />

            <TextView
                android:id="@+id/text4"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_stewie_4"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_dialog_textSize"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintTop_toBottomOf="@id/text3" />
        </androidx.constraintlayout.widget.ConstraintLayout>
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.90 popup_coin_terry.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/coinflipBackground"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_margin="@dimen/activity_vertical_margin"
        android:background="@drawable/rectangle_black_transparent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_max="@dimen/popUp_coin_window_max_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:id="@+id/imageLayout"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:orientation="vertical"
            app:layout_constraintEnd_toStartOf="@id/textLayout"
            app:layout_constraintStart_toStartOf="parent">

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_terry_1"
                android:src="@drawable/terry_1" />

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_terry_2"
                android:src="@drawable/terry_2" />

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="0dp"
                android:layout_weight="1"
                android:contentDescription="@string/content_description_coinpopup_terry_3"
                android:src="@drawable/terry_3" />
        </LinearLayout>

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/textLayout"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_horizontal_margin"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/imageLayout"
            app:layout_constraintWidth_max="@dimen/popUp_coin_textview_max_width">

            <TextView
                android:id="@+id/text1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_terry_1"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_title_textSize"
                app:layout_constraintBottom_toTopOf="@id/text2"
                app:layout_constraintTop_toTopOf="parent" />

            <TextView
                android:id="@+id/text2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_terry_2"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_dialog_textSize"
                app:layout_constraintBottom_toTopOf="@id/text3"
                app:layout_constraintTop_toBottomOf="@id/text1" />

            <TextView
                android:id="@+id/text3"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="@dimen/border_margin"
                android:gravity="center"
                android:text="@string/popUp_coin_terry_3"
                android:textColor="@color/color_coin_text"
                android:textSize="@dimen/popUp_coin_dialog_textSize"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintTop_toBottomOf="@id/text2" />
        </androidx.constraintlayout.widget.ConstraintLayout>
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.91 popup_coin_vin.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/coinflipBackground"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_margin="@dimen/activity_vertical_margin"
        android:background="@drawable/rectangle_black_transparent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_max="@dimen/popUp_coin_window_max_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <ImageView
            android:id="@+id/imageView"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_vertical_margin"
            android:contentDescription="@string/content_description_coinpopup_vin"
            android:src="@drawable/vin"
            app:layout_constraintEnd_toStartOf="@id/textView"
            app:layout_constraintStart_toStartOf="parent" />

        <TextView
            android:id="@+id/textView"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:gravity="center"
            android:text="@string/popUp_coin_vin"
            android:textColor="@color/color_coin_text"
            android:textSize="@dimen/popUp_coin_title_textSize"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/imageView"
            app:layout_constraintWidth_max="@dimen/popUp_coin_textview_max_width" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.92 popup_faction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/factionBackground"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <ImageView
        android:id="@+id/monsterCardView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/border_margin"
        android:contentDescription="@string/content_description_factionpopup_monster"
        android:src="@drawable/card_monster"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="@id/monsterButton"
        app:layout_constraintStart_toStartOf="@id/monsterButton" />

    <Button
        android:id="@+id/monsterButton"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/popUp_faction_margin"
        android:background="@drawable/rectangle"
        android:text="@string/faction_monster"
        android:textAllCaps="false"
        android:textColor="@color/color_text_monster"
        android:textSize="@dimen/rectangle_textSize"
        android:theme="@style/MonsterTheme"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/nilfgaardButton"
        app:layout_constraintHeight_min="@dimen/rectangle_height"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/monsterCardView"
        app:layout_constraintWidth_max="@dimen/popUp_faction_button_max_width" />

    <ImageView
        android:id="@+id/nilfgaardCardView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/border_margin"
        android:contentDescription="@string/content_description_factionpopup_nilfgaard"
        android:src="@drawable/card_nilfgaard"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintEnd_toEndOf="@id/nilfgaardButton"
        app:layout_constraintStart_toStartOf="@id/nilfgaardButton" />

    <Button
        android:id="@+id/nilfgaardButton"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/popUp_faction_margin"
        android:background="@drawable/rectangle"
        android:text="@string/faction_nilfgaard"
        android:textAllCaps="false"
        android:textColor="@color/color_text_nilfgaard"
        android:textSize="@dimen/rectangle_textSize"
        android:theme="@style/NilfgaardTheme"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/northernKingdomsButton"
        app:layout_constraintHeight_min="@dimen/rectangle_height"
        app:layout_constraintStart_toEndOf="@id/monsterButton"
        app:layout_constraintTop_toBottomOf="@id/nilfgaardCardView"
        app:layout_constraintWidth_max="@dimen/popUp_faction_button_max_width" />

    <ImageView
        android:id="@+id/northernKingdomsCardView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/border_margin"
        android:contentDescription="@string/content_description_factionpopup_northern_kingdoms"
        android:src="@drawable/card_northern_kingdoms"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="@id/northernKingdomsButton"
        app:layout_constraintStart_toStartOf="@id/northernKingdomsButton" />

    <Button
        android:id="@+id/northernKingdomsButton"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/popUp_faction_margin"
        android:background="@drawable/rectangle"
        android:text="@string/faction_northern_kingdoms"
        android:textAllCaps="false"
        android:textColor="@color/color_text_northern_kingdoms"
        android:textSize="@dimen/rectangle_textSize"
        android:theme="@style/NorthernKingdomsTheme"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@id/scoiataelButton"
        app:layout_constraintHeight_min="@dimen/rectangle_height"
        app:layout_constraintStart_toEndOf="@id/nilfgaardButton"
        app:layout_constraintTop_toBottomOf="@id/northernKingdomsCardView"
        app:layout_constraintWidth_max="@dimen/popUp_faction_button_max_width" />

    <ImageView
        android:id="@+id/scoiataelCardView"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/border_margin"
        android:contentDescription="@string/content_description_factionpopup_scoiatael"
        android:src="@drawable/card_scoiatael"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="@id/scoiataelButton"
        app:layout_constraintStart_toStartOf="@id/scoiataelButton" />

    <Button
        android:id="@+id/scoiataelButton"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/popUp_faction_margin"
        android:background="@drawable/rectangle"
        android:text="@string/faction_scoiatael"
        android:textAllCaps="false"
        android:textColor="@color/color_text_scoiatael"
        android:textSize="@dimen/rectangle_textSize"
        android:theme="@style/ScoiataelTheme"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHeight_min="@dimen/rectangle_height"
        app:layout_constraintStart_toEndOf="@id/northernKingdomsButton"
        app:layout_constraintTop_toBottomOf="@id/scoiataelCardView"
        app:layout_constraintWidth_max="@dimen/popUp_faction_button_max_width" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.93 row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="@dimen/point_circle_diameter"
        android:layout_height="@dimen/point_circle_diameter"
        android:translationZ="5dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <ImageView
            android:id="@+id/pointBall"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="@null"
            android:contentDescription="@string/content_description_row_pointball"
            android:scaleType="fitXY"
            android:src="?attr/point_ball_mipmap"
            android:translationZ="5dp" />

        <TextView
            android:id="@+id/pointView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:gravity="center"
            android:textColor="@color/circle_text"
            android:textSize="@dimen/point_circle_textSize"
            android:translationZ="10dp" />
    </androidx.constraintlayout.widget.ConstraintLayout>

    <ImageView
        android:id="@+id/weatherView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginTop="@dimen/point_circle_diameter_div2"
        android:contentDescription="@string/content_description_row_weather"
        android:scaleType="centerCrop"
        app:layout_constraintBottom_toTopOf="@id/hornView"
        app:layout_constraintDimensionRatio="1048:674"
        app:layout_constraintEnd_toEndOf="@id/cardView"
        app:layout_constraintStart_toStartOf="@id/cardView"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/good_weather" />

    <ImageView
        android:id="@+id/hornView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:contentDescription="@string/content_description_row_horn"
        android:scaleType="centerCrop"
        app:layout_constraintBottom_toTopOf="@id/cardView"
        app:layout_constraintDimensionRatio="1048:674"
        app:layout_constraintEnd_toEndOf="@id/cardView"
        app:layout_constraintStart_toStartOf="@id/cardView"
        app:layout_constraintTop_toBottomOf="@id/weatherView"
        app:srcCompat="@drawable/horn_grey" />

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/cardView"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/hornView">

        <ImageView
            android:id="@+id/cardsImage"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:contentDescription="@string/content_description_row_card"
            android:src="?attr/card_view_mipmap"
            android:translationZ="5dp" />

        <TextView
            android:id="@+id/cardCountView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="?attr/card_count_text_color"
            android:textSize="@dimen/card_count_textSize"
            android:translationZ="10dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

6.94 icon_launcher.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
    <background android:drawable="@color/icon_launcher_background" />
    <foreground android:drawable="@mipmap/icon_launcher_foreground" />
    <monochrome android:drawable="@mipmap/icon_launcher_monochrome" />
</adaptive-icon>
```

6.95 icon_launcher_round.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
    <background android:drawable="@color/icon_launcher_background" />
    <foreground android:drawable="@mipmap/icon_launcher_foreground" />
    <monochrome android:drawable="@mipmap/icon_launcher_monochrome" />
</adaptive-icon>
```

6.96 arrays.xml

```
<resources>
    <string-array name="design_names">
        <item>@string/array_design_default</item>
        <item>@string/array_design_geralt</item>
        <item>@string/array_design_ciri</item>
        <item>@string/array_design_jaskier</item>
        <item>@string/array_design_yennefer</item>
        <item>@string/array_design_eredin</item>
    </string-array>

    <string-array name="design_values">
        <item>0</item>
        <item>1</item>
        <item>2</item>
        <item>3</item>
        <item>4</item>
        <item>5</item>
    </string-array>
</resources>
```

6.97 attr.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <attr name="card_count_text_color" format="color" />

    <attr name="card_view_mipmap" format="reference" />

    <attr name="point_ball_mipmap" format="reference" />

    <attr name="card_background_color" format="reference" />

    <attr name="card_background_gradient" format="reference" />

    <attr name="card_view_button_background" format="reference" />

    <declare-styleable name="theme">
        <attr name="point_ball_mipmap" />
        <attr name="card_view_mipmap" />
        <attr name="colorPrimary" />
        <attr name="android:alertDialogIcon" />
    </declare-styleable>
</resources>
```

6.98 colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="circle_text">#000000</color>

    <color name="color_popup_background">#BF000000</color>
    <color name="color_popup_cancel_text">#FFFFFF</color>
    <color name="color_coin_text">#FFFFFF</color>
    <color name="color_cancel_popup_background">#505050</color>
    <color name="color_cancel_popup_background_gradient">#000000</color>
    <color name="color_damage_textColor">#2F1700</color>
    <color name="color_damage_textColor_buffed">#10A010</color>
    <color name="color_damage_textColor_debuffed">#E42020</color>

    <color name="color_text_scoiatael">#CAC559</color>
    <color name="color_background_scoiatael">#339300</color>
    <color name="color_gradient_scoiatael">#164000</color>
    <color name="color_accent_scoiatael">#52EC00</color>
    <color name="color_card_view_button_background_scoiatael">#40B900</color>

    <color name="color_text_monster">#B6D1DB</color>
    <color name="color_background_monster">#99030E</color>
    <color name="color_gradient_monster">#430008</color>
    <color name="color_accent_monster">#FF1F30</color>
    <color name="color_card_view_button_background_monster">#C80917</color>

    <color name="color_text_northern_kingdoms">#D9D9D9</color>
    <color name="color_background_northern_kingdoms">#005ADF</color>
    <color name="color_gradient_northern_kingdoms">#002E71</color>
    <color name="color_accent_northern_kingdoms">#3285FF</color>
    <color name="color_card_view_button_background_northern_kingdoms">#003B92</color>

    <color name="color_text_nilfgaard">#EFC561</color>
    <color name="color_background_nilfgaard">#424242</color>
    <color name="color_gradient_nilfgaard">#0C0C0C</color>
    <color name="color_accent_nilfgaard">#CCCCCC</color>
    <color name="color_card_view_button_background_nilfgaard">#787878</color>
</resources>
```

6.99 dimens.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="point_circle_diameter">80dp</dimen>
    <dimen name="point_circle_diameter_div2">40dp</dimen>
    <dimen name="point_circle_textSize">40sp</dimen>
    <dimen name="overall_point_circle_textSize">45sp</dimen>
    <dimen name="card_count_textSize">40sp</dimen>

    <dimen name="border_margin">8dp</dimen>
    <dimen name="margin_row_buttons">16dp</dimen>
    <dimen name="margin_menu_buttons">12dp</dimen>

    <dimen name="rectangle_height">65dp</dimen>
    <dimen name="rectangle_stroke_width">4dp</dimen>
    <dimen name="rectangle_corner_radius">15dp</dimen>
    <dimen name="rectangle_textSize">20sp</dimen>

