

Kalkulator statystyk postaci na potrzeby gry RPG

Generated by Doxygen 1.9.1

1 Todo List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Class Documentation	9
5.1 appApp Class Reference	9
5.1.1 Detailed Description	10
5.1.2 Member Function Documentation	10
5.1.2.1 OnInit()	10
5.2 appFrame Class Reference	10
5.2.1 Detailed Description	11
5.2.2 Constructor & Destructor Documentation	11
5.2.2.1 appFrame()	12
5.2.3 Member Function Documentation	12
5.2.3.1 OnAddCharacter()	12
5.2.4 Member Data Documentation	13
5.2.4.1 addCharPanel	13
5.2.4.2 charNameTextCtrl	13
5.2.4.3 gmPanel	13
5.2.4.4 notebook	14
5.3 appGMPanel Class Reference	14
5.3.1 Detailed Description	15
5.3.2 Constructor & Destructor Documentation	15
5.3.2.1 appGMPanel()	16
5.3.3 Member Function Documentation	16
5.3.3.1 OnAddEquipment()	16
5.3.3.2 OnAddSlot()	16
5.3.3.3 OnAddStat()	17
5.3.3.4 OnAddState()	17
5.3.3.5 UpdateEqListCtrl()	18
5.3.3.6 UpdateSlotListCtrl()	18
5.3.3.7 UpdateStateListCtrl()	19
5.3.3.8 UpdateStatsListCtrl()	20
5.3.3.9 wxDECLARE_EVENT_TABLE()	21
5.3.4 Member Data Documentation	21
5.3.4.1 addElementPanel	21
5.3.4.2 addElementSizer	21

5.3.4.3 addEquipmentButton	21
5.3.4.4 addSlotButton	21
5.3.4.5 addStatButton	21
5.3.4.6 addStateButton	22
5.3.4.7 eqListCtrl	22
5.3.4.8 eqPanel	22
5.3.4.9 eqSizer	22
5.3.4.10 gmNotebook	22
5.3.4.11 m_gamedata	22
5.3.4.12 slotListCtrl	23
5.3.4.13 slotPanel	23
5.3.4.14 slotSizer	23
5.3.4.15 stateListCtrl	23
5.3.4.16 statePanel	23
5.3.4.17 stateSizer	23
5.3.4.18 statsListCtrl	24
5.3.4.19 statsPanel	24
5.3.4.20 statsSizer	24
5.4 appPlayerPanel Class Reference	24
5.4.1 Detailed Description	26
5.4.2 Constructor & Destructor Documentation	26
5.4.2.1 appPlayerPanel()	26
5.4.3 Member Function Documentation	26
5.4.3.1 OnAddEquipment()	26
5.4.3.2 OnAddStat()	26
5.4.3.3 OnAddState()	26
5.4.3.4 OnEquipEquipment()	27
5.4.3.5 OnUnequipEquipment()	27
5.4.3.6 UpdateEQListCtrl()	27
5.4.3.7 UpdateEquippedEQListCtrl()	27
5.4.3.8 UpdateStatesListCtrl()	27
5.4.3.9 UpdateStatsListCtrl()	28
5.4.3.10 wxDECLARE_EVENT_TABLE()	28
5.4.4 Member Data Documentation	28
5.4.4.1 addEquipmentButton	28
5.4.4.2 addPanel	28
5.4.4.3 addPanelSizer	28
5.4.4.4 addStatButton	28
5.4.4.5 addStateButton	29
5.4.4.6 eqListCtrl	29
5.4.4.7 eqPanel	29
5.4.4.8 eqSizer	29

5.4.4.9 equipEquipmentButton	29
5.4.4.10 equippedEQListCtrl	29
5.4.4.11 equippedEQPanel	30
5.4.4.12 equippedEQSizer	30
5.4.4.13 mainSizer	30
5.4.4.14 playerNotebook	30
5.4.4.15 statesListCtrl	30
5.4.4.16 statesPanel	30
5.4.4.17 statesSizer	31
5.4.4.18 statsListCtrl	31
5.4.4.19 statsPanel	31
5.4.4.20 statsSizer	31
5.4.4.21 unequipEquipmentButton	31
5.5 BasicGameData Class Reference	32
5.5.1 Detailed Description	33
5.5.2 Member Typedef Documentation	33
5.5.2.1 id_t	33
5.5.3 Constructor & Destructor Documentation	33
5.5.3.1 BasicGameData()	33
5.5.4 Member Function Documentation	34
5.5.4.1 getDescription()	34
5.5.4.2 getId()	34
5.5.4.3 getName()	35
5.5.4.4 isId()	35
5.5.4.5 setDescription()	36
5.5.4.6 setId()	36
5.5.4.7 setName()	37
5.5.4.8 validateIntegrity()	37
5.5.5 Friends And Related Function Documentation	38
5.5.5.1 GameData	38
5.5.5.2 GameMetadata	38
5.5.6 Member Data Documentation	38
5.5.6.1 INVALID_ID	39
5.5.6.2 m_description	39
5.5.6.3 m_id	39
5.5.6.4 m_name	39
5.6 Character Class Reference	40
5.6.1 Detailed Description	42
5.6.2 Member Typedef Documentation	42
5.6.2.1 equipment_t	42
5.6.2.2 inventory_t	42
5.6.2.3 itemQuantity_t	42

5.6.2.4	states_t	42
5.6.2.5	statValueContributors_t	43
5.6.2.6	statValues_t	43
5.6.3	Constructor & Destructor Documentation	43
5.6.3.1	Character()	43
5.6.4	Member Function Documentation	43
5.6.4.1	addItem()	43
5.6.4.2	addState()	44
5.6.4.3	equipItem()	45
5.6.4.4	getBaseStatValue()	46
5.6.4.5	getEquipedItem() [1/2]	47
5.6.4.6	getEquipedItem() [2/2]	48
5.6.4.7	getEquipment()	49
5.6.4.8	getGameData()	50
5.6.4.9	getInventory()	50
5.6.4.10	getStates()	51
5.6.4.11	getStatValue()	51
5.6.4.12	getStatValueContributors()	52
5.6.4.13	getStatValueEquipmentContributors()	53
5.6.4.14	getStatValueStatesContributors()	54
5.6.4.15	isEquipmentSlotUsed()	54
5.6.4.16	setBaseStatValue()	55
5.6.4.17	validateDataIntegrity()	56
5.6.5	Member Data Documentation	57
5.6.5.1	m_baseStatValues	57
5.6.5.2	m_equipment	57
5.6.5.3	m_gameData	58
5.6.5.4	m_inventory	58
5.6.5.5	m_states	58
5.7	EquipmentSlot Class Reference	59
5.7.1	Detailed Description	60
5.7.2	Member Typedef Documentation	60
5.7.2.1	id_t	60
5.7.3	Constructor & Destructor Documentation	61
5.7.3.1	EquipmentSlot()	61
5.7.4	Member Function Documentation	61
5.7.4.1	getDescription()	61
5.7.4.2	getId()	61
5.7.4.3	getName()	62
5.7.4.4	isId()	62
5.7.4.5	setDescription()	63
5.7.4.6	setId()	64

5.7.4.7 setName()	64
5.7.4.8 validateIntegrity()	65
5.7.5 Member Data Documentation	65
5.7.5.1 INVALID_ID	65
5.7.5.2 m_description	66
5.7.5.3 m_id	66
5.7.5.4 m_name	66
5.8 exceptionEquipmentSlotOccupied Class Reference	67
5.8.1 Detailed Description	67
5.8.2 Member Function Documentation	68
5.8.2.1 what()	68
5.9 exceptionEquipmentSlotUnused Class Reference	68
5.9.1 Detailed Description	69
5.9.2 Member Function Documentation	69
5.9.2.1 what()	69
5.10 exceptionIllegalId Class Reference	70
5.10.1 Detailed Description	70
5.10.2 Member Function Documentation	71
5.10.2.1 what()	71
5.11 exceptionInvalidGameMetadata Class Reference	71
5.11.1 Detailed Description	72
5.11.2 Member Function Documentation	72
5.11.2.1 what()	72
5.12 exceptionNonExistingId Class Reference	73
5.12.1 Detailed Description	73
5.12.2 Member Function Documentation	74
5.12.2.1 what()	74
5.13 exceptionEquipmentSlotIllegalUsage Class Reference	74
5.13.1 Detailed Description	75
5.13.2 Member Function Documentation	75
5.13.2.1 what()	75
5.14 exceptionGameDataMismatch Class Reference	76
5.14.1 Detailed Description	76
5.14.2 Member Function Documentation	77
5.14.2.1 what()	77
5.15 GameData Class Reference	77
5.15.1 Detailed Description	79
5.15.2 Member Typedef Documentation	79
5.15.2.1 equipmentSlotsCollection_t	79
5.15.2.2 itemcollection_t	80
5.15.2.3 stateCollcetion_t	80
5.15.2.4 statsCollection_t	80

5.15.3 Constructor & Destructor Documentation	80
5.15.3.1 ~GameData()	80
5.15.4 Member Function Documentation	80
5.15.4.1 addEquipmentSlot()	80
5.15.4.2 addItem()	81
5.15.4.3 addStat()	83
5.15.4.4 addState()	83
5.15.4.5 getEquipmentSlot()	84
5.15.4.6 getEquipmentSlots()	85
5.15.4.7 getItem()	86
5.15.4.8 getItems()	87
5.15.4.9 getStat()	87
5.15.4.10 getState()	88
5.15.4.11 getStates()	89
5.15.4.12 getStats()	90
5.15.4.13 validateDataIntegrity()	90
5.15.5 Member Data Documentation	91
5.15.5.1 Item	91
5.15.5.2 m_equipmentSlots	91
5.15.5.3 m_items	92
5.15.5.4 m_nextEquipmentSlotId	92
5.15.5.5 m_nextItemId	92
5.15.5.6 m_nextStateId	92
5.15.5.7 m_nextStatId	93
5.15.5.8 m_states	93
5.15.5.9 m_stats	93
5.16 GameMetadata Class Reference	93
5.16.1 Detailed Description	94
5.16.2 Member Typedef Documentation	94
5.16.2.1 equipmentSlotsCollection_t	95
5.16.2.2 statsCollection_t	95
5.16.3 Constructor & Destructor Documentation	95
5.16.3.1 ~GameMetadata()	95
5.16.4 Member Function Documentation	95
5.16.4.1 addEquipmentSlot()	95
5.16.4.2 addStat()	96
5.16.4.3 getEquipmentSlot()	97
5.16.4.4 getEquipmentSlots()	98
5.16.4.5 getStat()	99
5.16.4.6 getStats()	100
5.16.5 Member Data Documentation	100
5.16.5.1 m_equipmentSlots	101