    <dimen name="popUp_faction_button_max_width">200dp</dimen>
    <dimen name="popUp_faction_margin">24dp</dimen>

    <dimen name="popUp_card_list_height">240dp</dimen>
    <dimen name="popUp_card_width">160dp</dimen>
    <dimen name="popUp_card_textSize">22sp</dimen>
    <dimen name="popUp_card_button_width">200dp</dimen>
    <dimen name="popUp_card_button_height">80dp</dimen>
    <dimen name="popUp_card_image_diameter">50dp</dimen>
    <dimen name="popUp_card_button_margin">75dp</dimen>

    <dimen name="popUp_add_card_label_textSize">21sp</dimen>
    <dimen name="popUp_add_card_label_marginBottom">24dp</dimen>
    <dimen name="popUp_add_card_max_card_height">300dp</dimen>

    <dimen name="popUp_coin_textview_max_width">275dp</dimen>
    <dimen name="popUp_coin_diameter">90dp</dimen>
    <dimen name="popUp_coin_window_max_height">400dp</dimen>
    <dimen name="popUp_coin_title_textSize">24sp</dimen>
    <dimen name="popUp_coin_dialog_textSize">18sp</dimen>

    <dimen name="alertDialog_checkBox_marginStart">40dp</dimen>
    <dimen name="alertDialog_checkBox_marginEnd">25dp</dimen>
    <dimen name="alertDialog_checkBox_minHeight">48dp</dimen>

    <dimen name="rules_textSize">20sp</dimen>

    <dimen name="divider_strength">2dp</dimen>
    <dimen name="indicator_radius">8dp</dimen>

    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
</resources>
```

6.100 icon_launcher_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="icon_launcher_background">#FFFFFF</color>
</resources>
```

6.101 preferences.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="preference_key_design" translatable="false">design</string>
    <string name="design_preference_default" translatable="false">0</string>
    <bool name="warning_preference_default">true</bool>
    <string name="preference_key_warning" translatable="false">warnings</string>
    <bool name="faction_reset_preference_default">false</bool>
    <string name="preference_key_faction_reset" translatable="false">factionReset</string>
    <string name="preference_key_introduction" translatable="false">introduction</string>