5.16.5.2 m_nextEquipmentSlotId	101
5.16.5.3 m_nextStatId	101
5.16.5.4 m_stats	101
5.17 Item Class Reference	102
5.17.1 Detailed Description	104
5.17.2 Member Typedef Documentation	104
5.17.2.1 equipableSlots_t	104
5.17.2.2 id_t	104
5.17.2.3 modifiersCollection_t	105
5.17.2.4 modifier_t	105
5.17.3 Constructor & Destructor Documentation	105
5.17.3.1 Item()	105
5.17.4 Member Function Documentation	106
5.17.4.1 addModifier()	106
5.17.4.2 getDescription()	107
5.17.4.3 getEquipableSlots()	107
5.17.4.4 getGameMetadata()	108
5.17.4.5 getId()	108
5.17.4.6 getModifiers()	109
5.17.4.7 getModifierValue()	109
5.17.4.8 getName()	110
5.17.4.9 isEquipableOn()	110
5.17.4.10 isId()	111
5.17.4.11 setDescription()	112
5.17.4.12 setEquipableOn()	112
5.17.4.13 setId()	113
5.17.4.14 setName()	114
5.17.4.15 unsetEquipableOn()	114
5.17.4.16 validateIntegrity()	115
5.17.5 Member Data Documentation	116
5.17.5.1 INVALID_ID	116
5.17.5.2 m_description	116
5.17.5.3 m_equipableOn	116
5.17.5.4 m_gameMetadata	116
5.17.5.5 m_id	117
5.17.5.6 m_modifiers	117
5.17.5.7 m_name	117
5.18 Stat Class Reference	118
5.18.1 Detailed Description	119
5.18.2 Member Typedef Documentation	119
5.18.2.1 id_t	120
5.18.2.2 value_t	120

5.18.3 Constructor & Destructor Documentation	120
5.18.3.1 Stat()	120
5.18.4 Member Function Documentation	120
5.18.4.1 getDescription()	120
5.18.4.2 getId()	121
5.18.4.3 getName()	121
5.18.4.4 isId()	121
5.18.4.5 setDescription()	122
5.18.4.6 setId()	123
5.18.4.7 setName()	124
5.18.4.8 validateIntegrity()	125
5.18.5 Member Data Documentation	125
5.18.5.1 INVALID_ID	126
5.18.5.2 m_description	126
5.18.5.3 m_id	126
5.18.5.4 m_name	126
5.19 State Class Reference	127
5.19.1 Detailed Description	129
5.19.2 Member Typedef Documentation	129
5.19.2.1 id_t	129
5.19.2.2 modifiersCollection_t	129
5.19.2.3 modifier_t	129
5.19.3 Constructor & Destructor Documentation	129
5.19.3.1 State()	129
5.19.4 Member Function Documentation	130
5.19.4.1 addModifier()	130
5.19.4.2 getDescription()	131
5.19.4.3 getGameMetadata()	131
5.19.4.4 getId()	132
5.19.4.5 getModifiers()	133
5.19.4.6 getModifierValue()	133
5.19.4.7 getName()	134
5.19.4.8 isId()	134
5.19.4.9 setDescription()	135
5.19.4.10 setId()	135
5.19.4.11 setName()	136
5.19.4.12 validateIntegrity()	136
5.19.5 Member Data Documentation	137
5.19.5.1 INVALID_ID	137
5.19.5.2 m_description	137
5.19.5.3 m_gameMetadata	138
5.19.5.4 m_id	138

5.19.5.5 m_modifiers	138
5.19.5.6 m_name	138
5.20 StatModifyingEntity Class Reference	139
5.20.1 Detailed Description	141
5.20.2 Member Typedef Documentation	141
5.20.2.1 id_t	141
5.20.2.2 modifiersCollection_t	141
5.20.2.3 modifier_t	141
5.20.3 Constructor & Destructor Documentation	141
5.20.3.1 StatModifyingEntity()	141
5.20.4 Member Function Documentation	142
5.20.4.1 addModifier()	142
5.20.4.2 getDescription()	143
5.20.4.3 getGameMetadata()	143
5.20.4.4 getId()	144
5.20.4.5 getModifiers()	145
5.20.4.6 getModifierValue()	145
5.20.4.7 getName()	146
5.20.4.8 isId()	146
5.20.4.9 setDescription()	147
5.20.4.10 setId()	147
5.20.4.11 setName()	148
5.20.4.12 validateIntegrity()	148
5.20.5 Member Data Documentation	149
5.20.5.1 INVALID_ID	149
5.20.5.2 m_description	149
5.20.5.3 m_gameMetadata	150
5.20.5.4 m_id	150
5.20.5.5 m_modifiers	150
5.20.5.6 m_name	150
6 File Documentation	151
6.1 appApp.cpp File Reference	151
6.1.1 Function Documentation	151
6.1.1.1 wxIMPLEMENT_APP()	152
6.2 appApp.hpp File Reference	152
6.3 appGM.cpp File Reference	153
6.3.1 Function Documentation	153
6.3.1.1 wxBEGIN_EVENT_TABLE()	153
6.4 appGM.hpp File Reference	154
6.4.1 Enumeration Type Documentation	155
6.4.1.1 anonymous enum	155

6.5 appMain.cpp File Reference	156
6.6 appMain.hpp File Reference	156
6.6.1 Enumeration Type Documentation	157
6.6.1.1 anonymous enum	157
6.7 appPlayer.cpp File Reference	157
6.7.1 Function Documentation	158
6.7.1.1 wxBEGIN_EVENT_TABLE()	158
6.8 appPlayer.hpp File Reference	159
6.8.1 Enumeration Type Documentation	160
6.8.1.1 anonymous enum	160
6.9 basicGamedata.cpp File Reference	161
6.9.1 Detailed Description	161
6.10 basicGamedata.hpp File Reference	161
6.10.1 Detailed Description	162
6.11 character.cpp File Reference	163
6.11.1 Detailed Description	163
6.12 character.hpp File Reference	163
6.12.1 Detailed Description	165
6.13 equipmentSlot.cpp File Reference	165
6.13.1 Detailed Description	165
6.14 equipmentSlot.hpp File Reference	165
6.14.1 Detailed Description	166
6.15 gameData.cpp File Reference	167
6.15.1 Detailed Description	167
6.16 gameData.hpp File Reference	167
6.16.1 Detailed Description	168
6.17 gameMetadata.cpp File Reference	168
6.17.1 Detailed Description	168
6.18 gameMetadata.hpp File Reference	169
6.18.1 Detailed Description	170
6.19 item.cpp File Reference	170
6.19.1 Detailed Description	170
6.20 item.hpp File Reference	171
6.20.1 Detailed Description	172
6.21 stat.cpp File Reference	172
6.21.1 Detailed Description	172
6.22 stat.hpp File Reference	172
6.22.1 Detailed Description	173
6.23 state.cpp File Reference	174
6.23.1 Detailed Description	174
6.24 state.hpp File Reference	174
6.24.1 Detailed Description	176

6.25 statModifyingEntity.cpp File Reference	176
6.25.1 Detailed Description	176
6.26 statModifyingEntity.hpp File Reference	176
6.26.1 Detailed Description	177
Index	179

Chapter 1

Todo List

Member `Character::addItem` (`const Item *const item`, `itemQuantity_t quantity=1`)

make it remove item from inventory if quantity would go below or equal 0.

Member `GameData::addItem` (`Item *item`)

LOW change exception to it's own exception class

Member `StatModifyingEntity::addModifier` (`Stat::id_t`, `Stat::value_t` by)

Check if entity has modifier of that `Stat` already.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

BasicGameData	32
EquipmentSlot	59
Stat	118
StatModifyingEntity	139
Item	102
State	127
Character	40
std::exception	
exceptionEquipmentSlotOccupied	67
exceptionEquipmentSlotUnused	68
exceptionIllegalId	70
exceptionInvalidGameMetadata	71
exceptionNonExistingId	73
excpetionEquipmentSlotIllegalUsage	74
excpetionGameDataMismatch	76
GameMetadata	93
GameData	77
wxApp	
appApp	9
wxFrame	
appFrame	10
wxPanel	
appGMPanel	14
appPlayerPanel	24

Chapter 3

Class Index

3.1 Class List

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

appApp	9
appFrame	10
appGMPanel	14
appPlayerPanel	24
BasicGameData	
Basic information about game	32
Character	
Represents character	40
EquipmentSlot	
Equipment slot representation	59
exceptionEquipmentSlotOccupied	
Tried to perform operation on occupied already slot	67
exceptionEquipmentSlotUnused	
Tried to extract data from unused slot	68
exceptionIllegalId	
Illegal id exception	70
exceptionInvalidGameMetadata	
Invalid game data	71
exceptionNonExistingId	
Exception	73
excpetionEquipmentSlotIllegalUsage	
Tried to put stuff where it is not supposed to go	74
excpetionGameDataMismatch	
Thrown when attempted to do operation that requires 2 objects to use common GameData but different were used	76
GameData	
On top of what GameMetadata does. Holds items cataloge	77
GameMetadata	
Holds game metadata. That is what statistics exist and what equipable slots exist	93
Item	
Represents an item in game	102
Stat	
Statistics	118
State	
Reperesents State in game	127
StatModifyingEntity	
Represents collection of stat modifiers	139

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

appApp.cpp	151
appApp.hpp	152
appGM.cpp	153
appGM.hpp	154
appMain.cpp	156
appMain.hpp	156
appPlayer.cpp	157
appPlayer.hpp	159
basicGamedata.cpp	
BasicGameData implementation	161
basicGamedata.hpp	
BasicGameData interface	161
character.cpp	
Character implementation	163
character.hpp	
Character interface	163
equipmentSlot.cpp	
EquipmentSlot implementation	165
equipmentSlot.hpp	
EquipmentSlot interface	165
gameData.cpp	
GameData implementation	167
gameData.hpp	
GameData interface	167
gameMetadata.cpp	
GameMetadata implementation	168
gameMetadata.hpp	
GameMetadata interface	169
item.cpp	
Item implementation	170
item.hpp	
Item interface	171
stat.cpp	
Stat implementation	172
stat.hpp	
Stat interface	172

state.cpp	
State implementation	174
state.hpp	
State interface	174
statModifyingEntity.cpp	
StatModifyingEntity implementation	176
statModifyingEntity.hpp	
StatModifyingEntity interface	176

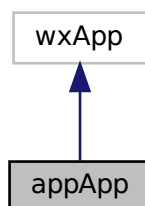
Chapter 5

Class Documentation

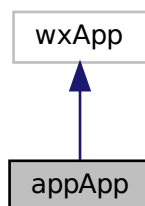
5.1 appApp Class Reference

```
#include <appApp.hpp>
```

Inheritance diagram for appApp:



Collaboration diagram for appApp:



Public Member Functions

- virtual bool [OnInit](#) ()

5.1.1 Detailed Description

Definition at line 16 of file appApp.hpp.

5.1.2 Member Function Documentation

5.1.2.1 OnInit()

```
bool appApp::OnInit ( ) [virtual]
```

Definition at line 15 of file appApp.cpp.

```
15     {  
16     appFrame *frame = new appFrame("My Application");  
17     frame->Show(true);  
18     return true;  
19 }
```

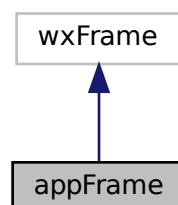
The documentation for this class was generated from the following files:

- [appApp.hpp](#)
- [appApp.cpp](#)

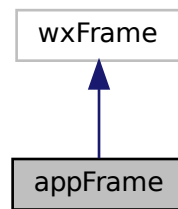
5.2 appFrame Class Reference

```
#include <appMain.hpp>
```

Inheritance diagram for appFrame:



Collaboration diagram for appFrame:



Public Member Functions

- [appFrame](#) (const wxString &title)

Private Member Functions

- void [OnAddCharacter](#) (wxCommandEvent &event)

Private Attributes

- wxNotebook * [notebook](#)
- wxPanel * [gmPanel](#)
- wxPanel * [addCharPanel](#)
- wxTextCtrl * [charNameTextCtrl](#)

5.2.1 Detailed Description

Definition at line 20 of file appMain.hpp.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 appFrame()

```
appFrame::appFrame (
    const wxString & title )
```

Definition at line 12 of file appMain.cpp.

```
13 : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(800, 600)) {
14     wxBoxSizer *sizer = new wxBoxSizer(wxVERTICAL);
15
16     notebook = new wxNotebook(this, wxID_ANY);
17
18     gmPanel = new appGMPanel(notebook);
19     notebook->AddPage(gmPanel, "gamemaster");
20
21     addCharPanel = new wxPanel(notebook, wxID_ANY);
22     wxBoxSizer *addCharSizer = new wxBoxSizer(wxVERTICAL);
23     wxStaticText *charNameLabel =
24         new wxStaticText(addCharPanel, wxID_ANY, "Character's Name:");
25     charNameTextCtrl = new wxTextCtrl(addCharPanel, wxID_ANY);
26     wxButton *addButton = new wxButton(addCharPanel, ID_ADD_CHARACTER, "Add");
27
28     addCharSizer->Add(charNameLabel, 0, wxALIGN_LEFT | wxALL, 5);
29     addCharSizer->Add(charNameTextCtrl, 0, wxEXPAND | wxALL, 5);
30     addCharSizer->Add(addButton, 0, wxALIGN_LEFT | wxALL, 5);
31     addCharPanel->SetSizer(addCharSizer);
32
33     notebook->AddPage(addCharPanel, "Add character");
34
35     sizer->Add(notebook, 1, wxEXPAND);
36     SetSizer(sizer);
37
38     addButton->Bind(wxEVT_BUTTON, &appFrame::OnAddCharacter, this);
39 }
```

References `addCharPanel`, `charNameTextCtrl`, `gmPanel`, `ID_ADD_CHARACTER`, `notebook`, and `OnAddCharacter()`.

Here is the call graph for this function:



5.2.3 Member Function Documentation

5.2.3.1 OnAddCharacter()

```
void appFrame::OnAddCharacter (
    wxCommandEvent & event ) [private]
```

Definition at line 41 of file appMain.cpp.

```
41 {
42     wxString charName = charNameTextCtrl->GetValue();
43
44     if (charName.IsEmpty()) {
45         wxMessageBox("Character's Name is empty", "Error", wxOK | wxICON_ERROR);
46         return;
47     }
```

```
48
49 wxPanel *newCharPanel = new appPlayerPanel(notebook, charName);
50 notebook->AddPage(newCharPanel, charName);
51
52 charNameTextCtrl->Clear();
53 }
```

References charNameTextCtrl, and notebook.

Referenced by appFrame().

Here is the caller graph for this function:



5.2.4 Member Data Documentation

5.2.4.1 addCharPanel

```
wxPanel* appFrame::addCharPanel [private]
```

Definition at line 27 of file appMain.hpp.

Referenced by appFrame().

5.2.4.2 charNameTextCtrl

```
wxTextCtrl* appFrame::charNameTextCtrl [private]
```

Definition at line 28 of file appMain.hpp.

Referenced by appFrame(), and OnAddCharacter().

5.2.4.3 gmPanel

```
wxPanel* appFrame::gmPanel [private]
```

Definition at line 26 of file appMain.hpp.

Referenced by appFrame().

5.2.4.4 notebook

```
wxNotebook* appFrame::notebook [private]
```

Definition at line 25 of file appMain.hpp.

Referenced by appFrame(), and OnAddCharacter().

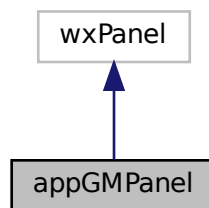
The documentation for this class was generated from the following files:

- [appMain.hpp](#)
- [appMain.cpp](#)

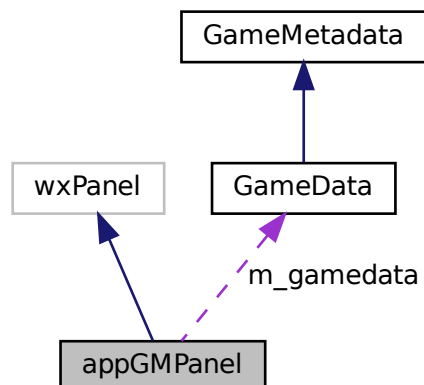
5.3 appGMPanel Class Reference

```
#include <appGM.hpp>
```

Inheritance diagram for appGMPanel:



Collaboration diagram for appGMPanel:



Public Member Functions

- [appGMPanel](#) (wxNotebook *parent)

Private Member Functions

- void [OnAddStat](#) (wxCommandEvent &event)
- void [OnAddEquipment](#) (wxCommandEvent &event)
- void [OnAddSlot](#) (wxCommandEvent &event)
- void [OnAddState](#) (wxCommandEvent &event)
- void [UpdateStatsListCtrl](#) ()
- void [UpdateEqListCtrl](#) ()
- void [UpdateSlotListCtrl](#) ()
- void [UpdateStateListCtrl](#) ()
- [wxDECLARE_EVENT_TABLE](#) ()

Private Attributes

- wxNotebook * [gmNotebook](#)
- wxPanel * [addElementPanel](#)
- wxPanel * [statsPanel](#)
- wxPanel * [eqPanel](#)
- wxPanel * [slotPanel](#)
- wxPanel * [statePanel](#)
- wxBoxSizer * [addElementSizer](#)
- wxBoxSizer * [statsSizer](#)
- wxBoxSizer * [eqSizer](#)
- wxBoxSizer * [slotSizer](#)
- wxBoxSizer * [stateSizer](#)
- wxListCtrl * [statsListCtrl](#)
- wxListCtrl * [eqListCtrl](#)
- wxListCtrl * [slotListCtrl](#)
- wxListCtrl * [stateListCtrl](#)
- wxButton * [addStatButton](#)
- wxButton * [addEquipmentButton](#)
- wxButton * [addSlotButton](#)
- wxButton * [addStateButton](#)
- [GameData](#) * [m_gamedata](#) {nullptr}

5.3.1 Detailed Description

Definition at line 13 of file appGM.hpp.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 appGMPanel()

```
appGMPanel::appGMPanel (
    wxNotebook * parent )
```

5.3.3 Member Function Documentation

5.3.3.1 OnAddEquipment()

```
void appGMPanel::OnAddEquipment (
    wxCommandEvent & event ) [private]
```

Definition at line 95 of file appGM.cpp.

```
95     {
96         /*wxString name = wxGetTextFromUser("Enter name:", "Add Item");
97         wxString description = wxGetTextFromUser("Enter description:", "Add Item");
98
99         if (name.IsEmpty()) {
100             wxMessageBox("Name or description cannot be empty!", "Error",
101                 wxOK | wxICON_ERROR);
102             return;
103         }
104
105         Item* item(new Item(m_gamedata, name.ToStdString(),
106             description.ToStdString()));
107
108         if(m_gamedata == nullptr){
109             std::cerr<<"you are fucking dumb\n";
110             std::abort();
111         }
112         m_gamedata->addItem(item);*/
113
114         // UpdateEqListCtrl();
115     }
```

5.3.3.2 OnAddSlot()

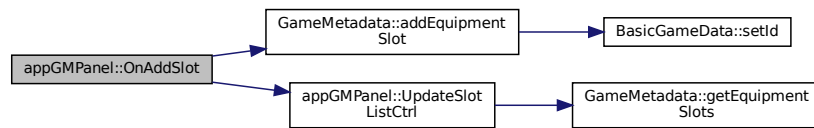
```
void appGMPanel::OnAddSlot (
    wxCommandEvent & event ) [private]
```

Definition at line 117 of file appGM.cpp.

```
117     {
118         wxString name = wxGetTextFromUser("Enter name:", "Add State");
119         wxString description = wxGetTextFromUser("Enter description:", "Add State");
120
121         if (name.IsEmpty()) {
122             wxMessageBox("Name or description cannot be empty!", "Error",
123                 wxOK | wxICON_ERROR);
124             return;
125         }
126
127         m_gamedata->addEquipmentSlot(
128             new EquipmentSlot(name.ToStdString(), description.ToStdString()));
129         UpdateSlotListCtrl();
130
131     }
```

References `GameMetadata::addEquipmentSlot()`, `m_gamedata`, and `UpdateSlotListCtrl()`.

Here is the call graph for this function:



5.3.3.3 OnAddStat()

```
void appGMPanel::OnAddStat (
    wxCommandEvent & event ) [private]
```

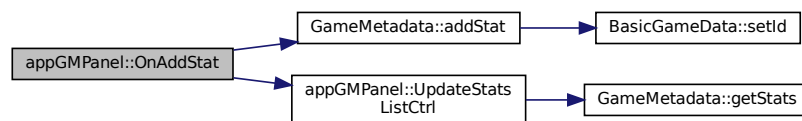
Definition at line 80 of file appGM.cpp.

```

80     {
81         wxString name = wxGetTextFromUser("Enter name:", "Add Stat");
82         wxString description = wxGetTextFromUser("Enter description:", "Add Stat");
83
84         if (name.IsEmpty()) {
85             wxMessageBox("Name or description cannot be empty!", "Error",
86                 wxOK | wxICON_ERROR);
87             return;
88         }
89
90         m_gamedata->addStat(new Stat(name.ToStdString(), description.ToStdString()));
91         UpdateStatsListCtrl();
92     }
93 }
```

References `GameMetadata::addStat()`, `m_gamedata`, and `UpdateStatsListCtrl()`.

Here is the call graph for this function:



5.3.3.4 OnAddState()

```
void appGMPanel::OnAddState (
    wxCommandEvent & event ) [private]
```

Definition at line 133 of file appGM.cpp.

```

133     {
134         wxString name = wxGetTextFromUser("Enter name:", "Add State");
```

```

135   wxString description = wxGetTextFromUser("Enter description:", "Add State");
136
137   if (name.IsEmpty()) {
138       wxMessageBox("Name or description cannot be empty!", "Error",
139                   wxOK | wxICON_ERROR);
140       return;
141   }
142
143   State *state{
144       new State(m_gamedata, name.ToStdString(), description.ToStdString());
145   }
146   UpdateStateListCtrl();
147 }

```

References `m_gamedata`, and `UpdateStateListCtrl()`.