    <!-- Sound preferences -->
    <bool name="sound_preference_default">true</bool>

    <string name="preference_key_sounds_all" translatable="false">sound_all</string>
    <string name="preference_key_sounds_cards" translatable="false">sound_cards</string>
    <string name="preference_key_sounds_reset" translatable="false">sound_reset</string>
    <string name="preference_key_sounds_weather" translatable="false">sound_weather</string>
    <string name="preference_key_sounds_horn" translatable="false">sound_horn</string>
    <string name="preference_key_sounds_burn" translatable="false">sound_burn</string>
    <string name="preference_key_sounds_coin" translatable="false">sound_coin</string>
</resources>
```

6.102 strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<resources>
    <string name="app_name">Gwint</string>
    <string name="app_description">Eine App, die es ermöglicht, das Kartenspiel Gwint aus The Witcher 3 mit analogen Karten zu spielen.</string>
    <string name="settings_title">Einstellungen</string>

    <string name="faction_nilfgaard">Kaiserreich Nilfgaard</string>
    <string name="faction_northern_kingdoms">Nördliche Königreiche</string>

    <string name="popUp_cards_add">Karte hinzufügen</string>
    <string name="popUp_cards_save">Speichern</string>
    <string name="popUp_add_card_label_dmg">Stärke</string>
    <string name="popUp_add_card_label_ability">Fähigkeit</string>
    <string name="popUp_add_card_label_binding">Gruppe</string>
    <string name="popUp_add_card_label_number">Anzahl</string>

    <string name="popUp_coin_normal_win">Du beginnst.</string>
    <string name="popUp_coin_normal_lose">Dein Gegner beginnt.</string>
    <string name="popUp_coin_terry_1">Terry Jeffords hat die Münze verbogen!</string>
    <string name="popUp_coin_stewie_1">Chris Griffin hat die Münze gegessen!</string>
    <string name="popUp_coin_vin">Vin hat Stahl verbrannt und die Münze weggedrückt!</string>

    <string name="unit_toString">%1$s%2$s (%3$d) - Fähigkeit: %4$s %5$s</string>
    <string name="unit_toString_multiplicity">%1$d× %2$s</string>
    <string name="unit_toString_melee">Nahkampf</string>
    <string name="unit_toString_range">Fernkampf</string>
    <string name="unit_toString_siege">Belagerungs</string>
    <string name="unit_toString_epic">held</string>
    <string name="unit_toString_unit">einheit</string>
    <string name="unit_toString_ability_none">Keine</string>
    <string name="unit_toString_squad">(%1$d)</string>
    <string name="unit_collection_toString_accumulation_word">%1$s und %2$s</string>

    <string name="alertDialog_cancel">Abbrechen</string>

    <string name="alertDialog_burn_title">Verbrennen</string>
    <string name="alertDialog_burn_positive">Verbrennen</string>
    <string name="alertDialog_burn_msg">Möchtest du wirklich %1$s verbrennen?</string>
    <string name="alertDialog_burn_negative">Nicht verbrennen</string>

    <string name="alertDialog_reset_title">Spielfeld zurücksetzen</string>
    <string name="alertDialog_reset_msg_default">Willst du wirklich das gesamte Spielfeld zurücksetzen?</string>
    <string name="alertDialog_reset_msg_faction_switch">Du hast \"Zurücksetzen bei Fraktionswechsel\" aktiviert. Soll das Spielfeld wirklich zurückgesetzt werden?</string>
    <string name="alertDialog_reset_checkbox">Die Fähigkeit der Monsterfraktion auslösen (du behältst eine zufällige Einheit)</string>
    <string name="alertDialog_factionreset_monster_toast_keep">Du behältst: %1$s</string>
    <string name="alertDialog_reset_positive">Zurücksetzen</string>
    <string name="alertDialog_reset_negative">Nicht zurücksetzen</string>

    <string name="alertDialog_revenge_title">Rächerfähigkeit</string>
    <string name="alertDialog_revenge_msg">Du hast eine Einheit vom Spielfeld entfernt, die einen Rächer herbeirufen kann. Möchtest du diese Fähigkeit auslösen?</string>
    <string name="alertDialog_revenge_positive">Rächer rufen</string>

    <string name="preference_design">Hintergrundbild</string>
    <string name="preference_warnings_title">Warnungen</string>
    <string name="preference_warnings_desc">Legt fest, ob beim Zurücksetzen eine Warnung erscheinen soll.</string>
    <string name="preference_faction_reset_title">Zurücksetzen bei Fraktionswechsel</string>
    <string name="preference_faction_reset_desc">Legt fest, ob das Spielfeld beim Fraktionswechsel automatisch zurückgesetzt werden soll.</string>

    <string name="preference_sounds_header">Tonausgabe</string>
    <string name="preference_sounds_category">Ereignisse</string>
    <string name="preference_sounds_cards_new">Neue Einheit</string>
    <string name="preference_sounds_cards_delete">Einheit entfernt</string>
    <string name="preference_sounds_weather">Wetteränderung</string>
    <string name="preference_sounds_horn">Horn des Kommandanten</string>
    <string name="preference_sounds_burn">Verbrennen</string>
    <string name="preference_sounds_coin">Münzwurf</string>

    <string name="preference_info_introduction">Einführung</string>

    <string name="preference_rules_header">Spielregeln</string>
    <string name="preference_rules_general_title">Allgemein</string>
    <string name="preference_rules_course_title">Spielablauf</string>
    <string name="preference_rules_factions_title">Fraktionen</string>
    <string name="preference_rules_commander_title">Anführer</string>
    <string name="preference_rules_cards_title">Kartensatz und Spielfeld</string>
    <string name="preference_rules_card_abilities_title">Kartenfähigkeiten</string>
    <string name="preference_rules_special_cards_title">Spezialkarten</string>

    <string name="introduction_button_skip">Überspringen</string>
    <string name="introduction_button_finish">Fertig</string>

    <string name="content_description_activity_main_background">Hintergrundbild</string>
    <string name="content_description_activity_main_point_ball">Gesamtpunktzahl</string>
    <string name="content_description_activity_main_faction_button">Fraktionslogo</string>
    <string name="content_description_activity_main_weather_button">Gutes-Wetter-Icon</string>
    <string name="content_description_activity_main_burn_button">Verbrennen-Icon</string>
    <string name="content_description_activity_main_coin_button">Münzen-Icon</string>
    <string name="content_description_activity_main_settings_button">Einstellungs-Icon</string>
    <string name="content_description_card_ability">Fähigkeits-Icon</string>
    <string name="content_description_card_delete">Löschen-Icon</string>
    <string name="content_description_card_copy">Kopieren-Icon</string>
    <string name="content_description_coinpopup_coin">Münze</string>
    <string name="content_description_coinpopup_stewie_1">Chris Griffin, der betroffen die Arme hebt</string>
    <string name="content_description_coinpopup_stewie_2">Stewie Griffin, der etwas erzählt</string>
    <string name="content_description_coinpopup_stewie_3">Stewie Griffin, der etwas schreit</string>
    <string name="content_description_coinpopup_terry_1">Eine Münze, die verbogen auf einem Tisch liegt</string>
    <string name="content_description_coinpopup_terry_2">Jake Peralta (Andy Samberg), der erstaunt etwas sagt</string>
    <string name="content_description_coinpopup_terry_3">Terry Jeffords (Terry Crews), der wütend, mit erhobenem Finger etwas sagt und dabei von Charles Boyle (Joe Lo Truglio) erschrocken angesehen wird</string>
    <string name="content_description_coinpopup_vin">Ein Bild von Vin aus der Mistborn-Reihe von Brandon Sanderson</string>
    <string name="content_description_factionpopup_monster">Karte mit Symbol der Monsterfraktion</string>
    <string name="content_description_factionpopup_nilfgaard">Karte mit Symbol der Kaiserreich-Nilfgaard-Fraktion</string>
    <string name="content_description_factionpopup_northern_kingdoms">Karte mit Symbol der Nördliche-Königreiche-Fraktion</string>
    <string name="content_description_factionpopup_scoiatael">Karte mit Symbol der Scoia\’taelfraktion</string>
    <string name="content_description_row_pointball">Punktanzeige der Angriffsreihe</string>
    <string name="content_description_row_weather">Wetteranzeige der Angriffsreihe</string>
    <string name="content_description_row_horn">Hornanzeige der Angriffsreihe</string>
    <string name="content_description_row_card">Kartenanzeige der Angriffsreihe</string>
    <string name="content_description_introduction_image">Screenshots der App</string>
    <string name="content_description_introduction_indicator">Punkt, der den Fortschritt der Einführung zeigt</string>
    <string name="content_description_introduction_next">Pfeil, der zum nächsten Schritt führt</string>

    <string name="rules_general_text"><![CDATA[<p>Gwint ist ein altes Kartenspiel für zwei Spieler, das vor allem bei Zwergen beliebt ist. Es stellt die Konfrontation zweier Armeen in einer Schlacht nach. Die Spieler sind die Generäle und die Karten ihre Streitkräfte. Jeder Gwintspieler verwendet seinen eigenen, individuell zusammengestellten Kartensatz. Für jede der vier Fraktionen lässt sich ein eigenes Deck erstellen: Monster, Kaiserreich Nilfgaard, Nördliche Königreiche und Scoia\’tael. Jede Fraktion enthält einige einzigartige Karten, die eine unterschiedliche Spielweise ermöglichen. Des Weiteren gibt es eine Anzahl neutraler Karten, die in allen vier Fraktionen eingesetzt werden können. Ziel des Spiels ist es, den Gegner zu besiegen, indem man in zwei von drei Runden eine höhere Punktzahl als der Gegner erreicht.</p>]]></string>
    <string name="rules_course_text"><![CDATA[<p>Zu Beginn der ersten Runde ziehen beide Spieler jeweils 10 Karten aus ihrem Deck. Danach dürfen zwei Karten abgelegt und neu gezogen werden. Dieser Schritt ist optional, die abgelegten Karten werden wieder in den Nachziehstapel eingemischt. Der Spieler, der den ersten Zug macht, wird per Münzwurf ermittelt.<br/>
    Die Spieler legen abwechselnd Karten. Es besteht Legezwang! Wenn ein Spieler keine Karte ausspielen möchte, muss er passen und darf den Rest der Runde keinen Spielzug mehr durchführen. Daraufhin kann der Gegner noch weitere Karten ausspielen, bis er selbst passt. Sollte ein Spieler keine Handkarten mehr haben, muss er ebenfalls passen.<br />
    Wenn beide Spieler gepasst haben, ist die Runde beendet und der Gewinner wird anhand der Gesamtstärke der Einheiten ermittelt. Sollte die Gesamtstärke gleich sein, zählt die Runde als Sieg für beide Spieler. Nun werden alle Karten vom Spielfeld auf den Ablagestapel gelegt. Die Handkarten werden in der nächsten Runde weiter verwendet, es werden keine neuen Karten gezogen. Der Gewinner hat in der nächsten Runde den ersten Zug. Bei Unentschieden beginnt derjenige, der in der vergangenen Runde den zweiten Zug hatte. Ziel des Spiels ist es, zwei Runden zu gewinnen.</p>
    ]]></string>
    <string name="rules_factions_text"><![CDATA[<p>
    Bei Gwint unterscheidet man vier Fraktionen: Monster, Kaiserreich Nilfgaard, Nördliche Königreiche und die Scoia\’tael. Jede dieser Fraktionen besitzt einzigartige Karten, Anführer und jeweils einen bestimmten Fraktionsvorteil. Zusätzlich gibt es noch neutrale Einheiten, welche keiner Fraktion zugehörig sind und somit in jedem Deck auftauchen können.</p>
    Die Monster werden vom König der Wilden Jagd Eredin Bréacc Glas, auch Sperber genannt, angeführt. Ihr Vorteil ist die Eisenhaut, welche es ermöglicht, dass nach jeder Runde eine zufällige Einheit auf dem Feld verbleibt.<br />
    Emhyr var Emreis, der sich auch „Die Weiße Flamme, Die Auf den Grabhügeln Der Feinde tanzt“ nennt, ist Kaiser von Nilfgaard, dessen Stärke der Verrat ist. Dieser ermöglicht es, dass Spieler mit einem Nilfgaard-Deck jede Runde gewinnen, die sonst als Unentschieden enden würde.<br />
    Den expandierenden Nilfgaardern stellen sich die Nördlichen Königreiche unter Foltest, König Temeriens, entgegen. Diese erhalten nach einer gewonnenen Runde eine Verstärkung, dürfen also eine Karte vom Nachziehstapel ziehen.<br />
    In ihrem Kampf versuchen die Königreiche, Anderlinge zu dezimieren, darunter die Scoia\’tael, welche als nicht-menschliche Rebellen gegen die Repressionen aufbegehren und sich dafür den Hinterhalt zunutze machen: Die Scoia\’tael können in der ersten Runde entscheiden, wer den ersten Zug macht. Sie werden von der Elfin Francesca Findabair angeführt, die auch als Enid an Gleanna (Gänseblümchen des Tals) bekannt ist.
    ]]></string>
    <string name="rules_commander_text"><![CDATA[<p>
    <p>Jede Fraktion besitzt bestimmte Anführer mit unterschiedlichen Fähigkeiten. Diese werden hier aufgezählt.</p>
    <h1>Monster: Eredin Bréacc Glas</h1>
    <p>Der Trügerische: <br />Passiv - Verdoppelt die Stärke aller Karten mit der Fähigkeit "Spion" (wirkt auf beide Spieler).</p>
    <p>Bringer des Todes: <br />Wirf 2 Karten ab und ziehe 1 Karte deiner Wahl aus deinem Kartensatz.</p>
    <p>Führer der Roten Reiter: <br />Durchsuche deinen Nachziehstapel nach einer Wetterkarte und spiele sie sofort aus.</p>
    <p>König der Wilden Jagd: <br />Verdopple die Stärke deiner Nahkampfeinheiten, sofern kein "Horn des Kommandanten" in der Reihe liegt.</p>
    <p>Zerstörer der Welten: <br />Nimm eine Karte von deinem Ablagestapel zurück auf die Hand.</p>
    <h1>Kaiserreich Nilfgaard: Emhyr var Emreis</h1>
    <p>Der Unbarmherzige: <br />Zieh eine Karte vom Ablagestapel deines Gegners.</p>
    <p>Die Weiße Flamme: <br />Lass die Anführer-Fähigkeit deines Gegners unwirksam werden.</p>
    <p>Invasor des Nordens: <br />Fähigkeiten, die Einheiten auf dem Schlachtfeld wiederherstellen, stellen zufällig ausgewählte Einheiten wieder her. Wirkt auf beide Spieler.</p>
    <p>Kaiser von Nilfgaard: <br />Schau dir drei zufällig gewählte Karten aus der Hand deines Gegners an.</p>
    <p>Kaiserliche Majestät: <br />Durchsuche deinen Nachziehstapel nach "Strömender Regen" und spiele die Karte sofort aus.</p>
    <h1>Nördliche Königreiche: Foltest</h1>
    <p>Der Stählerne: <br />Zerstöre die stärkste(n) Belagerungseinheit(en) des Gegners, sofern deren Gesamtstärke mindestens 10 beträgt.</p>
    <p>Der Anführer des Nordens: <br />Entferne durch "Klirrende Kälte", "Strömender Regen" und "Extrem dichter Nebel" hervorgerufene Wettereffekte.</p>
    <p>Der Belagerer: <br />Verdopple die Stärke deiner Belagerungseinheiten, sofern kein "Horn des Kommandanten" in der Reihe liegt.</p>
    <p>König Temeriens: <br />Durchsuche deinen Nachziehstapel nach "Extrem dichter Nebel" und spiele die Karte sofort aus.</p>
    <p>Sohn des Medell: <br />Zerstöre die stärkste(n) Fernkampfeinheit(en) des Gegners, wenn die Gesamtstärke seiner Fernkampfeinheiten mindestens 10 beträgt.</p>
    <h1>Scoia\’tael: Francesca Findabair</h1>
    <p>Gänseblümchen des Tals: <br />Zieh zu Beginn der Schlacht eine Extrakarte.</p>
    <p>Hoffnung der Aen Seidhe: <br />Verschiebt Einheiten mit der Fähigkeit „Gewandtheit‟ auf die für sie optimale Reihe, sofern sie nicht bereits dort befinden.</p>
    <p>Königin Dol Blathannas: <br />Zerstöre die stärkste(n) Nahkampfeinheit(en) des Gegners, wenn die Gesamtstärke seiner Nahkampfeinheiten mindestens 10 beträgt.</p>
    <p>Die Schöne: <br />Verdopple die Stärke deiner Fernkampfeinheiten, sofern kein „Horn des Kommandanten‟ in der Reihe liegt.</p>
    <p>Reinblütige Elfin: <br />Durchsuche deinen Nachziehstapel nach „Klirrende Kälte‟ und spiele die Karte sofort aus.</p>
    ]]></string>
    <string name="rules_cards_text"><![CDATA[<p>
    Man unterscheidet bei den Kartentypen in drei Hauptkategorien: Anführer-, Einheiten- und Sonderkarten. In jedem Kartensatz gibt es genau eine Anführerkarte (je Fraktion kann der Spieler zwischen vier unterschiedlichen wählen). Diese hat eine bestimmte Fähigkeit, die nur einmal im ganzen Spiel während eines Zuges aktiviert werden kann, dafür aber unabhängig von den Handkarten immer verfügbar ist.<br />
    Dazu kommen mindestens 22 Einheitenkarten, zu denen auch die Heldenkarten zählen. Letztere sind gegen Karteneffekte jeglicher Art (positiv wie negativ) immun. Die Einheitenkarten werden während eines Spielzuges in Angriffsreihen gelegt. Jeder Spieler besitzt davon drei, jeweils eine für Nahkämpfer, Fernkämpfer und Belagerungsgeräte. In welche dieser Reihen die Einheit gelegt werden kann, erkennt man am orange hinterlegten Symbol.<br />
    Schließlich gibt es noch maximal zehn Sonderkarten, zu denen „Finte‟, „Verbrennen‟, „Horn des Kommandanten‟ und die Wetterkarten gehören (Effektbeschreibungen sind unter „Spezialkarten‟ nachzulesen). Wetterkarten werden in einen abgesonderten Bereich des Spielfelds platziert, da sie die Reihen beider Spieler betreffen. Das Horn des Kommandanten wird an die Reihe angelegt, für die es ausgespielt wird.
    ]]></string>
    <string name="rules_card_abilities_text"><![CDATA[
    <p>Einige Einheitenkarten besitzen besondere Fähigkeiten, welche hier genauer erklärt werden.</p>
    <p>Spion: <br />Lege diese Karte auf das Schlachtfeld des Gegners - sie wird auf die Gesamtstärke des Gegners angerechnet - und ziehe zwei Karten.</p>
    <p>Enge Bindung: <br />Liegt diese Karte neben einer mit demselben Namen, verdopple die Stärke beider Karten.</p>
    <p>Musterung: <br />Wird diese Karte gespielt, durchsuche deinen Nachziehstapel nach Einheiten mit demselben Namen und spiele diese sofort aus.</p>
    <p>Moralschub: <br />Verstärke alle anderen Einheiten der Reihe um einen Stärkepunkt.</p>
    <p>Heilung: <br />Wird diese Karte gespielt, wähle eine Einheitenkarte aus deinem Ablagestapel (Heldenkarten ausgenommen) und spiele diese aus. Karten mit der Fähigkeit Gewandtheit müssen wieder in ihre ursprüngliche Reihe zurückgelegt werden.</p>
    <p>Horn des Kommandanten: <br />Verdoppele die Stärke aller anderen Einheitenkarten dieser Reihe. Dieser Effekt kumuliert nicht mit dem Effekt der Spezialkarte Horn des Kommandanten! Liegen also beide Karten in der Angriffsreihe, wird der Schaden aller Einheiten (einschließlich dieser) nur einmal verdoppelt.</p>
    <p>Gewandtheit: <br />Platziere diese Karte in der Nah- oder Fernkampfreihe. Nach dem Ausspielen kannst du sie nicht mehr verschieben.</p>
    <p>Verbrennen: <br />Zerstöre die stärkste(n) Nahkampfeinheit(en) des Gegners, wenn die Gesamtstärke seiner Nahkampfeinheiten mindestens 10 beträgt.</p>
    <p>Rächer herbeirufen: <br />Wird diese Karte vom Spielfeld entfernt (egal auf welche Art), wird ein Rächer herbeigerufen.</p>
    ]]></string>
    <string name="rules_special_cards_text"><![CDATA[
    <p>Spezialkarten können alternativ zu Einheitenkarten gespielt werden, wobei bestimmte Effekte ausgelöst werden, die hier erklärt werden.</p>
    <p>Klirrende Kälte: <br />Setze die Stärke aller Nahkampfeinheiten beider Spieler auf 1. Wenn
```

```
diese Karte bereits ausgespielt wurde, lege sie sofort auf den Ablagestapel.</p>
    <p>Extrem dichter Nebel: <br />Setze die Stärke aller Fernkampfeinheiten beider Spieler auf 1. Wenn diese Karte bereits ausgespielt wurde, lege sie sofort auf den Ablagestapel.</p>
    <p>Strömender Regen: <br />Setze die Stärke aller Belagerungseinheiten beider Spieler auf 1. Wenn diese Karte bereits ausgespielt wurde, lege sie sofort auf den Ablagestapel.</p>
    <p>Gutes Wetter: <br />Entferne alle Effekte von Wetterkarten und lege sie mit dieser Karte auf den Ablagestapel.</p>
    <p>Finte: <br />Tausche diese Karte gegen eine Einheitenkarte auf dem Schlachtfeld aus, um sie wieder deiner Hand hinzuzufügen. </p>
    <p>Horn des Kommandanten: <br />Verdoppele die Stärke aller Einheitenkarte dieser Reihe (maximal einmal pro Reihe). Dieser Effekt kumuliert nicht mit dem Karteneffekt Rittersporns! Liegen also beide Karten in der Angriffsreihe, wird der Schaden aller Einheiten (einschließlich Rittersporn) nur einmal verdoppelt.</p>
    <p>Verbrennen: <br />Entferne die stärkste(n) Karte(n) beider Spieler auf dem Schlachtfeld und lege sie mit dieser Karte auf den Ablagestapel.</p>
    ]]></string>
    <string name="introduction_page1"><![CDATA[
    <p>Danke, dass du „Gwint - Eine Rechenhilfe“ installiert hast! Bitte beachte, dass diese App nur dazu da ist, dich bei einem Spiel mit analogen Gwintkarten zu unterstützen, indem sie deinen Punktestand errechnet. Du kannst in dieser App nicht gegen KI-Gegner oder ohne analoge Gwintkarten spielen!</p>
    <p>Also falls du es noch nicht getan hast: Drucke dir einen Satz Gwintkarten aus (<a href="https://github.com/PeterNaggschga/Gwent#overview">Anleitung hier</a>) und suche dir einen Gegner, der dasselbe tut (ihr braucht beide diese App)!</p>
    ]]></string>
    <string name="introduction_page2"><![CDATA[
    <p>Dein Spielfeld besteht aus drei Angriffsreihen: Nahkampfreihe, Fernkampfreihe und Belagerungsreihe (von links nach rechts). In diese Reihen werden die Einheitenkarten beim Ausspielen gelegt. Jede Angriffsreihe kann außerdem durch eine bestimmte Wetterlage geschwächt oder durch ein Horn des Kommandanten verstärkt werden.</p>
    ]]></string>
    <string name="introduction_page3"><![CDATA[
    <p>Die Angriffsreihen sind wie folgt aufgebaut:</p>
    <p>1: Die Gesamtstärke aller Einheiten in der Reihe.</p>
    <p>2: Das aktuelle Wetter (kann durch antippen geändert werden).</p>
    <p>3: Die Verstärkung durch ein Horn des Kommandanten (kann durch antippen geändert werden).</p>
    <p>4: Die Anzahl der Einheiten in dieser Reihe (durch Tippen können Einheiten hinzugefügt oder entfernt werden).</p>
    ]]></string>
    <string name="introduction_page4"><![CDATA[
    <p>Neben deinem eigentlichen Spielfeld gibt es an der Seite ein Menü mit weiteren nützlichen Funktionen:</p>
    <p>1: Gesamtstärke aller Einheiten auf dem Spielfeld</p>
    <p>2: Fraktion ändern</p>
    <p>3: Spielfeld zurücksetzen</p>
    <p>4: Wettereffekte entfernen</p>
    <p>5: Stärkste Einheit(en) verbrennen</p>
    <p>6: Münzwurf</p>
    <p>7: Einstellungen</p>
    ]]></string>
    <string name="introduction_page5"><![CDATA[
    <p>Weitere Informationen zum Spiel und seinen Regeln findest du in den Einstellungen. Dort kannst du diese Tour auch wiederholen!</p>
    <p>Doch jetzt erstmal: viel Spaß mit analogem Gwint! Über Verbesserungsvorschläge, Fehlermeldungen und sonstiges Feedback freue ich mich sehr, nutze dazu gerne den <a href="https://play.google.com/store/apps/details?id=com.peternaggschga.gwint">Google Play Store</a> oder <a href="https://github.com/PeterNaggschga/Gwent/issues/new/choose">GitHub</a>!</p>
    ]]></string>
    <string name="faction_monster">Monster</string>
    <string name="array_design_jaskier">Rittersporn</string>
    <string name="array_design_default">Standard</string>
    <string name="array_design_geralt">Geralt von Riva</string>
    <string name="array_design_yennefer">Yennefer von Vengerberg</string>
    <string name="add_picker_epic_epic">Held</string>
    <string name="add_picker_ability_default">Keine</string>
    <string name="add_picker_ability_moralBoost">Moralschub</string>
    <string name="add_picker_ability_revenge">Rächer</string>
    <string name="add_picker_ability_binding">Enge Bindung</string>
    <string name="popUp_add_card_binding_count">Anzahl Einheiten in %1$d. Gruppe: %2$d</string>
    <string name="alertDialog_revenge_negative">Nicht auslösen</string>
</resources>
```