Here is the call graph for this function:



5.3.3.5 UpdateEqListCtrl()

```
void appGMPanel::UpdateEqListCtrl ( ) [private]
```

Definition at line 159 of file `appGM.cpp`.

```

159 {
160     /*eqListCtrl->DeleteAllItems();
161
162     GameData::
163     for (size_t i = 0; i < equipment.size(); i++) {
164         Equipment *eq = equipment[i];
165         long index = eqListCtrl->InsertItem(i, eq->GetName());
166         eqListCtrl->SetItem(index, 1, eq->GetDescription());
167         eqListCtrl->SetItemData(index, reinterpret_cast<wxUIntPtr>(eq));
168     }*/
169 }

```

5.3.3.6 UpdateSlotListCtrl()

```
void appGMPanel::UpdateSlotListCtrl ( ) [private]
```

Definition at line 171 of file `appGM.cpp`.

```

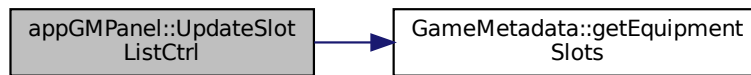
171 {
172     slotListCtrl->DeleteAllItems();
173     long i{};
174     for (const auto &it : m_gamedata->getEquipmentSlots()) {
175         long index = slotListCtrl->InsertItem(i, it->getName());
176         ++i;
177     }
178 }

```

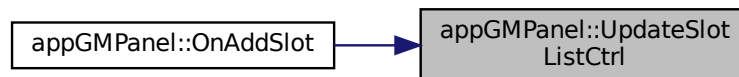
References `GameMetadata::getEquipmentSlots()`, `m_gamedata`, and `slotListCtrl`.

Referenced by `OnAddSlot()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.3.3.7 UpdateStateListCtrl()

```
void appGMPanel::UpdateStateListCtrl ( ) [private]
```

Definition at line 180 of file `appGM.cpp`.

```

180     {
181         stateListCtrl->DeleteAllItems();
182         long i{};
183         for (const auto &it : m_gamedata->getStates()) {
184             long index = stateListCtrl->InsertItem(i, it->getName());
185             ++i;
186         }
187     }
  
```

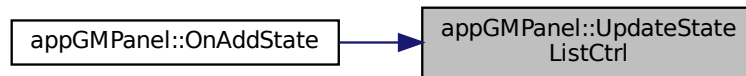
References `GameData::getStates()`, `m_gamedata`, and `stateListCtrl`.

Referenced by `OnAddState()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.3.3.8 UpdateStatsListCtrl()

```
void appGMPanel::UpdateStatsListCtrl ( ) [private]
```

Definition at line 149 of file `appGM.cpp`.

```

149     {
150         statsListCtrl->DeleteAllItems();
151         long i{};
152         for (const auto &it : m_gamedata->getStats()) {
153             long index = statsListCtrl->InsertItem(i, it->getName());
154             statsListCtrl->SetItem(index, 1, it->getDescription());
155             ++i;
156         }
157     }
  
```

References `GameMetadata::getStats()`, `m_gamedata`, and `statsListCtrl`.

Referenced by `OnAddStat()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.3.3.9 wxDECLARE_EVENT_TABLE()

```
appGMPanel::wxDECLARE_EVENT_TABLE ( ) [private]
```

5.3.4 Member Data Documentation

5.3.4.1 addElementPanel

```
wxPanel* appGMPanel::addElementPanel [private]
```

Definition at line 20 of file appGM.hpp.

5.3.4.2 addElementSizer

```
wxBoxSizer* appGMPanel::addElementSizer [private]
```

Definition at line 26 of file appGM.hpp.

5.3.4.3 addEquipmentButton

```
wxButton* appGMPanel::addEquipmentButton [private]
```

Definition at line 38 of file appGM.hpp.

5.3.4.4 addSlotButton

```
wxButton* appGMPanel::addSlotButton [private]
```

Definition at line 39 of file appGM.hpp.

5.3.4.5 addStatButton

```
wxButton* appGMPanel::addStatButton [private]
```

Definition at line 37 of file appGM.hpp.

5.3.4.6 addStateButton

```
wxButton* appGMPanel::addStateButton [private]
```

Definition at line 40 of file appGM.hpp.

5.3.4.7 eqListCtrl

```
wxListCtrl* appGMPanel::eqListCtrl [private]
```

Definition at line 33 of file appGM.hpp.

5.3.4.8 eqPanel

```
wxPanel* appGMPanel::eqPanel [private]
```

Definition at line 22 of file appGM.hpp.

5.3.4.9 eqSizer

```
wxBoxSizer* appGMPanel::eqSizer [private]
```

Definition at line 28 of file appGM.hpp.

5.3.4.10 gmNotebook

```
wxNotebook* appGMPanel::gmNotebook [private]
```

Definition at line 18 of file appGM.hpp.

5.3.4.11 m_gamedata

```
GameData* appGMPanel::m_gamedata {nullptr} [private]
```

Definition at line 42 of file appGM.hpp.

Referenced by OnAddSlot(), OnAddStat(), OnAddState(), UpdateSlotListCtrl(), UpdateStateListCtrl(), and UpdateStatsListCtrl().

5.3.4.12 slotListCtrl

```
wxListCtrl* appGMPanel::slotListCtrl [private]
```

Definition at line 34 of file appGM.hpp.

Referenced by UpdateSlotListCtrl().

5.3.4.13 slotPanel

```
wxPanel* appGMPanel::slotPanel [private]
```

Definition at line 23 of file appGM.hpp.

5.3.4.14 slotSizer

```
wxBoxSizer* appGMPanel::slotSizer [private]
```

Definition at line 29 of file appGM.hpp.

5.3.4.15 stateListCtrl

```
wxListCtrl* appGMPanel::stateListCtrl [private]
```

Definition at line 35 of file appGM.hpp.

Referenced by UpdateStateListCtrl().

5.3.4.16 statePanel

```
wxPanel* appGMPanel::statePanel [private]
```

Definition at line 24 of file appGM.hpp.

5.3.4.17 stateSizer

```
wxBoxSizer* appGMPanel::stateSizer [private]
```

Definition at line 30 of file appGM.hpp.

5.3.4.18 statsListCtrl

```
wxListCtrl* appGMPanel::statsListCtrl [private]
```

Definition at line 32 of file appGM.hpp.

Referenced by UpdateStatsListCtrl().

5.3.4.19 statsPanel

```
wxPanel* appGMPanel::statsPanel [private]
```

Definition at line 21 of file appGM.hpp.

5.3.4.20 statsSizer

```
wxBoxSizer* appGMPanel::statsSizer [private]
```

Definition at line 27 of file appGM.hpp.

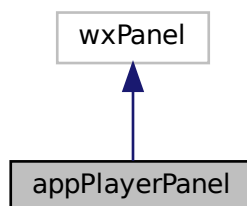
The documentation for this class was generated from the following files:

- [appGM.hpp](#)
- [appGM.cpp](#)

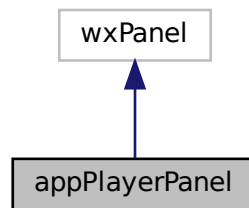
5.4 appPlayerPanel Class Reference

```
#include <appPlayer.hpp>
```

Inheritance diagram for appPlayerPanel:



Collaboration diagram for appPlayerPanel:



Public Member Functions

- [appPlayerPanel](#) (wxNotebook *parent, const wxString &playerName)

Private Member Functions

- void [OnAddStat](#) (wxCommandEvent &event)
- void [OnAddEquipment](#) (wxCommandEvent &event)
- void [OnAddState](#) (wxCommandEvent &event)
- void [OnEquipEquipment](#) (wxCommandEvent &event)
- void [OnUnequipEquipment](#) (wxCommandEvent &event)
- void [UpdateStatsListCtrl](#) ()
- void [UpdateEquippedEQListCtrl](#) ()
- void [UpdateEQListCtrl](#) ()
- void [UpdateStatesListCtrl](#) ()
- [wxDECLARE_EVENT_TABLE](#) ()

Private Attributes

- wxNotebook * [playerNotebook](#)
- wxPanel * [statsPanel](#)
- wxPanel * [equippedEQPanel](#)
- wxPanel * [eqPanel](#)
- wxPanel * [statesPanel](#)
- wxPanel * [addPanel](#)
- wxBoxSizer * [statsSizer](#)
- wxBoxSizer * [equippedEQSizer](#)
- wxBoxSizer * [eqSizer](#)
- wxBoxSizer * [statesSizer](#)
- wxBoxSizer * [addPanelSizer](#)
- wxBoxSizer * [mainSizer](#)
- wxListCtrl * [statsListCtrl](#)
- wxListCtrl * [equippedEQListCtrl](#)
- wxListCtrl * [eqListCtrl](#)
- wxListCtrl * [statesListCtrl](#)
- wxButton * [addStatButton](#)
- wxButton * [addEquipmentButton](#)
- wxButton * [addStateButton](#)
- wxButton * [equipEquipmentButton](#)
- wxButton * [unequipEquipmentButton](#)

5.4.1 Detailed Description

Definition at line 19 of file appPlayer.hpp.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 appPlayerPanel()

```
appPlayerPanel::appPlayerPanel (
    wxNotebook * parent,
    const wxString & playerName )
```

5.4.3 Member Function Documentation

5.4.3.1 OnAddEquipment()

```
void appPlayerPanel::OnAddEquipment (
    wxCommandEvent & event ) [private]
```

Definition at line 94 of file appPlayer.cpp.

```
95 {
96     // TODO: Implement adding an equipment
97 }
```

5.4.3.2 OnAddStat()

```
void appPlayerPanel::OnAddStat (
    wxCommandEvent & event ) [private]
```

Definition at line 89 of file appPlayer.cpp.

```
90 {
91     // TODO: Implement adding a stat
92 }
```

5.4.3.3 OnAddState()

```
void appPlayerPanel::OnAddState (
    wxCommandEvent & event ) [private]
```

Definition at line 99 of file appPlayer.cpp.

```
100 {
101     // TODO: Implement adding a state
102 }
```


5.4.3.4 OnEquipEquipment()

```
void appPlayerPanel::OnEquipEquipment (
    wxCommandEvent & event ) [private]
```

Definition at line 104 of file appPlayer.cpp.

```
105 {
106     // TODO: Implement equipping an equipment
107 }
```

5.4.3.5 OnUnequipEquipment()

```
void appPlayerPanel::OnUnequipEquipment (
    wxCommandEvent & event ) [private]
```

Definition at line 109 of file appPlayer.cpp.

```
110 {
111     // TODO: Implement unequipping an equipment
112 }
```

5.4.3.6 UpdateEQListCtrl()

```
void appPlayerPanel::UpdateEQListCtrl ( ) [private]
```

Definition at line 124 of file appPlayer.cpp.

```
125 {
126     // TODO: Update the equipment list control with data from eqCollection
127 }
```

5.4.3.7 UpdateEquippedEQListCtrl()

```
void appPlayerPanel::UpdateEquippedEQListCtrl ( ) [private]
```

Definition at line 119 of file appPlayer.cpp.

```
120 {
121     // TODO: Update the equipped equipment list control with data from eqCollection
122 }
```

5.4.3.8 UpdateStatesListCtrl()

```
void appPlayerPanel::UpdateStatesListCtrl ( ) [private]
```

Definition at line 129 of file appPlayer.cpp.

```
130 {
131     // TODO: Update the states list control with data from stateCollection
132 }
```

5.4.3.9 UpdateStatsListCtrl()

```
void appPlayerPanel::UpdateStatsListCtrl ( ) [private]
```

Definition at line 114 of file appPlayer.cpp.

```
115 {  
116     // TODO: Update the stats list control with data from statsCollection  
117 }
```

5.4.3.10 wxDECLARE_EVENT_TABLE()

```
appPlayerPanel::wxDECLARE_EVENT_TABLE ( ) [private]
```

5.4.4 Member Data Documentation

5.4.4.1 addEquipmentButton

```
wxButton* appPlayerPanel::addEquipmentButton [private]
```

Definition at line 46 of file appPlayer.hpp.

5.4.4.2 addPanel

```
wxPanel* appPlayerPanel::addPanel [private]
```

Definition at line 31 of file appPlayer.hpp.

5.4.4.3 addPanelSizer

```
wxBoxSizer* appPlayerPanel::addPanelSizer [private]
```

Definition at line 37 of file appPlayer.hpp.

5.4.4.4 addStatButton

```
wxButton* appPlayerPanel::addStatButton [private]
```

Definition at line 45 of file appPlayer.hpp.

5.4.4.5 addStateButton

```
wxButton* appPlayerPanel::addStateButton [private]
```

Definition at line 47 of file appPlayer.hpp.

5.4.4.6 eqListCtrl

```
wxListCtrl* appPlayerPanel::eqListCtrl [private]
```

Definition at line 42 of file appPlayer.hpp.

5.4.4.7 eqPanel

```
wxPanel* appPlayerPanel::eqPanel [private]
```

Definition at line 29 of file appPlayer.hpp.

5.4.4.8 eqSizer

```
wxBoxSizer* appPlayerPanel::eqSizer [private]
```

Definition at line 35 of file appPlayer.hpp.

5.4.4.9 equipEquipmentButton

```
wxButton* appPlayerPanel::equipEquipmentButton [private]
```

Definition at line 48 of file appPlayer.hpp.

5.4.4.10 equippedEQListCtrl

```
wxListCtrl* appPlayerPanel::equippedEQListCtrl [private]
```

Definition at line 41 of file appPlayer.hpp.

5.4.4.11 equippedEQPanel

```
wxPanel* appPlayerPanel::equippedEQPanel [private]
```

Definition at line 28 of file appPlayer.hpp.

5.4.4.12 equippedEQSizer

```
wxBoxSizer* appPlayerPanel::equippedEQSizer [private]
```

Definition at line 34 of file appPlayer.hpp.

5.4.4.13 mainSizer

```
wxBoxSizer* appPlayerPanel::mainSizer [private]
```

Definition at line 38 of file appPlayer.hpp.

5.4.4.14 playerNotebook

```
wxNotebook* appPlayerPanel::playerNotebook [private]
```

Definition at line 25 of file appPlayer.hpp.

5.4.4.15 statesListCtrl

```
wxListCtrl* appPlayerPanel::statesListCtrl [private]
```

Definition at line 43 of file appPlayer.hpp.

5.4.4.16 statesPanel

```
wxPanel* appPlayerPanel::statesPanel [private]
```

Definition at line 30 of file appPlayer.hpp.

5.4.4.17 statesSizer

```
wxBoxSizer* appPlayerPanel::statesSizer [private]
```

Definition at line 36 of file appPlayer.hpp.

5.4.4.18 statsListCtrl

```
wxListCtrl* appPlayerPanel::statsListCtrl [private]
```

Definition at line 40 of file appPlayer.hpp.

5.4.4.19 statsPanel

```
wxPanel* appPlayerPanel::statsPanel [private]
```

Definition at line 27 of file appPlayer.hpp.

5.4.4.20 statsSizer

```
wxBoxSizer* appPlayerPanel::statsSizer [private]
```

Definition at line 33 of file appPlayer.hpp.

5.4.4.21 unequipEquipmentButton

```
wxButton* appPlayerPanel::unequipEquipmentButton [private]
```

Definition at line 49 of file appPlayer.hpp.

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

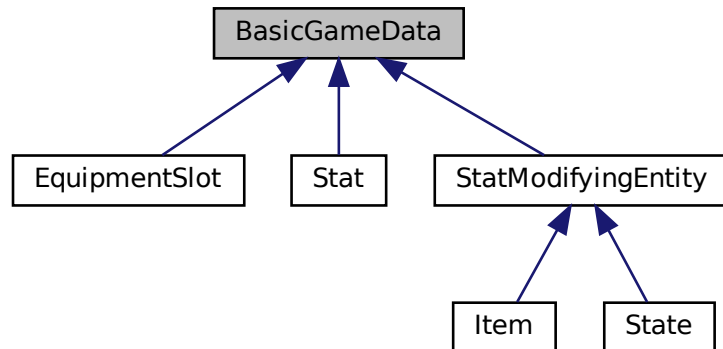
- [appPlayer.hpp](#)
- [appPlayer.cpp](#)

5.5 BasicGameData Class Reference

Basic information about game.

```
#include <basicGamedata.hpp>
```

Inheritance diagram for BasicGameData:



Public Types

- using `id_t` = long long
Type used for ids.

Public Member Functions

- `BasicGameData` (std::string name, std::string description="")
Constructor creating object with given name and optional description.
- void `setName` (std::string name)
Sets name.
- void `setDescription` (std::string description)
Sets description.
- std::string `getName` () const
name getter.
- std::string `getDescription` () const
Description getter.
- `id_t` `getId` () const
id getter.
- bool `isId` (`id_t` id) const
Check if id is equal.

Static Public Attributes

- static constexpr [id_t](#) [INVALID_ID](#) {std::numeric_limits<[id_t](#)>::min()}
Value indicating that id is invalid.

Protected Member Functions

- void [setId](#) ([id_t](#) id)
id setter.
- void [validateIntegrity](#) () const
Check integrity of data.

Private Attributes

- [id_t](#) [m_id](#) {[INVALID_ID](#)}
id.
- std::string [m_name](#)
name.
- std::string [m_description](#)
description.

Friends

- class [GameMetadata](#)
- class [GameData](#)

5.5.1 Detailed Description

Basic information about game.

Definition at line 27 of file [basicGamedata.hpp](#).

5.5.2 Member Typedef Documentation

5.5.2.1 [id_t](#)

```
using BasicGameData::id\_t = long long
```

Type used for ids.

Definition at line 33 of file [basicGamedata.hpp](#).

5.5.3 Constructor & Destructor Documentation

5.5.3.1 [BasicGameData\(\)](#)

```
BasicGameData::BasicGameData (  
    std::string name,  
    std::string description = "" )
```

Constructor creating object with given name and optional description.

Parameters

<i>name</i>	Name.
<i>description</i>	optional description.

Definition at line 14 of file basicGamedata.cpp.

```
15      : m_name(name), m_description(description) {}
```

5.5.4 Member Function Documentation

5.5.4.1 getDescription()

```
std::string BasicGameData::getDescription ( ) const
```

Description getter.

Returns

Description.

Definition at line 38 of file basicGamedata.cpp.

```
38 { return m_description; }
```

References `m_description`.

5.5.4.2 getId()

```
BasicGameData::id_t BasicGameData::getId ( ) const
```

id getter.

Returns

id.

Exceptions

<i>exceptionIllegalId</i>	When tred to get id of instance that has BasicGameData::INVALID_ID .
-------------------------------------------	--------------------------------------------------------------------------------------

Definition at line 40 of file basicGamedata.cpp.

```
40      {  
41  if (m_id == INVALID_ID)  
44      throw exceptionIllegalId();  
45  return m_id;  
46  };
```


References INVALID_ID, and m_id.

5.5.4.3 getName()

```
std::string BasicGameData::getName ( ) const
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References m_name.

5.5.4.4 isId()

```
bool BasicGameData::isId (
    id_t id ) const
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

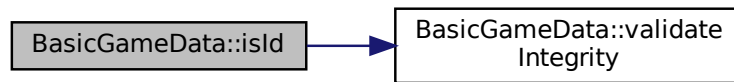
Definition at line 48 of file basicGamedata.cpp.

```
48                                     {
49     validateIntegrity();
50     bool res{m_id == id};
51     return res;
52 }
```

References m_id, and validateIntegrity().

Referenced by GameData::getItem().

Here is the call graph for this function:



Here is the caller graph for this function:



5.5.4.5 setDescription()

```
void BasicGameData::setDescription (
    std::string description )
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file basicGamedata.cpp.

```
32                                     {
33     m_description = description;
34 }
```

References `m_description`.

5.5.4.6 setId()

```
void BasicGameData::setId (
    id_t id ) [protected]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

Definition at line 17 of file basicGamedata.cpp.

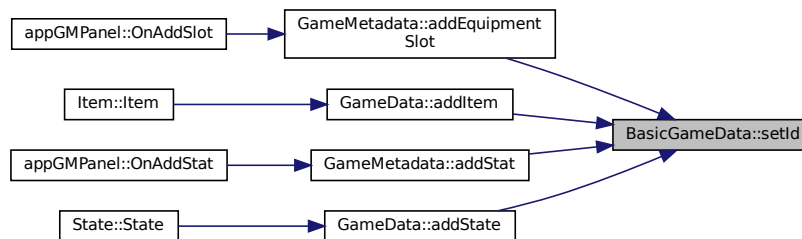
```

17         {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References INVALID_ID, and m_id.

Referenced by GameMetadata::addEquipmentSlot(), GameData::addItem(), GameMetadata::addStat(), and GameData::addState().

Here is the caller graph for this function:



5.5.4.7 setName()

```

void BasicGameData::setName (
    std::string name )
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```

30 { m_name = name; }
```

References m_name.

5.5.4.8 validateIntegrity()

```

void BasicGameData::validateIntegrity ( ) const [protected]
```

Check integrity of data.

Checks if id is INVALID_ID.

Exceptions

<code>exceptionIllegalId</code>	id is illegal.
-------------------------------------------------	----------------

Definition at line 22 of file basicGamedata.cpp.

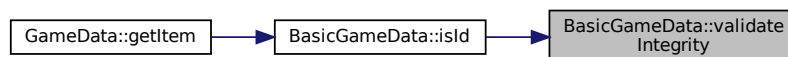
```

22                                     {
24     if (m_id == INVALID_ID) {
26         throw exceptionIllegalId();
27     }
28 }
```

References INVALID_ID, and m_id.

Referenced by isId().

Here is the caller graph for this function:



5.5.5 Friends And Related Function Documentation

5.5.5.1 GameData

```
friend class GameData [friend]
```

Definition at line 29 of file basicGamedata.hpp.

5.5.5.2 GameMetadata

```
friend class GameMetadata [friend]
```

Definition at line 28 of file basicGamedata.hpp.

5.5.6 Member Data Documentation

5.5.6.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by getId(), setId(), and validateIntegrity().

5.5.6.2 m_description

```
std::string BasicGameData::m_description [private]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by getDescription(), and setDescription().

5.5.6.3 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by getId(), isId(), setId(), and validateIntegrity().

5.5.6.4 m_name

```
std::string BasicGameData::m_name [private]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by getName(), and setName().

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

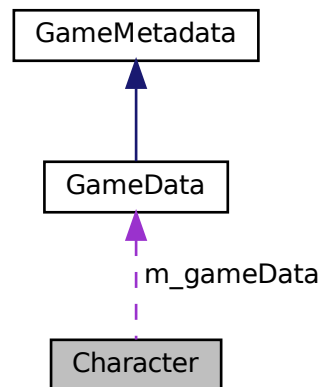
- [basicGamedata.hpp](#)
- [basicGamedata.cpp](#)

5.6 Character Class Reference

Represents character.