6.103 strings.xml

```
<resources>
    <string name="app_name">Gwent</string>
    <string name="app_description">An app enabling you to play the Gwent card game from The Witcher 3 with analog cards.</string>

    <string name="faction_monster">Monsters</string>
    <string name="faction_nilfgaard">Nilfgaardian Empire</string>
    <string name="faction_northern_kingdoms">Northern Kingdoms</string>
    <string name="faction_scoiatael" translatable="false">Scoia\'tael</string>

    <string name="popUp_cards_add">Add card</string>
    <string name="popUp_cards_save">Save</string>
    <string name="popUp_add_card_label_epic" translatable="false">Status</string>
    <string name="add_picker_epic_normal" translatable="false">Normal</string>
    <string name="add_picker_epic_epic">Hero</string>
    <string name="popUp_add_card_label_dmg">Strength</string>
    <string name="popUp_add_card_label_ability">Ability</string>
    <string name="add_picker_ability_default">None</string>
    <string name="add_picker_ability_moralBoost">Morale boost</string>
    <string name="add_picker_ability_horn" translatable="false">Horn</string>
    <string name="add_picker_ability_revenge">Avenger</string>
    <string name="add_picker_ability_binding">Tight bond</string>
    <string name="popUp_add_card_label_binding">Squad</string>
    <string name="popUp_add_card_label_number">Number</string>
    <string name="popUp_add_card_binding_count">Number of units in squad %1$d: %2$d</string>

    <string name="popUp_coin_normal_win">You begin.</string>
    <string name="popUp_coin_normal_lose">Your opponent begins.</string>
    <string name="popUp_coin_terry_1">Terry Jeffords folded your coin!</string>
    <string name="popUp_coin_terry_2" translatable="false">Oh my god! Did you just fold my quarter in half?</string>
    <string name="popUp_coin_terry_3" translatable="false">Fate\'s a bitch--you make the choice!</string>
    <string name="popUp_coin_stewie_1">Chris Griffin ate your coin!</string>
    <string name="popUp_coin_stewie_2" translatable="false">But don\'t worry, it will come out of my body sooner or later!</string>
    <string name="popUp_coin_stewie_3" translatable="false">He\'s right. I ate a dime once.</string>
    <string name="popUp_coin_stewie_4" translatable="false">It became a manhole cover for like three days--but then pow!</string>
    <string name="popUp_coin_vin">Vin burned steel and pushed away your coin!</string>

    <string name="unit_toString">%1$s %2$s (%3$d) with %4$s ability %5$s</string>
    <string name="unit_toString_multiplicity">%1$d× %2$s</string>
    <string name="unit_toString_melee">Close combat</string>
    <string name="unit_toString_range">Ranged combat</string>
    <string name="unit_toString_siege">Siege combat</string>
    <string name="unit_toString_epic">hero</string>
    <string name="unit_toString_unit">unit</string>
    <string name="unit_toString_ability_none">no</string>
    <string name="unit_toString_squad">(%1$d)</string>
    <string name="unit_collection_toString_accumulation_symbol" translatable="false">%1$s, %2$s</string>
    <string name="unit_collection_toString_accumulation_word">%1$s, and %2$s</string>

    <string name="alertDialog_cancel">Cancel</string>

    <string name="alertDialog_burn_title">Scorch</string>
    <string name="alertDialog_burn_positive">Scorch</string>
    <string name="alertDialog_burn_negative">Don\'t scorch</string>
    <string name="alertDialog_burn_msg">Do you really want to scorch %1$s?</string>

    <string name="alertDialog_reset_title">Reset board</string>
    <string name="alertDialog_reset_msg_default">Do you really want to reset the board?</string>
    <string name="alertDialog_reset_msg_faction_switch">You activated \"Reset on change of faction\". Do you really want to reset the board?</string>
    <string name="alertDialog_reset_checkbox">Activate perk of monster faction (you keep one random unit)</string>
    <string name="alertDialog_factionreset_monster_toast_keep">You keep: %1$s</string>
    <string name="alertDialog_reset_positive">Reset</string>
    <string name="alertDialog_reset_negative">Don\'t reset</string>

    <string name="alertDialog_revenge_title">Avenger ability</string>
    <string name="alertDialog_revenge_msg">You removed units that can summon an avenger. Do you want to activate their ability?</string>
    <string name="alertDialog_revenge_positive">Summon avenger</string>
    <string name="alertDialog_revenge_negative">Don\'t activate</string>

    <string name="array_design_default">Default</string>
    <string name="array_design_geralt">Geralt of Rivia</string>
    <string name="array_design_ciri" translatable="false">Cirilla</string>
    <string name="array_design_jaskier">Dandelion</string>
    <string name="array_design_yennefer">Yennefer of Vengerberg</string>
    <string name="array_design_eredin" translatable="false">Eredin Bréacc Glas</string>
    <string name="preference_design">Background</string>
    <string name="preference_warnings_title">Warnings</string>
    <string name="preference_warnings_desc">Determines if warnings are shown before you can reset the board.</string>
    <string name="preference_faction_reset_title">"Reset on change of faction"</string>
    <string name="preference_faction_reset_desc">Determines if the board is reset on change of faction.</string>

    <string name="preference_sounds_header">Sounds</string>
    <string name="preference_sounds_category">Events</string>
    <string name="preference_sounds_cards_new">New unit</string>
    <string name="preference_sounds_cards_delete">Unit removed</string>
    <string name="preference_sounds_weather">Change of weather</string>
    <string name="preference_sounds_horn">Commander\'s horn</string>
    <string name="preference_sounds_burn">Scorch</string>
    <string name="preference_sounds_coin">Coin-flip</string>

    <string name="preference_info_category_header" translatable="false">Info</string>
    <string name="preference_info_introduction">Introduction</string>

    <string name="preference_rules_header">Rules</string>
    <string name="preference_rules_general_title">General</string>
    <string name="preference_rules_course_title">Gameplay</string>
    <string name="preference_rules_factions_title">Factions</string>
    <string name="preference_rules_commander_title">Leaders</string>
    <string name="preference_rules_cards_title">Deck and playboard</string>
    <string name="preference_rules_card_abilities_title">Card abilities</string>
    <string name="preference_rules_special_cards_title">Special cards</string>
    <string name="preference_rules_general_key" translatable="false">rules_general</string>
    <string name="preference_rules_course_key" translatable="false">rules_course</string>
    <string name="preference_rules_factions_key" translatable="false">rules_factions</string>
    <string name="preference_rules_commander_key" translatable="false">rules_commander</string>
    <string name="preference_rules_cards_key" translatable="false">rules_cards</string>
    <string name="preference_rules_card_abilities_key" translatable="false">rules_card_abilities</string>
    <string name="preference_rules_special_cards_key" translatable="false">rules_special_cards</string>
    <string name="preference_first_use_key" translatable="false">firstUse</string>

    <string name="introduction_button_skip">Skip</string>
    <string name="introduction_button_finish">Finish</string>

    <string name="content_description_activity_main_background">background</string>
    <string name="content_description_activity_main_point_ball">total score</string>
    <string name="content_description_activity_main_faction_button">faction logo</string>
    <string name="content_description_activity_main_reset_button" translatable="false">Reset-Icon</string>
    <string name="content_description_activity_main_weather_button">Clear-Weather-Icon</string>
    <string name="content_description_activity_main_burn_button">Scorch-Icon</string>
    <string name="content_description_activity_main_coin_button">Coin-Icon</string>
    <string name="content_description_activity_main_settings_button">Settings-Icon</string>
    <string name="content_description_card_ability">Ability-Icon</string>
    <string name="content_description_card_delete">Delete-Icon</string>
    <string name="content_description_card_copy">Copy-Icon</string>
    <string name="content_description_coinpopup_coin">coin</string>
    <string name="content_description_coinpopup_stewie_1">Chris Griffin who raises his arms in consternation</string>
    <string name="content_description_coinpopup_stewie_2">Stewie Griffin who tells a story</string>
    <string name="content_description_coinpopup_stewie_3">Stewie Griffin who yells</string>
    <string name="content_description_coinpopup_terry_1">A bent coin lying on a table</string>
    <string name="content_description_coinpopup_terry_2">Jake Peralta (Andy Samberg) who says something in astonishment</string>
    <string name="content_description_coinpopup_terry_3">Terry Jeffords (Terry Crews) who says something angrily and Charles Boyle (Joe Lo Truglio) looking at him in terror</string>
    <string name="content_description_coinpopup_vin">A fan art showing Vin from Brandon Sanderson\’s Mistborn-series.</string>
    <string name="content_description_factionpopup_monster">Card with symbol of monster faction</string>
    <string name="content_description_factionpopup_nilfgaard">Card with symbol of nilfgaardian empire faction</string>
    <string name="content_description_factionpopup_northern_kingdoms">Card with symbol of Northern Kingdoms faction</string>
    <string name="content_description_factionpopup_scoiatael">Card with symbol of Scoia\’tael faction</string>
    <string name="content_description_row_pointball">point display of combat row</string>
    <string name="content_description_row_weather">weather display of combat row</string>
    <string name="content_description_row_horn">horn display of combat row</string>
    <string name="content_description_row_card">card display of combat row</string>
    <string name="content_description_introduction_image">screenshots of this app</string>
    <string name="content_description_introduction_indicator">dot that indicates progress</string>
    <string name="content_description_introduction_next">forward arrow</string>

    <string name="rules_general_text"><![CDATA[<p>Gwent is an old card game especially fancied by dwarfs. It represents the confrontation of two opposing armies. The players are generals and cards represent their units. Every player uses his own unique deck of cards. Decks can be created for each of the four factions: Monsters, Nilfgaardian Empire, Northern Kingdoms and Scoia\’tael. Each faction has unique cards which lead to different play styles. Moreover, there are cards that can be used for every faction. Your goal is to win two rounds out of three by having more strength than your opponent.</p>]]></string>
    <string name="rules_course_text"><![CDATA[<p>At the beginning of the first round each player draws ten cards from their deck. After drawing, you can discard two cards and draw a new card for each. This step is optional, discarded cards are put back into the deck. The beginning player is determined by flipping a coin.<br/>The players place cards alternately. You have to play one card every round! If a player doesn\’t want to play a card, they have to pass and aren\’t allowed to play another card until the round has ended. When one player passes, their opponent can still play cards until they pass too. If one of the opponents uses up all of his cards, he has to pass.<br />A round ends when both players pass. The player with the highest total score wins the round. When both opponents have the same number of points, the round is seen as a win for both sides. All cards are moved to the discard pile at the end of each round. Your hand is reused in the next round--you don\’t draw new cards. The winner of the last round gets the first turn. Goal of the game is to win two rounds.</p>]]></string>
    <string name="rules_factions_text"><![CDATA[<p>There are four factions: Monsters, Nilfgaardian Empire, Northern Kingdoms and Scoia\’tael. Every faction has got unique cards, leaders and perks.
```

```
Moreover, there are some additional cards that are neutral and can be used in every faction.</p>Leader of monster faction is Eredin Bréacc Glas, also known as Sparrowhawk who is the King of the Wild Hunt. Their perk is iron skin which makes it possible to keep one unit at the end of each round.<br />Emhyr var Emreis, also known as \"The White Flame Dancing on the Barrows of his Enemies\", is emperor of Nilfgaard. Treachery causes any player with a nilfgaardian deck to win if points draw level.<br />Archenemy of Nilfgaard is Foltest, King of Temeria and leader of the Northern Kingdoms faction. They are able to draw one card after a won round.<br />In their fight, the Northern Kingdoms try to decimate Nonhumans which are represented by the Scoia\'tael faction. Scoia\'tael players may decide who gets the first turn in the first round. They are led by Francesca Findabair, also known as Enid an Gleanna (Daisy of the Valleys).]]></string>
    <string name="rules_commander_text"><![CDATA[<p>
    <p>Each faction has certain leaders with different abilities. They are listed here.</p>
    <h5>Monsters: Eredin Bréacc Glas</h5>
    <p>The Treacherous: <br />Passive - Doubles the strength of all spy cards (affects both players).</p>
    <p>Bringer of Death: <br />Restore a card from your discard pile to your hand.</p>
    <p>Commander of the Red Riders: <br />Double the strength of all your Close combat units (unless a Commander\'s horn is also present in that row).</p>
    <p>King of the Wild Hunt: <br />Pick any weather card from your deck and play it instantly.</p>
    <p>Destroyer of Worlds: <br />Discard two cards and draw one card of your choice from your deck.</p>
    <h5>Nilfgaardian Empire: Emhyr var Emreis</h5>
    <p>The Relentless: <br />Draw a card from your opponent\'s discard pile.</p>
    <p>The White Flame: <br />Cancel your opponent\'s Leader Ability.</p>
    <p>Invader of the North: <br />Abilities that restore a unit to the battlefield restore a randomly-chosen unit. Affects both players.</p>
    <p>Emperor of Nilfgaard: <br />Look at three random cards from your opponent\'s hand.</p>
    <p>His Imperial Majesty: <br />Pick a Torrential Rain card from your deck and play it instantly.</p>
    <h5>Northern Kingdoms: Foltest</h5>
    <p>The Steel-Forged: <br />Destroy your enemy\'s strongest Siege unit(s) if the combined strength of all their Siege units is 10 or more.</p>
    <p>Lord Commander of the North: <br />Clear any weather effects (resulting from Biting Frost, Torrential Rain or Impenetrable Fog cards) in play.</p>
    <p>The Siegesmaster: <br />Doubles the strength of all your Siege units (unless a Commander\'s Horn is also present on that row).</p>
    <p>King of Temeria: <br />Pick an Impenetrable Fog card from your deck and play it instantly.</p>
    <p>Son of Medell: <br />Destroy your enemy\'s strongest Ranged combat unit(s) if the combined strength of all their Ranged combat units is 10 or more.</p>
    <h5>Scoia\'tael: Francesca Findabair</h5>
    <p>Daisy of the Valley: <br />Draw an extra card at the beginning of the battle.</p>
    <p>Hope of the Aen Seidhe: <br />Move agile units to whichever valid row maximizes their strength (don\'t move units already in optimal row).</p>
    <p>Queen of Dol Blathanna: <br />Destroy your enemy\'s strongest Close combat unit(s) if the combined strength of all their Close combat units is 10 or more.</p>
    <p>The Beautiful: <br />Doubles the strength of all your Ranged combat units (unless a Commander\'s Horn is also present on that row).</p>
    <p>Pureblood Elf: <br />Pick a Biting Frost card from your deck and play it instantly.</p>
    ]]></string>
    <string name="rules_cards_text"><![CDATA[<p>
    There are three major types of cards: Leaders, units and Special Cards.
    Each deck has one leader card.
    It has a certain ability which can be activated once per game.<br />
    Moreover, there are at least 22 Unit Cards including Heroes.
    Heroes are immune to any boost or weakening.
    Units are played in turns.
    Each player has got three combat rows, one for Close combat, Ranged combat and Siege units each.
    It can be seen on the cards which row must be used.<br />
    Also there are Special Cards (maximum 10), \"Decoy\",
    \"Scorch\", \"Commander\'s Horn\" and weather cards.
    Weather cards are placed in a separate section of the board
    because they influence both players equally.
    Commander\'s Horn is placed at the designated combat row.</p>
    ]]></string>
    <string name="rules_card_abilities_text"><![CDATA[
    <p>Some unit cards have got special abilities which are explained in the following.</p>
    <p>Spy: <br />Must be placed on your opponent\'s battlefield (and counts toward your opponent\'s total score), and allows you to draw 2 cards extra from your deck.</p>
    <p>Tight Bond: <br />When placed next to a card with the same name, double the strength of both cards.</p>
    <p>Muster: <br />When played, automatically find any cards with the same name in your deck and play them instantly.</p>
    <p>Morale Boost: <br />When played, add +1 strength to all other units in the row it is played in.</p>
    <p>Medic: <br />Allow you to choose one card from your discard pile and play it instantly (no Heroes or Special Cards). Agile units have to be placed in the combat row they have been in before.</p>
    <p>Commander\'s Horn: <br />Double the strength of all other units in this combat row.</p>
    <p>Agile: <br />Can be placed in either the Close combat or the Ranged combat row. They cannot be moved once placed.</p>
    <p>Scorch: <br />Kill the strongest card(s) in the specified row of the enemy.</p>
    <p>Summon Avenger: <br />Summon a powerful new unit card to take their place when they are removed from the battlefield.</p>
    ]]></string>
    <string name="rules_special_cards_text"><![CDATA[
    <p>Special Cards can be played instead of Unit Cards and trigger certain effects which are listed below.</p>
    <p>Biting Frost: <br />Sets all Close combat units to strength 1 for both players.</p>
    <p>Impenetrable Fog: <br />Sets all Ranged combat units to strength 1 for both players.</p>
    <p>Torrential Rain: <br />Sets all Siege units to strength 1 for both players.</p>
    <p>Clear Weather: <br />Removes all weather effects.</p>
    <p>Decoy: <br />Swap with a card on the battlefield.</p>
    <p>Commander\'s Horn: <br />Doubles the strength of the selected row.</p>
    <p>Scorch: <br />Discard after playing. Kills the strongest card(s) on the battlefield.</p>
    ]]></string>
    <string name="introduction_page1"><![CDATA[
    <p>Thanks for installing \"Gwent - A Calculator\"! Please note that this app is made to support you in playing Gwent using analog cards by calculating your points. This app does not allow you to play Gwent against AI or without analog cards!</p>
    <p>So if you didn\'t do so yet: Print out a deck of Gwent cards (<a href="https://github.com/PeterNaggschga/Gwent#overview">guidance here</a>) and find an opponent who does the same (both of you need to install this app).</p>
    ]]></string>
    <string name="introduction_page2"><![CDATA[
    <p>Your game board consists of three combat rows: Close combat, Ranged combat and Siege row
    (left to right).
    These rows contain unit cards.
    Each of them can be weakened by certain weather cards or boosted by a Commander\'s Horn.</p>
    ]]></string>
    <string name="introduction_page3"><![CDATA[
    <p>Combat rows consist of the following parts:</p>
    <p>1: Overall strength of all units placed in this row.</p>
    <p>2: The current weather (can be changed by tapping).</p>
    <p>3: The boost of a Commander\'s Horn (can be changed by tapping).</p>
    <p>4: Number of units in this combat row (you can add or remove units by tapping).</p>
    ]]></string>
    <string name="introduction_page4"><![CDATA[
    <p>Next to the game board there is a menu providing useful functions:</p>
    <p>1: Overall strength of all units</p>
    <p>2: Change faction</p>
    <p>3: Reset board</p>
    <p>4: Clear weather</p>
    <p>5: Scorch strongest unit(s)</p>
    <p>6: Coin-flip</p>
    <p>7: Settings</p>
    ]]></string>
    <string name="introduction_page5"><![CDATA[
    <p>More information on Gwent and its rules can be found in the settings. There you can repeat this introduction too!</p>
    <p>But first: Have fun with analog Gwent! Please use the <a href="https://play.google.com/store/apps/details?id=com.peternaggschga.gwint">Google Play Store</a> or <a href="https://github.com/PeterNaggschga/Gwent/issues/new/choose">GitHub</a> to share improvement suggestions, bugs and other feedback with me!</p>
    ]]></string>
    <string name="settings_title">Settings</string>
</resources>
```