```
#include <character.hpp>
```

Collaboration diagram for Character:



Public Member Functions

- `Character (GameData *gameData)`
constructor
- `void setBaseStatValue (Stat::id_t statId, Stat::value_t val)`
Sets base stat value.
- `Stat::value_t getBaseStatValue (Stat::id_t id) const`
Base statistics value getter.
- `statValueContributors_t getStatValueStatesContributors (Stat::id_t id) const`
Gets collection of stat modifiers caused by states.
- `statValueContributors_t getStatValueEquipmentContributors (Stat::id_t id) const`
Gets collection of stat modifiers caused by equipment.
- `statValueContributors_t getStatValueContributors (Stat::id_t id) const`
Gets collection of stat modifiers.
- `Stat::value_t getStatValue (Stat::id_t id) const`
Gets final stat value.
- `const GameData *const getGameData () const`
Gets game data used by Character.
- `void addItem (const Item *const item, itemQuantity_t quantity=1)`
Adds item to character inventory.
- `const inventory_t & getInventory () const`
Inventory getter.
- `const equipment_t & getEquipment () const`
equipment getter

- bool `isEquipmentSlotUsed` (`EquipmentSlot::id_t` eqSlotId) const
Check if slot is used.
- const `Item` *const `getEquipedItem` (`EquipmentSlot::id_t` slotId) const
Get item equiped in given slot.
- void `equipItem` (const `Item` *const item, `EquipmentSlot::id_t` eqSlot)
Equip item into eqSlot.
- void `addState` (const `State` *state)
Adds State to the Character.
- const `states_t` & `getStates` () const
States getter.

Private Types

- using `statValues_t` = std::map< `Stat::id_t`, `Stat::value_t` >
Type used for representation of base character stats.
- using `statValueContributors_t` = std::vector< const `StatModifyingEntity` * >
Stat value contrubitors collection.
- using `itemQuantity_t` = long long
Quantity of Items.
- using `inventory_t` = std::map< const `Item` *const, `itemQuantity_t` >
Used for inventory as <Item, qunatity possed by Character>
- using `equipment_t` = std::map< const `EquipmentSlot` *const, const `Item` *const >
Used to store equiped items of character.
- using `states_t` = std::set< const `State` * >
Used to store sates affecting Character.

Private Member Functions

- void `validateDataIntegrity` (const `StatModifyingEntity` &entity) const
Validate data integrity.
- const `Item` *const `getEquipedItem` (const `EquipmentSlot` *const eqSlot) const
Get item equiped in given slot.

Private Attributes

- const `GameData` *const `m_gameData`
Game data used by character.
- `statValues_t` `m_baseStatValues`
Base values of stats.
- `inventory_t` `m_inventory`
Inventorty.
- `equipment_t` `m_equipment`
Equiped Items.
- `states_t` `m_states`
States affecting Character.

5.6.1 Detailed Description

Represents character.

Base character stats are stats that character has without any modifiers applied. If Base stat is not set it will be assumed to be 0;

Definition at line 62 of file character.hpp.

5.6.2 Member Typedef Documentation

5.6.2.1 equipment_t

```
using Character::equipment_t = std::map<const EquipmentSlot *const, const Item *const> [private]
```

Used to store equipped items of character.

Definition at line 75 of file character.hpp.

5.6.2.2 inventory_t

```
using Character::inventory_t = std::map<const Item *const, itemQuantity_t> [private]
```

Used for inventory as <Item, quantity possessed by Character>

Definition at line 72 of file character.hpp.

5.6.2.3 itemQuantity_t

```
using Character::itemQuantity_t = long long [private]
```

Quantity of Items.

Definition at line 69 of file character.hpp.

5.6.2.4 states_t

```
using Character::states_t = std::set<const State *> [private]
```

Used to store states affecting Character.

Definition at line 78 of file character.hpp.

5.6.2.5 statValueContributors_t

```
using Character::statValueContributors_t = std::vector<const StatModifyingEntity *> [private]
```

Stat value contributors collection.

Definition at line 66 of file character.hpp.

5.6.2.6 statValues_t

```
using Character::statValues_t = std::map<Stat::id_t, Stat::value_t> [private]
```

Type used for representation of base character stats.

Definition at line 64 of file character.hpp.

5.6.3 Constructor & Destructor Documentation

5.6.3.1 Character()

```
Character::Character (
    GameData * gameData )
```

constructor

Parameters

<i>gameData</i>	Used by character. <code>Character *character{new Character(gd)}; character->setBaseStatValue(1, 4);</code>
-----------------	-------------------------------------------------------------------------------------------------------------------

Definition at line 30 of file character.cpp.

```
30 : m_gameData(gameData) {}
```

5.6.4 Member Function Documentation

5.6.4.1 addItem()

```
void Character::addItem (
    const Item *const item,
    itemQuantity_t quantity = 1 )
```

Adds item to character inventory.

Parameters

<i>item</i>	Item to add.
<i>quantity</i>	Qunatity of item to add.

```
// extract item from gamedata accessed from character.
const Item *itemToAdd{character->getGameData()->getItem(1)};
character->addItem(itemToAdd);
character->addItem(itemToAdd);
character->addItem(character->getGameData()->getItem(2));
```

Todo make it remove item from inventory if quantity would go below or equal 0.

Definition at line 98 of file character.cpp.

```
98
99 validateDataIntegrity(*item);
100 if (quantity == 0)
101     return;
102
103 auto temp{m_inventory.insert({item, quantity})};
104
105 const bool didAdd{temp.second};
106 // If not added, change item quantity.
107 if (!didAdd) {
108     // iterator to item
109     auto itemIterator{temp.first};
110     itemQuantity_t &itemQuantity{itemIterator->second};
111     itemQuantity += quantity;
112 }
113 }
114 }
115 }
```

References `m_inventory`, and `validateDataIntegrity()`.

Here is the call graph for this function:



5.6.4.2 addState()

```
void Character::addState (
    const State * state )
```

Adds `State` to the `Character`.

Parameters

<i>state</i>	State to add.
--------------	---------------

```
// Getting State which has id 2.
const State *state{character->getGameData()->getState(2)};
// Adding State to character.
```

```
character->addState(state);
```

Definition at line 160 of file character.cpp.

```
160 {
161     validateDataIntegrity(*state);
162     m_states.insert(state);
163 }
```

References `m_states`, and `validateDataIntegrity()`.

Here is the call graph for this function:



5.6.4.3 equipItem()

```
void Character::equipItem (
    const Item *const item,
    EquipmentSlot::id_t eqSlot )
```

Equip item into eqSlot.

Parameters

<i>item</i>	Item to equip.
<i>eqSlot</i>	Where to equip it to.

Note

It does not check whenever [Character](#) has item in inventory.

Exceptions

excpetionEquipmentSlotIllegalUsage	When attempting to equip item into slot that it can not be equipped to.
exceptionEquipmentSlotOccupied	When attempting to equip item into already occupied slot;

Warning

If moving [Item](#) from inventory to equipment(as if it was take out of inventory and put on by [Character](#)). Removal of [Item](#) should be performed after this method call. If it's removed before and exception is thrown [Item](#) would have been removed from inventory and not be equipped.

```
// Item to equip.
```

```

const Item *item{character->getGameData()->getItem(2)};
// Slot where to equip that item into.
EquipmentSlot::id_t slot{*item->getEquipableSlots().begin()};
// Optional checking if Character has that item in inventory should be here.
character->equipItem(item, slot);
// Optional removal of item from Inventory.

```

Definition at line 144 of file character.cpp.

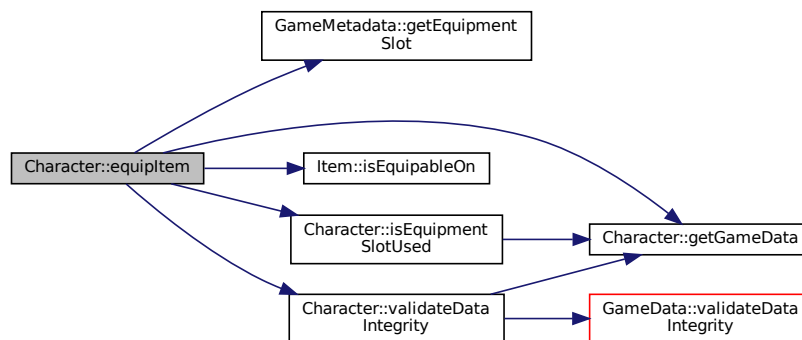
```

144 {
145     validateDataIntegrity(*item);
146     // check if item can even be equipped into that slot.
147     if (!item->isEquipableOn(eqSlot))
148         throw exceptionEquipmentSlotIllegalUsage();
149
150     // check if desired slot is occupied.
151     if (isEquipmentSlotUsed(eqSlot))
152         throw exceptionEquipmentSlotOccupied();
153
154     // get slot pointer as it's used for internal equipment information.
155     const EquipmentSlot *const eqSlotPtr{getGameData()->getEquipmentSlot(eqSlot)};
156
157     m_equipment.insert({eqSlotPtr, item});
158 }

```

References GameMetadata::getEquipmentSlot(), getGameData(), Item::isEquipableOn(), isEquipmentSlotUsed(), m_equipment, and validateDataIntegrity().

Here is the call graph for this function:



5.6.4.4 getBaseStatValue()

```

Stat::value_t Character::getBaseStatValue (
    Stat::id_t id ) const

```

Base statistics value getter.

Parameters

<i>id</i>	of stat to get base value of.
-----------	-------------------------------

Returns

Base value of stat with given id.

If [Character](#) does not have base [Stat](#) value of given id 0 is returned as per class invariant.

Definition at line 38 of file character.cpp.

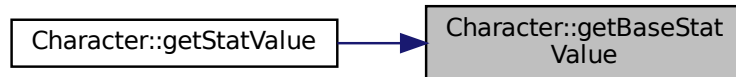
```

38                                     {
39     auto res{m_baseStatValues.find(id)};
40     if (res != m_baseStatValues.end())
41         return res->second;
42     return 0;
43 }
```

References `m_baseStatValues`.

Referenced by `getStatValue()`.

Here is the caller graph for this function:

**5.6.4.5 getEquippedItem() [1/2]**

```

const Item *const Character::getEquippedItem (
    const EquipmentSlot *const eqSlot ) const [private]
```

Get item equipped in given slot.

Parameters

<i>eqSlot</i>	EquipmentSlot to get equipped item of.
---------------	--------------------------------------------------------

Exceptions

exceptionEquipmentSlotUnused	When trying to get equipped item of unused slot.
----------------------------------------------	--------------------------------------------------

Returns

Equiped item.

Definition at line 132 of file character.cpp.

```

132                                     {
```

```

133   equipment_t::const_iterator r{getEquipment().find(eqSlot)};
134   if (r == getEquipment().end())
135       throw exceptionEquipmentSlotUnused();
136   return r->second;
137 }

```

References `getEquipment()`.

Referenced by `getEquipedItem()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.6.4.6 `getEquipedItem()` [2/2]

```

const Item *const Character::getEquipedItem (
    EquipmentSlot::id_t slotId ) const

```

Get item equipped in given slot.

Parameters

<i>slotId</i>	<code>EquipmentSlot</code> to get equipped item of.
---------------	-----------------------------------------------------

Returns

Equiped item.

Definition at line 139 of file `character.cpp`.