6.104 styles.xml

```
<resources>

    <style name="BaseTheme" parent="Theme.AppCompat.NoActionBar">
        <item name="windowNoTitle">true</item>
        <item name="windowActionBar">false</item>
        <item name="android:windowFullscreen">true</item>
        <item name="android:windowContentOverlay">@null</item>
        <item name="android:windowTranslucentNavigation">true</item>
    </style>

    <style name="MonsterTheme" parent="BaseTheme">
        <item name="colorPrimary">@color/color_text_monster</item>
        <item name="colorAccent">@color/color_accent_monster</item>
        <item name="card_count_text_color">@color/color_text_monster</item>
        <item name="android:alertDialogIcon">@drawable/icon_round_monster</item>
        <item name="point_ball_mipmap">@drawable/ball_red</item>
        <item name="card_view_mipmap">@drawable/card_monster_landscape_free</item>
        <item name="card_background_color">@color/color_background_monster</item>
        <item name="card_background_gradient">@color/color_gradient_monster</item>
        <item name="card_view_button_background">@color/color_card_view_button_background_monster
        </item>
    </style>

    <style name="NilfgaardTheme" parent="BaseTheme">
        <item name="colorPrimary">@color/color_text_nilfgaard</item>
        <item name="colorAccent">@color/color_accent_nilfgaard</item>
        <item name="card_count_text_color">@color/color_text_nilfgaard</item>
        <item name="android:alertDialogIcon">@drawable/icon_round_nilfgaard</item>
        <item name="point_ball_mipmap">@drawable/ball_grey</item>
        <item name="card_view_mipmap">@drawable/card_nilfgaard_landscape_free</item>
        <item name="card_background_color">@color/color_background_nilfgaard</item>
        <item name="card_background_gradient">@color/color_gradient_nilfgaard</item>
        <item name="card_view_button_background">
```

```
            @color/color_card_view_button_background_nilfgaard
        </item>
    </style>

    <style name="NorthernKingdomsTheme" parent="BaseTheme">
        <item name="colorPrimary">@color/color_text_northern_kingdoms</item>
        <item name="colorAccent">@color/color_accent_northern_kingdoms</item>
        <item name="card_count_text_color">@color/color_text_northern_kingdoms</item>
        <item name="android:alertDialogIcon">@drawable/icon_round_northern_kingdoms</item>
        <item name="point_ball_mipmap">@drawable/ball_blue</item>
        <item name="card_view_mipmap">@drawable/card_northern_kingdoms_landscape_free</item>
        <item name="card_background_color">@color/color_background_northern_kingdoms</item>
        <item name="card_background_gradient">@color/color_gradient_northern_kingdoms</item>
        <item name="card_view_button_background">
            @color/color_card_view_button_background_northern_kingdoms
        </item>
    </style>

    <style name="ScoiataelTheme" parent="BaseTheme">
        <item name="colorPrimary">@color/color_text_scoiatael</item>
        <item name="colorAccent">@color/color_accent_scoiatael</item>
        <item name="card_count_text_color">@color/color_text_scoiatael</item>
        <item name="android:alertDialogIcon">@drawable/icon_round_scoiatael</item>
        <item name="point_ball_mipmap">@drawable/ball_green</item>
        <item name="card_view_mipmap">@drawable/card_scoiatael_landscape_free</item>
        <item name="card_background_color">@color/color_background_scoiatael</item>
        <item name="card_background_gradient">@color/color_gradient_scoiatael</item>
        <item name="card_view_button_background">
            @color/color_card_view_button_background_scoiatael
        </item>
    </style>
</resources>
```

6.105 header_preferences.xml

```
<PreferenceScreen xmlns:app="http://schemas.android.com/apk/res-auto">

    <ListPreference
        app:defaultValue="0"
        app:entries="@array/design_names"
        app:entryValues="@array/design_values"
        app:icon="@drawable/icon_design"
        app:key="@string/preference_key_design"
        app:title="@string/preference_design"
        app:useSimpleSummaryProvider="true" />

    <SwitchPreference
        app:defaultValue="@bool/warning_preference_default"
        app:icon="@drawable/icon_warnings"
        app:key="@string/preference_key_warning"
        app:summary="@string/preference_warnings_desc"
        app:title="@string/preference_warnings_title" />

    <SwitchPreference
        app:defaultValue="@bool/faction_reset_preference_default"
        app:icon="@drawable/icon_factionreset"
        app:key="@string/preference_key_faction_reset"
        app:summary="@string/preference_faction_reset_desc"
        app:title="@string/preference_faction_reset_title" />

    <Preference
        app:fragment="com.peternaggschga.gwent.ui.settings.SettingsSoundFragment"
        app:icon="@drawable/icon_sounds"
        app:title="@string/preference_sounds_header" />

    <PreferenceCategory
        app:allowDividerAbove="true"
        app:key="info"
        app:title="@string/preference_info_category_header">

        <Preference
            app:fragment="com.peternaggschga.gwent.ui.settings.SettingsRuleFragment"
            app:icon="@drawable/icon_rules"
            app:title="@string/preference_rules_header" />

        <Preference
            app:icon="@drawable/icon_explore"
            app:key="@string/preference_key_introduction"
            app:title="@string/preference_info_introduction" />

    </PreferenceCategory>
</PreferenceScreen>
```

6.106 rule_preferences.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:app="http://schemas.android.com/apk/res-auto">
    <Preference
        app:key="@string/preference_rules_general_key"
        app:title="@string/preference_rules_general_title" />

    <Preference
        app:key="@string/preference_rules_course_key"
        app:title="@string/preference_rules_course_title" />

    <Preference
        app:key="@string/preference_rules_factions_key"
        app:title="@string/preference_rules_factions_title" />

    <Preference
        app:key="@string/preference_rules_commander_key"
        app:title="@string/preference_rules_commander_title" />

    <Preference
        app:key="@string/preference_rules_cards_key"
        app:title="@string/preference_rules_cards_title" />

    <Preference
        app:key="@string/preference_rules_card_abilities_key"
        app:title="@string/preference_rules_card_abilities_title" />

    <Preference
        app:key="@string/preference_rules_special_cards_key"
        app:title="@string/preference_rules_special_cards_title" />
</PreferenceScreen>
```

6.107 sound_preferences.xml

```
<PreferenceScreen xmlns:app="http://schemas.android.com/apk/res-auto">

    <SwitchPreference
        app:defaultValue="@bool/sound_preference_default"
        app:iconSpaceReserved="false"
        app:key="@string/preference_key_sounds_all"
        app:title="@string/preference_sounds_header" />

    <PreferenceCategory
        app:allowDividerAbove="@bool/sound_preference_default"
        app:dependency="@string/preference_key_sounds_all"
        app:iconSpaceReserved="false"
        app:title="@string/preference_sounds_category">

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_cards"
            app:title="@string/preference_sounds_cards_new" />

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_reset"
            app:title="@string/preference_sounds_cards_delete" />

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_weather"
            app:title="@string/preference_sounds_weather" />

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_horn"
            app:title="@string/preference_sounds_horn" />

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_burn"
            app:title="@string/preference_sounds_burn" />