```

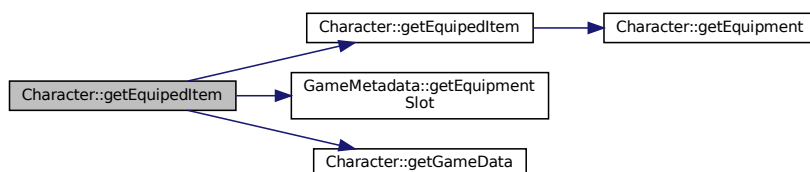
139   {
140   const EquipmentSlot *const eqSlot{getGameData()->getEquipmentSlot(slotId)};
141   return getEquipedItem(eqSlot);

```

```
142 }
```

References `getEquipedItem()`, `GameMetadata::getEquipmentSlot()`, and `getGameData()`.

Here is the call graph for this function:



5.6.4.7 `getEquipment()`

```
const Character::equipment_t & Character::getEquipment ( ) const
```

equipment getter

Returns

`m_equipment`

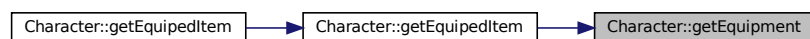
Definition at line 121 of file `character.cpp`.

```
121                                     {
122     return m_equipment;
123 }
```

References `m_equipment`.

Referenced by `getEquipedItem()`.

Here is the caller graph for this function:



5.6.4.8 getGameData()

```
const GameData *const Character::getGameData ( ) const
```

Gets game data used by [Character](#).

Returns

::m_gameData.

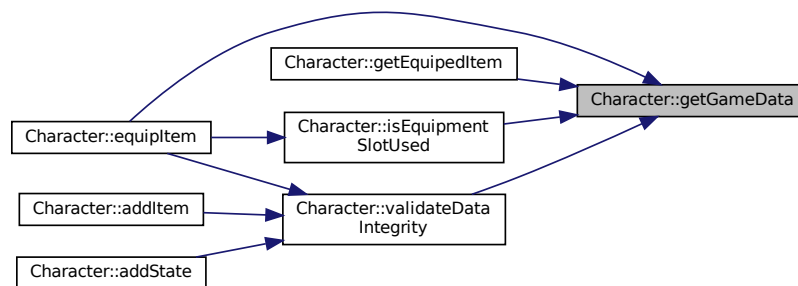
Definition at line 96 of file character.cpp.

```
96 { return m_gameData; }
```

References m_gameData.

Referenced by [equipItem\(\)](#), [getEquipedItem\(\)](#), [isEquipmentSlotUsed\(\)](#), and [validateDataIntegrity\(\)](#).

Here is the caller graph for this function:



5.6.4.9 getInventory()

```
const Character::inventory\_t & Character::getInventory ( ) const
```

Inventory getter.

Returns

m_inventory

Definition at line 117 of file character.cpp.

```
117                                     {
118     return m_inventory;
119 }
```

References m_inventory.

5.6.4.10 getStates()

```
const Character::states_t & Character::getStates ( ) const
```

States getter.

Returns

m_states

Definition at line 165 of file character.cpp.

```
165 { return m_states; }
```

References m_states.

5.6.4.11 getStatValue()

```
Stat::value_t Character::getStatValue (
    Stat::id_t id ) const
```

Gets final stat value.

Parameters

<i>id</i>	Id of stat to get value of.
-----------	-----------------------------

Returns

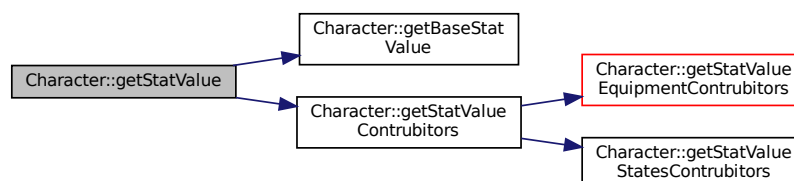
Value of stat.

Definition at line 85 of file character.cpp.

```
85 {
86     statValueContributors_t contributors{getStatValueContributors(id)};
87
88     Stat::value_t result{getBaseStatValue(id)};
89
90     for (const StatModifyingEntity *contr : contributors) {
91         result += contr->getModifierValue(id);
92     }
93     return result;
94 }
```

References getBaseStatValue(), and getStatValueContributors().

Here is the call graph for this function:



5.6.4.12 getStatValueContributors()

```
Character::statValueContributors_t Character::getStatValueContributors (
    Stat::id_t id ) const
```

Gets collection of stat modifiers.

Parameters

<i>id</i>	Id of stat.
-----------	-------------

Returns

Contributors to [Stat](#) value.

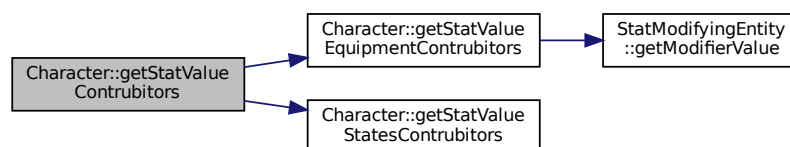
Definition at line 73 of file character.cpp.

```
73                                     {
74     statValueContributors_t result{};
75     auto temp{getStatValueEquipmentContributors(id)};
76     // equipment contribs.
77     result.insert(result.end(), temp.begin(), temp.end());
78     temp = getStatValueStatesContributors(id);
79     // states contribs.
80     result.insert(result.end(), temp.begin(), temp.end());
81
82     return result;
83 }
```

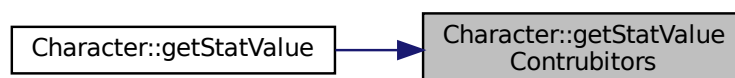
References [getStatValueEquipmentContributors\(\)](#), and [getStatValueStatesContributors\(\)](#).

Referenced by [getStatValue\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



5.6.4.13 getStatValueEquipmentContrubitors()

```
Character::statValueContrubitors_t Character::getStatValueEquipmentContrubitors (
    Stat::id_t id ) const
```

Gets collection of stat modifiers caused by equipment.

Parameters

<i>id</i>	Id of stat.
-----------	-------------

Returns

Contrubitors to Stat value.

Definition at line 58 of file character.cpp.

```
58                                     {
59     statValueContrubitors_t result{};
60     // For each equipment piece
61     for (auto it : m_equipment) {
62         const StatModifyingEntity *modifier{it.second};
63         // Get modifier of stat caused by that piece.
64         Stat::value_t modv{modifier->getModifierValue(id)};
65         // if it modifies stat add it to result.
66         if (modv != 0)
67             result.push_back(modifier);
68     }
69     return result;
70 }
```

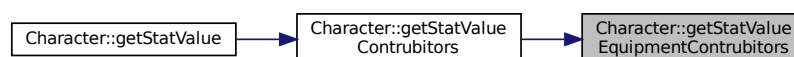
References StatModifyingEntity::getModifierValue(), and m_equipment.

Referenced by getStatValueContrubitors().

Here is the call graph for this function:



Here is the caller graph for this function:



5.6.4.14 getStatValueStatesContrubitors()

```
Character::statValueContrubitors_t Character::getStatValueStatesContrubitors (
    Stat::id_t id ) const
```

Gets collection of stat modifiers caused by states.

Parameters

<i>id</i>	Id of stat.
-----------	-------------

Returns

Contrubitors to [Stat](#) value.

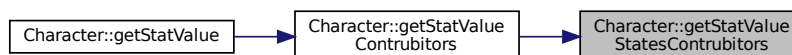
Definition at line 46 of file character.cpp.

```
46                                     {
47     statValueContrubitors_t result{};
48     for (auto it : m_states) {
49         Stat::value_t modv{it->getModifierValue(id)};
50         // if it modifies stat add it to result.
51         if (modv != 0)
52             result.push_back(it);
53     }
54     return result;
55 }
```

References `m_states`.

Referenced by `getStatValueContrubitors()`.

Here is the caller graph for this function:



5.6.4.15 isEquipmentSlotUsed()

```
bool Character::isEquipmentSlotUsed (
    EquipmentSlot::id_t eqSlotId ) const
```

Check if slot is used.

Parameters

<i>eq</i> ↔ <i>SlotId</i>	id of eq slot to check.
------------------------------	-------------------------

Returns

True if slot is used. False if it is not used.

Definition at line 125 of file character.cpp.

```

125                                     {
126   bool result{m_equipment.find(getGameData()->getEquipmentSlot(eqSlotId)) !=
127                           m_equipment.end()};
128   return result;
129 }
```

References `getGameData()`, and `m_equipment`.

Referenced by `equiplItem()`.

Here is the call graph for this function:



Here is the caller graph for this function:

**5.6.4.16 setBaseStatValue()**

```

void Character::setBaseStatValue (
    Stat::id_t statId,
    Stat::value_t val )
```

Sets base stat value.

Parameters

<i>statId</i>	Id to set value of.
<i>val</i>	Value to set.

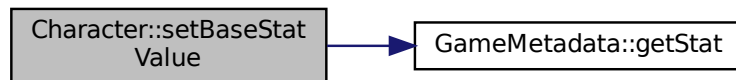
Definition at line 32 of file character.cpp.

```

32                                     {
33     m_gameData->getStat(statId);
34
35     m_baseStatValues[statId] = val;
36 }
```

References GameMetadata::getStat(), m_baseStatValues, and m_gameData.

Here is the call graph for this function:



5.6.4.17 validateDataIntegrity()

```

void Character::validateDataIntegrity (
    const StatModifyingEntity & entity ) const [private]
```

Validate data integrity.

Parameters

<i>entity</i>	Entity to validate agints.
---------------	----------------------------

Exceptions

<i>excpetionGameDataMismatch</i>	When Character and item do not use same GameData .
----------------------------------	------------------------------------------------------------------------------------

Definition at line 26 of file character.cpp.

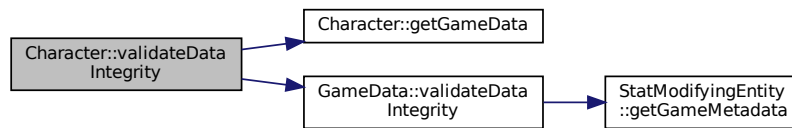
```

26                                     {
27     getGameData()->validateDataIntegrity(entity);
28 }
```

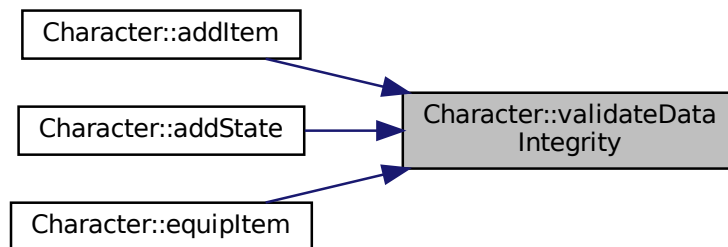
References getGameData(), and GameData::validateDataIntegrity().

Referenced by addItem(), addState(), and equipItem().

Here is the call graph for this function:



Here is the caller graph for this function:



5.6.5 Member Data Documentation

5.6.5.1 m_baseStatValues