        <SwitchPreference
            app:defaultValue="@bool/sound_preference_default"
            app:iconSpaceReserved="false"
            app:key="@string/preference_key_sounds_coin"
            app:title="@string/preference_sounds_coin" />
    </PreferenceCategory>
</PreferenceScreen>
```

Index

/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/background_drawable.xml, 252
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_copy.xml, 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_delete.xml, 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_design.xml, 253
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_explore.xml, 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_factionreset.xml, 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_next.xml, 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_rules.xml, 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_sounds.xml, 254
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/icon_warnings.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/indicator_selected.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/indicator_unselected.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle_black_transparent.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/drawable/rectangle_cancel_popup.xml, 255
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-land-v26/fragment_introduction.xml, 256
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-land/fragment_introduction.xml, 256
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-v26/activity_rules.xml, 258
/home/runner/work/Gwent/Gwent/app/src/main/res/layout-v26/fragment_introduction.xml, 257
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_introduction.xml, 259
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_main.xml, 260
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_rules.xml, 258
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/activity_settings.xml, 262
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/alertdialog_checkbox.xml, 263
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/card.xml, 263
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/fragment_introdu, 257
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_add_card, 264
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_cards.xm, 266
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_nor, 267
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_ste, 267
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_terr, 269
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_coin_vin, 270
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/popup_faction.x, 270
/home/runner/work/Gwent/Gwent/app/src/main/res/layout/row.xml, 272
/home/runner/work/Gwent/Gwent/app/src/main/res/mipmap-anydpi-v26/icon_launcher.xml, 273
/home/runner/work/Gwent/Gwent/app/src/main/res/mipmap-anydpi-v26/icon_launcher_round.xml, 273
/home/runner/work/Gwent/Gwent/app/src/main/res/values-de-rDE/strings.xml, 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/arrays.xml, 273
/home/runner/work/Gwent/Gwent/app/src/main/res/values/attr.xml, 274
/home/runner/work/Gwent/Gwent/app/src/main/res/values/colors.xml, 274
/home/runner/work/Gwent/Gwent/app/src/main/res/values/dimens.xml, 274
/home/runner/work/Gwent/Gwent/app/src/main/res/values/icon_launcher_, 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/preferences.xm, 275
/home/runner/work/Gwent/Gwent/app/src/main/res/values/strings.xml, 279
/home/runner/work/Gwent/Gwent/app/src/main/res/values/styles.xml, 283
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/header_preference, 284
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/rule_preferences.x, 285
/home/runner/work/Gwent/Gwent/app/src/main/res/xml/sound_preference, 285

Ability, 13
BINDING, 13

HORN, 13
MORAL_BOOST, 14
NONE, 14
REVENGE, 14
ability
UnitEntity, 183
abilityImageId
CardUiState, 38
abilityPicker
CardNumberPickerAdapter, 32
abilityView
CardListAdapter.CardViewHolder, 45
accept
MenuUiStateObserver, 102
RowUiStateObserver, 137
activated
Sound, 158
adapteeBuilder
ResetAlertDialogBuilderAdapter, 114
RevengeAlertDialogBuilderAdapter, 123
AddCardDialog, 14
AddCardDialog, 16
disposables, 17
onCreate, 16
pickerAdapter, 17
row, 17
soundManager, 17
addSelectedUnits
CardNumberPickerAdapter, 32
animatedSwitch
ImageViewSwitchAnimator, 87
AppDatabase, 18
rows, 19
units, 19
ARG_SECTION_NUMBER
IntroductionFragment, 93
AVENGER_ABILITY
RevengeAlertDialogBuilderAdapter, 123
AVENGER_DAMAGE
RevengeAlertDialogBuilderAdapter, 123
AVENGER_EPIC
RevengeAlertDialogBuilderAdapter, 123
AVENGER_ROW
RevengeAlertDialogBuilderAdapter, 123
AVENGER_SQUAD
RevengeAlertDialogBuilderAdapter, 123

BACKGROUND
OverlayDialog, 109
ballViews
FactionSwitchListener, 70
BINDING
Ability, 13
bindingView
CardListAdapter.CardViewHolder, 45
BondDamageCalculatorDecorator, 19
BondDamageCalculatorDecorator, 21
calculateDamage, 21
idToSquadSize, 22
isBuffed, 22
BUFFED
DamageCalculator.Color, 52
burn
BurnDialogUseCase, 23
MenuUiState, 100
burnButton
MenuUiStateObserver, 102
BurnDialogUseCase, 22
burn, 23
getBurnUnits, 24
BUTTON_CLICK
ResetDialogUseCase.Trigger, 173

calculateDamage
BondDamageCalculatorDecorator, 21
DamageCalculator, 53
HornDamageCalculatorDecorator, 85
MoralDamageCalculatorDecorator, 105
UnitEntity, 179
WeatherDamageCalculator, 197
calculator
DamageCalculatorBuilder, 59
callback
ChangeFactionDialog, 49
CARD_ABILITIES
RuleSection, 141
CardListAdapter, 26
CardListAdapter, 28
getItemId, 28
onBindViewHolder, 29
onCopy, 29
onCreateViewHolder, 29
onRemove, 29
cardListAdapter
ShowUnitsDialog, 154
CardListAdapter.CardViewHolder, 42
abilityView, 45
bindingView, 45
CardViewHolder, 44
damageView, 45
getAbilityView, 44
getBindingView, 44
getDamageView, 44
itemId, 46
setItemId, 45
CardNumberPickerAdapter, 30
abilityPicker, 32
addSelectedUnits, 32
CardNumberPickerAdapter, 31
damagePicker, 32
EPIC_UNIT_ABILITIES, 33
epicPicker, 33
getDelayedOnValueChangeListener, 32
numberPicker, 33
squadManager, 33
squadPicker, 33
CARDS
RuleSection, 141

CardUiState, 33
- abilityImageId, 38
- CardUiState, 35
- damageBackgroundImageId, 38
- damageString, 38
- damageTextColor, 38
- DIFF_CALLBACK, 38
- equals, 35
- getAbilityImageId, 36
- getDamageBackgroundImageId, 36
- getDamageString, 36
- getDamageTextColor, 36
- getSquadString, 37
- getUnitId, 37
- showAbility, 37
- showSquad, 37
- squadString, 39
- unitId, 39
- UNUSED, 39

CardUiStateFactory, 40
- CardUiStateFactory, 40
- createCardUiState, 41
- damageTextColors, 41
- horn, 42
- weather, 42

CardViewHolder
- CardListAdapter.CardViewHolder, 44

cardViews
- FactionSwitchListener, 71

ChangeFactionDialog, 46
- callback, 49
- ChangeFactionDialog, 48
- getOnThemeClickListener, 48
- onCreate, 48

ChangeFactionDialog.Callback, 25
- onThemeSelect, 26

changeListener
- SoundManager, 163

clearRows
- RowDao, 125

clearWeather
- RowDao, 125
- UnitRepository, 187

CoinFlipDialog, 49
- CoinFlipDialog, 51
- onCreate, 51
- RANDOM, 51

collectionToString
- UnitEntity, 179

COMMANDER
- RuleSection, 141

component
- DamageCalculatorDecorator, 61

copy
- UnitRepository, 187

countUnits
- UnitDao, 174
- UnitRepository, 187

COURSE
- RuleSection, 141

create
- ResetAlertDialogBuilderAdapter, 113
- RevengeAlertDialogBuilderAdapter, 121

createCardUiState
- CardUiStateFactory, 41

createFragment
- SectionsPagerAdapter, 143

createSound
- Sound, 156

CURRENT_FRAGMENT_KEY
- SettingsActivity, 146

currentView
- IndicatorManager, 89

damage
- MenuUiState, 100
- RowUiState, 134
- UnitEntity, 183

damageBackgroundImageId
- CardUiState, 38

DamageCalculator, 52
- calculateDamage, 53
- isBuffed, 54

DamageCalculator.Color, 51
- BUFFED, 52
- DEBUFFED, 52
- DEFAULT, 52

DamageCalculatorBuildDirector, 54
- getCalculator, 55
- setHorns, 56
- setMoralBoosts, 56
- setSquads, 56

DamageCalculatorBuilder, 57
- calculator, 59
- getResult, 58
- setBond, 58
- setHorn, 58
- setMoral, 59
- setWeather, 59

DamageCalculatorDecorator, 59
- component, 61
- DamageCalculatorDecorator, 61

DamageCalculatorUseCase, 61
- getDamageCalculator, 62

damagePicker
- CardNumberPickerAdapter, 32

damageString
- CardUiState, 38

damageTextColor
- CardUiState, 38

damageTextColors
- CardUiStateFactory, 41

DamageValuePicker, 63
- DamageValuePicker, 64
- epicValues, 66
- getDisplayString, 64
- getValue, 64

setEpicValues, 65
setSelectableValues, 65
setValue, 65
damageView
CardListAdapter.CardViewHolder, 45
MenuUiStateObserver, 103
RowUiStateObserver, 138
database
GwentApplication, 82
UnitRepository, 194
DEBUFFED
DamageCalculator.Color, 52
DEFAULT
DamageCalculator.Color, 52
ResetDialogUseCase.DialogType, 67
delete
UnitRepository, 188
deleteUnit
UnitDao, 174
deleteUnits
UnitDao, 175
dialogCallback
ResetAlertDialogBuilderAdapter, 114
DIFF_CALLBACK
CardUiState, 38
dismissViewId
OverlayDialog, 109
disposables
AddCardDialog, 17
MainActivity, 97
ShowUnitsDialog, 154
doubleDamage
HornDamageCalculatorDecorator, 85

epic
UnitEntity, 184
EPIC_DAMAGE_VALUES
UnitEntity, 184
EPIC_UNIT_ABILITIES
CardNumberPickerAdapter, 33
epicPicker
CardNumberPickerAdapter, 33
epicValues
DamageValuePicker, 66
equals
CardUiState, 35
MenuUiState, 99
RowUiState, 133

FACTION_SWITCH
ResetDialogUseCase.Trigger, 173
factionButton
FactionSwitchListener, 71
FACTIONS
RuleSection, 141
FactionSwitchListener, 67
ballViews, 70
cardViews, 71
factionButton, 71
FactionSwitchListener, 69
getListener, 69
onSharedPreferenceChanged, 70
setTheme, 70
THEME_MONSTER, 71
THEME_NILFGAARD, 71
THEME_NORTHERN_KINGDOMS, 71
THEME_PREFERENCE_KEY, 71
THEME_SCOIATAEL, 72
unitViews, 72
factionSwitchListener
MainActivity, 97

GameBoardViewModel, 72
GameBoardViewModel, 74
getMenuUiState, 74
getModel, 74
getRepository, 75
getRowUiState, 75
getSoundManager, 76
INITIALIZER, 79
menuUiState, 79
onBurnButtonPressed, 76
onFactionSwitchReset, 76
onHornViewPressed, 77
onResetButtonPressed, 77
onWeatherButtonPressed, 78
onWeatherViewPressed, 78
reset, 78
rowUiStates, 79
soundManager, 79
gameBoardViewModel
MainActivity, 97
GENERAL
RuleSection, 141
getAbility
UnitEntity, 180
getAbilityImageId
CardUiState, 36
getAbilityView
CardListAdapter.CardViewHolder, 44
getBindingView
CardListAdapter.CardViewHolder, 44
getBurnUnits
BurnDialogUseCase, 24
getCalculator
DamageCalculatorBuildDirector, 55
getDamage
MenuUiState, 99
RowUiState, 133
UnitEntity, 180
getDamageBackgroundImageId
CardUiState, 36
getDamageCalculator
DamageCalculatorUseCase, 62
getDamageString
CardUiState, 36
getDamageTextColor
CardUiState, 36

getDamageView
  CardListAdapter.CardViewHolder, 44
getDelayedOnValueChangeListener
  CardNumberPickerAdapter, 32
getDialog
  ShowUnitsDialog, 153
getDialogType
  ResetDialogUseCase, 116
getDisplayString
  DamageValuePicker, 64
getFirstSquadWithMembers
  SquadManager, 167
getId
  RowEntity, 130
  UnitEntity, 180
getItemCount
  SectionsPagerAdapter, 143
getItemId
  CardListAdapter, 28
getListener
  FactionSwitchListener, 69
getMemberBaseDamage
  SquadState, 170
getMenuUiState
  GameBoardViewModel, 74
getModel
  GameBoardViewModel, 74
getObserver
  RowUiStateObserver, 137
getOnThemeClickListener
  ChangeFactionDialog, 48
getOverlayView
  ImageViewSwitchAnimator, 87
getPreferenceKey
  Sound, 157
getRandomUnit
  ResetRepositoryUseCase, 118
getRepository
  GameBoardViewModel, 75
  GwentApplication, 81
  UnitRepository, 188
getResult
  DamageCalculatorBuilder, 58
getRevengeDialog
  RemoveUnitsUseCase, 110
  ResetRepositoryUseCase, 118
getRow
  ShowUnitsDialog, 153
  UnitEntity, 181
getRowUiState
  GameBoardViewModel, 75
getSectionClickListener
  SettingsRuleFragment, 148
getSoundId
  Sound, 157
getSoundManager
  GameBoardViewModel, 76
getSquad
  UnitEntity, 181
getSquadMembers
  SquadState, 170
getSquadNumber
  SquadState, 170
getSquadString
  CardUiState, 37
getState
  SquadState, 170
getUnit
  UnitDao, 175
  UnitRepository, 189
getUnitId
  CardUiState, 37
getUnits
  RowUiState, 134
  UnitDao, 175
  UnitRepository, 189
getUnitsFlowable
  UnitDao, 176
  UnitRepository, 190
getValue
  DamageValuePicker, 64
Gwent - A Calculator, 1
gwent/data/Ability.java, 199
gwent/data/AppDatabase.java, 200
gwent/data/RowDao.java, 200
gwent/data/RowEntity.java, 200
gwent/data/RowType.java, 201
gwent/data/UnitDao.java, 201
gwent/data/UnitEntity.java, 202
gwent/data/UnitRepository.java, 204
gwent/domain/cases/BurnDialogUseCase.java, 207
gwent/domain/cases/DamageCalculatorUseCase.java, 208
gwent/domain/cases/RemoveUnitsUseCase.java, 209
gwent/domain/cases/ResetAlertDialogBuilderAdapter.java, 210
gwent/domain/cases/ResetDialogUseCase.java, 210
gwent/domain/cases/ResetRepositoryUseCase.java, 212
gwent/domain/cases/RevengeAlertDialogBuilderAdapter.java, 213
gwent/domain/damage/BondDamageCalculatorDecorator.java, 214
gwent/domain/damage/DamageCalculator.java, 214
gwent/domain/damage/DamageCalculatorBuildDirector.java, 214
gwent/domain/damage/DamageCalculatorBuilder.java, 215
gwent/domain/damage/DamageCalculatorDecorator.java, 216
gwent/domain/damage/HornDamageCalculatorDecorator.java, 216
gwent/domain/damage/MoralDamageCalculatorDecorator.java, 216
gwent/domain/damage/WeatherDamageCalculator.java, 217

gwent/GwentApplication.java, 217
gwent/ui/dialogs/addcard/AddCardDialog.java, 218
gwent/ui/dialogs/addcard/CardNumberPickerAdapter.java, 219
gwent/ui/dialogs/addcard/DamageValuePicker.java, 220
gwent/ui/dialogs/addcard/SquadManager.java, 221
gwent/ui/dialogs/addcard/SquadState.java, 222
gwent/ui/dialogs/addcard/StringValuePicker.java, 223
gwent/ui/dialogs/addcard/ValuePicker.java, 224
gwent/ui/dialogs/cards/CardListAdapter.java, 225
gwent/ui/dialogs/cards/CardUiState.java, 227
gwent/ui/dialogs/cards/CardUiStateFactory.java, 228
gwent/ui/dialogs/cards/ShowUnitsDialog.java, 229
gwent/ui/dialogs/ChangeFactionDialog.java, 231
gwent/ui/dialogs/CoinFlipDialog.java, 232
gwent/ui/dialogs/OverlayDialog.java, 232
gwent/ui/introduction/IndicatorManager.java, 233
gwent/ui/introduction/IntroductionActivity.java, 233
gwent/ui/introduction/IntroductionFragment.java, 234
gwent/ui/introduction/SectionsPagerAdapter.java, 235
gwent/ui/main/FactionSwitchListener.java, 235
gwent/ui/main/GameBoardViewModel.java, 237
gwent/ui/main/ImageViewSwitchAnimator.java, 239
gwent/ui/main/MainActivity.java, 240
gwent/ui/main/MenuUiState.java, 243
gwent/ui/main/MenuUiStateObserver.java, 243
gwent/ui/main/RowUiState.java, 244
gwent/ui/main/RowUiStateObserver.java, 245
gwent/ui/settings/RuleActivity.java, 246
gwent/ui/settings/RuleSection.java, 247
gwent/ui/settings/SettingsActivity.java, 247
gwent/ui/settings/SettingsHeaderFragment.java, 248
gwent/ui/settings/SettingsRuleFragment.java, 248
gwent/ui/settings/SettingsSoundFragment.java, 249
gwent/ui/sounds/Sound.java, 250
gwent/ui/sounds/SoundManager.java, 250
GwentApplication, 80
    database, 82
    getRepository, 81
    onCreate, 81
    repository, 82

hasMembers
    SquadState, 171
hasNonEpicUnitsFlowable
    UnitDao, 176
    UnitRepository, 190
HORN
    Ability, 13
horn
    CardUiStateFactory, 42
    RowEntity, 131
    RowUiState, 134
HornDamageCalculatorDecorator, 82
    calculateDamage, 85
    doubleDamage, 85
    HornDamageCalculatorDecorator, 84
    isBuffed, 85
    unitIds, 86
hornView
    RowUiStateObserver, 138

id
    RowEntity, 131
    UnitEntity, 184
idToSquadSize
    BondDamageCalculatorDecorator, 22
imageIds
    IntroductionFragment, 94
ImageViewSwitchAnimator, 86
    animatedSwitch, 87
    getOverlayView, 87
IndicatorManager, 88
    currentView, 89
    IndicatorManager, 88
    indicators, 89
    updateIndicators, 89
indicatorManager
    IntroductionActivity, 91
indicators
    IndicatorManager, 89
inflateCoinFlipPopup
    MainActivity, 96
inflateFactionPopup
    MainActivity, 96
INITIALIZER
    GameBoardViewModel, 79
initializeRows
    UnitRepository, 190
initializeViewModel
    MainActivity, 96
insertAvengers
    RevengeAlertDialogBuilderAdapter, 121
insertRow
    RowDao, 125
insertUnit
    UnitDao, 176
    UnitRepository, 190, 191
INTENT_EXTRA_TAG
    RuleActivity, 140
IntroductionActivity, 90
    indicatorManager, 91
    onCreate, 91
IntroductionFragment, 91
    ARG_SECTION_NUMBER, 93
    imageIds, 94
    newInstance, 93
    onCreateView, 93
    PAGES_COUNT, 94
    stringIds, 94
isActivated
    Sound, 157
isBuffed
    BondDamageCalculatorDecorator, 22
    DamageCalculator, 54
    HornDamageCalculatorDecorator, 85
    MoralDamageCalculatorDecorator, 106
    UnitEntity, 181

WeatherDamageCalculator, 197
isBurn
MenuUiState, 99
isEpic
UnitEntity, 181
isHorn
RowDao, 125
RowEntity, 130
RowUiState, 134
UnitRepository, 192
isHornFlowable
RowDao, 126
UnitRepository, 192
isReset
MenuUiState, 99
isWeather
MenuUiState, 100
RowDao, 126
RowEntity, 130
RowUiState, 134
UnitRepository, 192
isWeatherFlowable
RowDao, 126
UnitRepository, 193
itemId
CardListAdapter.CardViewHolder, 46

layout
OverlayDialog, 109

MainActivity, 94
disposables, 97
factionSwitchListener, 97
gameBoardViewModel, 97
inflateCoinFlipPopup, 96
inflateFactionPopup, 96
initializeViewModel, 96
onCreate, 96
onDestroy, 97
onResume, 97
onWindowFocusChanged, 97
MAX_NR_SQUADS
SquadManager, 168
MELEE
RowType, 132
memberBaseDamage
SquadState, 171
MenuUiState, 98
burn, 100
damage, 100
equals, 99
getDamage, 99
isBurn, 99
isReset, 99
isWeather, 100
MenuUiState, 99
reset, 100
weather, 100
menuUiState
GameBoardViewModel, 79
MenuUiStateObserver, 100
accept, 102
burnButton, 102
damageView, 103
MenuUiStateObserver, 102
resetButton, 103
weatherButton, 103
MONSTER
ResetDialogUseCase.DialogType, 67
MORAL_BOOST
Ability, 14
MoralDamageCalculatorDecorator, 103
calculateDamage, 105
isBuffed, 106
MoralDamageCalculatorDecorator, 105
unitIds, 106

newInstance
IntroductionFragment, 93
NO_DISMISS_VIEW
OverlayDialog, 109
NON_EPIC_DAMAGE_VALUES_UPPER_BOUND
UnitEntity, 184
NONE
Ability, 14
ResetDialogUseCase.DialogType, 67
numberPicker
CardNumberPickerAdapter, 33

onBindViewHolder
CardListAdapter, 29
onBurnButtonPressed
GameBoardViewModel, 76
onCopy
CardListAdapter, 29
onCreate
AddCardDialog, 16
ChangeFactionDialog, 48
CoinFlipDialog, 51
GwentApplication, 81
IntroductionActivity, 91
MainActivity, 96
OverlayDialog, 109
RuleActivity, 139
SettingsActivity, 145
ShowUnitsDialog, 153
onCreatePreferences
SettingsHeaderFragment, 147
SettingsRuleFragment, 149
SettingsSoundFragment, 150
onCreateView
IntroductionFragment, 93
onCreateViewHolder
CardListAdapter, 29
onDestroy
MainActivity, 97
onFactionSwitchReset
GameBoardViewModel, 76

onHornViewPressed
    GameBoardViewModel, 77
onOptionsItemSelected
    RuleActivity, 140
    SettingsActivity, 145
onPreferenceStartFragment
    SettingsActivity, 145
onRemove
    CardListAdapter, 29
onResetButtonPressed
    GameBoardViewModel, 77
onResume
    MainActivity, 97
onSaveInstanceState
    SettingsActivity, 145
onSharedPreferenceChanged
    FactionSwitchListener, 70
onSquadChanged
    SquadManager, 167
onThemeSelect
    ChangeFactionDialog.Callback, 26
OnValueChangeListener< T extends Comparable, 106
onWeatherButtonPressed
    GameBoardViewModel, 78
onWeatherViewPressed
    GameBoardViewModel, 78
onWindowFocusChanged
    MainActivity, 97
OverlayDialog, 107
    BACKGROUND, 109
    dismissViewId, 109
    layout, 109
    NO_DISMISS_VIEW, 109
    onCreate, 109
    OverlayDialog, 108

PAGES_COUNT
    IntroductionFragment, 94
pickerAdapter
    AddCardDialog, 17
playBurnSound
    SoundManager, 161
playCardAddSound
    SoundManager, 161
playCardRemovedSound
    SoundManager, 161
playClearWeatherSound
    SoundManager, 161
playCoinSound
    SoundManager, 161
playHornSound
    SoundManager, 162
playResetSound
    SoundManager, 162
playSound
    SoundManager, 162
playWeatherSound
    SoundManager, 163
preferenceKey
    Sound, 158

RANDOM
    CoinFlipDialog, 51
RANGE
    RowType, 132
remove
    RemoveUnitsUseCase, 111, 112
RemoveUnitsUseCase, 110
    getRevengeDialog, 110
    remove, 111, 112
repository
    GwentApplication, 82
reset
    GameBoardViewModel, 78
    MenuUiState, 100
    ResetAlertDialogBuilderAdapter.Callback, 25
    ResetDialogUseCase, 116
    ResetRepositoryUseCase, 119
    UnitRepository, 193
ResetAlertDialogBuilderAdapter, 112
    adapteeBuilder, 114
    create, 113
    dialogCallback, 114
    ResetAlertDialogBuilderAdapter, 113
    setMonsterDialog, 114
    setTrigger, 114
ResetAlertDialogBuilderAdapter.Callback, 24
    reset, 25
resetButton
    MenuUiStateObserver, 103
ResetDialogUseCase, 115
    getDialogType, 116
    reset, 116
ResetDialogUseCase.DialogType, 66
    DEFAULT, 67
    MONSTER, 67
    NONE, 67
ResetDialogUseCase.Trigger, 172
    BUTTON_CLICK, 173
    FACTION_SWITCH, 173
ResetRepositoryUseCase, 117
    getRandomUnit, 118
    getRevengeDialog, 118
    reset, 119
REVENGE
    Ability, 14
RevengeAlertDialogBuilderAdapter, 120
    adapteeBuilder, 123
    AVENGER_ABILITY, 123
    AVENGER_DAMAGE, 123
    AVENGER_EPIC, 123
    AVENGER_ROW, 123
    AVENGER_SQUAD, 123
    create, 121
    insertAvengers, 121
    RevengeAlertDialogBuilderAdapter, 121
    setNegativeCallback, 122
    setPositiveCallback, 122

row
- AddCardDialog, 17
- ShowUnitsDialog, 154
- UnitEntity, 184

RowDao, 124
- clearRows, 125
- clearWeather, 125
- insertRow, 125
- isHorn, 125
- isHornFlowable, 126
- isWeather, 126
- isWeatherFlowable, 126
- updateHorn, 127
- updateWeather, 127

RowEntity, 127
- getId, 130
- horn, 131
- id, 131
- isHorn, 130
- isWeather, 130
- RowEntity, 128
- setHorn, 130
- setWeather, 130
- weather, 131

rows
- AppDatabase, 19

RowType, 131
- MELEE, 132
- RANGE, 132
- SIEGE, 132

RowUiState, 132
- damage, 134
- equals, 133
- getDamage, 133
- getUnits, 134
- horn, 134
- isHorn, 134
- isWeather, 134
- RowUiState, 133
- units, 134
- weather, 135

RowUiStateObserver, 135
- accept, 137
- damageView, 138
- getObserver, 137
- hornView, 138
- RowUiStateObserver, 136
- unitView, 138
- weatherResource, 138
- weatherView, 138

rowUiStates
- GameBoardViewModel, 79

RuleActivity, 138
- INTENT_EXTRA_TAG, 140
- onCreate, 139
- onOptionsItemSelected, 140

RuleSection, 140
- CARD_ABILITIES, 141
- CARDS, 141
- COMMANDER, 141
- COURSE, 141
- FACTIONS, 141
- GENERAL, 141
- SPECIAL_CARDS, 141

SectionsPagerAdapter, 142
- createFragment, 143
- getItemCount, 143
- SectionsPagerAdapter, 143

setAbility
- UnitEntity, 181

setActivated
- Sound, 157, 158

setBond
- DamageCalculatorBuilder, 58

setDamage
- UnitEntity, 182

setEpic
- UnitEntity, 182

setEpicValues
- DamageValuePicker, 65

setHorn
- DamageCalculatorBuilder, 58
- RowEntity, 130

setHorns
- DamageCalculatorBuildDirector, 56

setId
- UnitEntity, 182

setItemId
- CardListAdapter.CardViewHolder, 45

setMonsterDialog
- ResetAlertDialogBuilderAdapter, 114

setMoral
- DamageCalculatorBuilder, 59

setMoralBoosts
- DamageCalculatorBuildDirector, 56

setNegativeCallback
- RevengeAlertDialogBuilderAdapter, 122

setPositiveCallback
- RevengeAlertDialogBuilderAdapter, 122

setRow
- UnitEntity, 182

setSelectableValues
- DamageValuePicker, 65

setSquad
- UnitEntity, 183

setSquads
- DamageCalculatorBuildDirector, 56

setTheme
- FactionSwitchListener, 70

SettingsActivity, 143
- CURRENT_FRAGMENT_KEY, 146
- onCreate, 145
- onOptionsItemSelected, 145
- onPreferenceStartFragment, 145
- onSaveInstanceState, 145

SettingsHeaderFragment, 146

onCreatePreferences, 147
SettingsRuleFragment, 147
getSectionClickListener, 148
onCreatePreferences, 149
SettingsSoundFragment, 149
onCreatePreferences, 150
setTrigger
ResetAlertDialogBuilderAdapter, 114
setValue
DamageValuePicker, 65
setWeather
DamageCalculatorBuilder, 59
RowEntity, 130
showAbility
CardUiState, 37
showSquad
CardUiState, 37
ShowUnitsDialog, 151
cardListAdapter, 154
disposables, 154
getDialog, 153
getRow, 153
onCreate, 153
row, 154
ShowUnitsDialog, 153
soundManager, 155
unitRecyclerView, 155
SIEGE
RowType, 132
Sound, 155
activated, 158
createSound, 156
getPreferenceKey, 157
getSoundId, 157
isActivated, 157
preferenceKey, 158
setActivated, 157, 158
Sound, 156
soundId, 158
SOUND_BURN
SoundManager, 163
SOUND_CARDS_EPIC
SoundManager, 163
SOUND_CARDS_MELEE
SoundManager, 163
SOUND_CARDS_RANGE
SoundManager, 164
SOUND_CARDS_SIEGE
SoundManager, 164
SOUND_COIN
SoundManager, 164
SOUND_HORN
SoundManager, 164
SOUND_RESET
SoundManager, 164
SOUND_WEATHER_FOG
SoundManager, 165
SOUND_WEATHER_FROST
SoundManager, 165
SOUND_WEATHER_GOOD
SoundManager, 165
SOUND_WEATHER_RAIN
SoundManager, 165
soundId
Sound, 158
SoundManager, 158
changeListener, 163
playBurnSound, 161
playCardAddSound, 161
playCardRemovedSound, 161
playClearWeatherSound, 161
playCoinSound, 161
playHornSound, 162
playResetSound, 162
playSound, 162
playWeatherSound, 163
SOUND_BURN, 163
SOUND_CARDS_EPIC, 163
SOUND_CARDS_MELEE, 163
SOUND_CARDS_RANGE, 164
SOUND_CARDS_SIEGE, 164
SOUND_COIN, 164
SOUND_HORN, 164
SOUND_RESET, 164
SOUND_WEATHER_FOG, 165
SOUND_WEATHER_FROST, 165
SOUND_WEATHER_GOOD, 165
SOUND_WEATHER_RAIN, 165
SoundManager, 160
soundPool, 165
sounds, 166
soundManager
AddCardDialog, 17
GameBoardViewModel, 79
ShowUnitsDialog, 155
soundPool
SoundManager, 165
sounds
SoundManager, 166
SPECIAL_CARDS
RuleSection, 141
squad
UnitEntity, 184
SquadManager, 166
getFirstSquadWithMembers, 167
MAX_NR_SQUADS, 168
onSquadChanged, 167
SquadManager, 167
states, 168
squadManager
CardNumberPickerAdapter, 33
squadMembers
SquadState, 171
squadNumber
SquadState, 171
squadPicker

CardNumberPickerAdapter, 33
SquadState, 168
getMemberBaseDamage, 170
getSquadMembers, 170
getSquadNumber, 170
getState, 170
hasMembers, 171
memberBaseDamage, 171
squadMembers, 171
squadNumber, 171
SquadState, 169
squadString
CardUiState, 39
states
SquadManager, 168
stringIds
IntroductionFragment, 94
StringValuePicker< T extends Comparable, 172
switchHorn
UnitRepository, 194
switchWeather
UnitRepository, 194

THEME_MONSTER
FactionSwitchListener, 71
THEME_NILFGAARD
FactionSwitchListener, 71
THEME_NORTHERN_KINGDOMS
FactionSwitchListener, 71
THEME_PREFERENCE_KEY
FactionSwitchListener, 71
THEME_SCOIATAEL
FactionSwitchListener, 72
toString
UnitEntity, 183

UnitDao, 173
countUnits, 174
deleteUnit, 174
deleteUnits, 175
getUnit, 175
getUnits, 175
getUnitsFlowable, 176
hasNonEpicUnitsFlowable, 176
insertUnit, 176
UnitEntity, 177
ability, 183
calculateDamage, 179
collectionToString, 179
damage, 183
epic, 184
EPIC_DAMAGE_VALUES, 184
getAbility, 180
getDamage, 180
getId, 180
getRow, 181
getSquad, 181
id, 184
isBuffed, 181
isEpic, 181
NON_EPIC_DAMAGE_VALUES_UPPER_BOUND, 184
row, 184
setAbility, 181
setDamage, 182
setEpic, 182
setId, 182
setRow, 182
setSquad, 183
squad, 184
toString, 183
UnitEntity, 179
unitId
CardUiState, 39
unitIds
HornDamageCalculatorDecorator, 86
MoralDamageCalculatorDecorator, 106
unitRecyclerView
ShowUnitsDialog, 155
UnitRepository, 184
clearWeather, 187
copy, 187
countUnits, 187
database, 194
delete, 188
getRepository, 188
getUnit, 189
getUnits, 189
getUnitsFlowable, 190
hasNonEpicUnitsFlowable, 190
initializeRows, 190
insertUnit, 190, 191
isHorn, 192
isHornFlowable, 192
isWeather, 192
isWeatherFlowable, 193
reset, 193
switchHorn, 194
switchWeather, 194
UnitRepository, 186
units
AppDatabase, 19
RowUiState, 134
unitView
RowUiStateObserver, 138
unitViews
FactionSwitchListener, 72
UNUSED
CardUiState, 39
updateHorn
RowDao, 127
updateIndicators
IndicatorManager, 89
updateWeather
RowDao, 127

ValuePicker< T extends Comparable, 195

weather
- CardUiStateFactory, 42
- MenuUiState, 100
- RowEntity, 131
- RowUiState, 135
- WeatherDamageCalculator, 198

weatherButton
- MenuUiStateObserver, 103

WeatherDamageCalculator, 195
- calculateDamage, 197
- isBuffed, 197
- weather, 198
- WeatherDamageCalculator, 197

weatherResource
- RowUiStateObserver, 138

weatherView
- RowUiStateObserver, 138