```
statValues_t Character::m_baseStatValues [private]
```

Base values of stats.

Definition at line 83 of file `character.hpp`.

Referenced by `getBaseStatValue()`, and `setBaseStatValue()`.

5.6.5.2 m_equipment

```
equipment_t Character::m_equipment [private]
```

Equiped Items.

Definition at line 87 of file `character.hpp`.

Referenced by `equipItem()`, `getEquipment()`, `getStatValueEquipmentContrubitors()`, and `isEquipmentSlotUsed()`.

5.6.5.3 m_gameData

```
const GameData\* const Character::m_gameData [private]
```

Game data used by character.

Definition at line 81 of file character.hpp.

Referenced by [getGameData\(\)](#), and [setBaseStatValue\(\)](#).

5.6.5.4 m_inventory

```
inventory\_t Character::m_inventory [private]
```

Inventory.

Definition at line 85 of file character.hpp.

Referenced by [addItem\(\)](#), and [getInventory\(\)](#).

5.6.5.5 m_states

```
states\_t Character::m_states [private]
```

States affecting [Character](#).

Definition at line 89 of file character.hpp.

Referenced by [addState\(\)](#), [getStates\(\)](#), and [getStatValueStatesContributors\(\)](#).

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

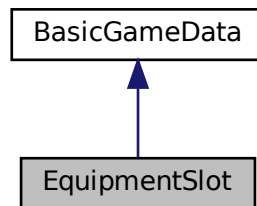
- [character.hpp](#)
- [character.cpp](#)

5.7 EquipmentSlot Class Reference

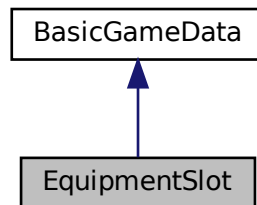
Equipment slot representation.

```
#include <equipmentSlot.hpp>
```

Inheritance diagram for EquipmentSlot:



Collaboration diagram for EquipmentSlot:



Public Types

- using `id_t` = long long
Type used for ids.

Public Member Functions

- `EquipmentSlot` (std::string name, std::string description="")
Constructor.
- void `setName` (std::string name)
Sets name.
- void `setDescription` (std::string description)
Sets description.

- `std::string getName () const`
name getter.
- `std::string getDescription () const`
Description getter.
- `id_t getId () const`
id getter.
- `bool isId (id_t id) const`
Check if id is equal.

Static Public Attributes

- static constexpr `id_t INVALID_ID` {std::numeric_limits<id_t>::min()}
Value indicating that id is invalid.

Protected Member Functions

- void `setId (id_t id)`
id setter.
- void `validateIntegrity () const`
Check integrity of data.

Private Attributes

- `id_t m_id` {INVALID_ID}
id.
- `std::string m_name`
name.
- `std::string m_description`
description.

5.7.1 Detailed Description

Equipment slot representation.

Definition at line 14 of file `equipmentSlot.hpp`.

5.7.2 Member Typedef Documentation

5.7.2.1 id_t

```
using BasicGameData::id_t = long long [inherited]
```

Type used for ids.

Definition at line 33 of file `basicGamedata.hpp`.

5.7.3 Constructor & Destructor Documentation

5.7.3.1 EquipmentSlot()

```
EquipmentSlot::EquipmentSlot (
    std::string name,
    std::string description = "" )
```

Constructor.

Parameters

<i>name</i>	Name.
<i>description</i>	Optional description.

Definition at line 8 of file equipmentSlot.cpp.

```
9      : BasicGameData(name, description) {}
```

5.7.4 Member Function Documentation

5.7.4.1 getDescription()

```
std::string BasicGameData::getDescription ( ) const [inherited]
```

Description getter.

Returns

Description.

Definition at line 38 of file basicGamedata.cpp.

```
38 { return m_description; };
```

References BasicGameData::m_description.

5.7.4.2 getId()

```
BasicGameData::id_t BasicGameData::getId ( ) const [inherited]
```

id getter.

Returns

id.

Exceptions

<i>exceptionIllegalId</i>	When tred to get id of instance that has BasicGameData::INVALID_ID .
-------------------------------------------	--------------------------------------------------------------------------------------

Definition at line 40 of file basicGamedata.cpp.

```

40                                     {
41     if (m_id == INVALID_ID)
44         throw exceptionIllegalId();
45     return m_id;
46 };

```

References [BasicGameData::INVALID_ID](#), and [BasicGameData::m_id](#).

5.7.4.3 getName()

```
std::string BasicGameData::getName ( ) const [inherited]
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References [BasicGameData::m_name](#).

5.7.4.4 isId()

```
bool BasicGameData::isId (
    id_t id ) const [inherited]
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

Definition at line 48 of file basicGamedata.cpp.

```

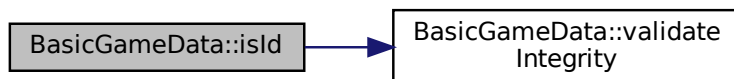
48                                     {
49     validateIntegrity();
50     bool res{m_id == id};
51     return res;
52 }

```

References BasicGameData::m_id, and BasicGameData::validateIntegrity().

Referenced by GameData::getItem().

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.5 setDescription()

```
void BasicGameData::setDescription (
    std::string description ) [inherited]
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file basicGamedata.cpp.

```
32                                     {
33     m_description = description;
34 }
```

References BasicGameData::m_description.

5.7.4.6 setId()

```
void BasicGameData::setId (
    id_t id ) [protected], [inherited]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

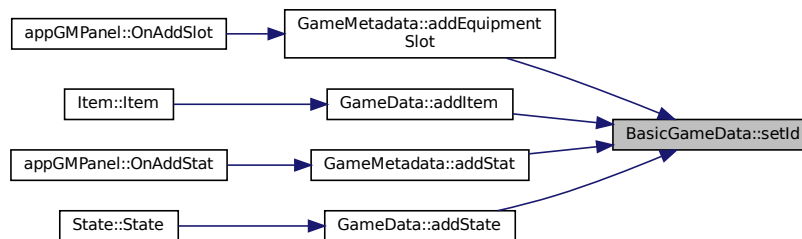
Definition at line 17 of file basicGamedata.cpp.

```
17 {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by GameMetadata::addEquipmentSlot(), GameData::addItem(), GameMetadata::addStat(), and GameData::addState().

Here is the caller graph for this function:



5.7.4.7 setName()

```
void BasicGameData::setName (
    std::string name ) [inherited]
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```
30 { m_name = name; }
```

References BasicGameData::m_name.

5.7.4.8 validateIntegrity()

```
void BasicGameData::validateIntegrity ( ) const [protected], [inherited]
```

Check integrity of data.

Checks if id is INVALID_ID.

Exceptions

<i>exceptionIllegalId</i>	id is illegal.
-------------------------------------------	----------------

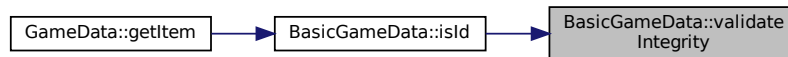
Definition at line 22 of file basicGamedata.cpp.

```
22 {
24     if (m_id == INVALID_ID) {
26         throw exceptionIllegalId();
27     }
28 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by BasicGameData::isId().

Here is the caller graph for this function:



5.7.5 Member Data Documentation

5.7.5.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr], [inherited]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.7.5.2 m_description

```
std::string BasicGameData::m_description [private], [inherited]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by BasicGameData::getDescription(), and BasicGameData::setDescription().

5.7.5.3 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private], [inherited]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::isId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.7.5.4 m_name

```
std::string BasicGameData::m_name [private], [inherited]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by BasicGameData::getName(), and BasicGameData::setName().

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

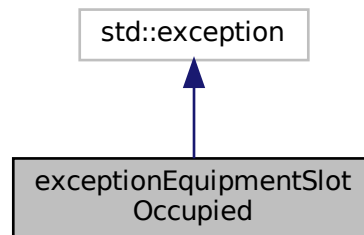
- [equipmentSlot.hpp](#)
- [equipmentSlot.cpp](#)

5.8 exceptionEquipmentSlotOccupied Class Reference

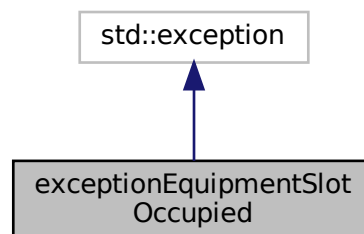
Tried to perform operation on occupied already slot.

```
#include <character.hpp>
```

Inheritance diagram for exceptionEquipmentSlotOccupied:



Collaboration diagram for exceptionEquipmentSlotOccupied:



Private Member Functions

- `std::string what ()`
What.

5.8.1 Detailed Description

Tried to perform operation on occupied already slot.

Definition at line 25 of file character.hpp.

5.8.2 Member Function Documentation

5.8.2.1 what()

```
std::string exceptionEquipmentSlotOccupied::what ( ) [private]
```

What.

Returns

Message.

Definition at line 14 of file character.cpp.

```
14 {  
15     return "tried to perform operation on occupied equipment slot";  
16 }
```

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

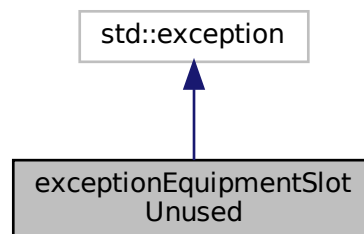
- [character.hpp](#)
- [character.cpp](#)

5.9 exceptionEquipmentSlotUnused Class Reference

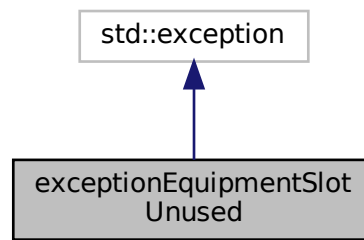
Tried to extract data from unused slot.

```
#include <character.hpp>
```

Inheritance diagram for exceptionEquipmentSlotUnused:



Collaboration diagram for exceptionEquipmentSlotUnused:



Private Member Functions

- `std::string what ()`
What.

5.9.1 Detailed Description

Tried to extract data from unused slot.

Definition at line 36 of file `character.hpp`.

5.9.2 Member Function Documentation

5.9.2.1 what()

```
std::string exceptionEquipmentSlotUnused::what ( ) [private]
```

What.

Returns

Message.

Definition at line 18 of file `character.cpp`.

```
18 {
19     return "Tried to extract data from unused slot";
20 }
```

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

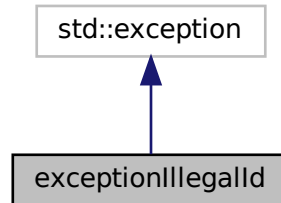
- [character.hpp](#)
- [character.cpp](#)

5.10 exceptionIllegalId Class Reference

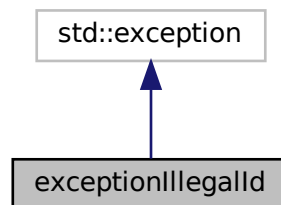
Illegal id exception.

```
#include <basicGamedata.hpp>
```

Inheritance diagram for exceptionIllegalId:



Collaboration diagram for exceptionIllegalId:



Private Member Functions

- `std::string what ()`
What.

5.10.1 Detailed Description

Illegal id exception.

See also

[BasicGameData::INVALID_ID](#)

Definition at line 17 of file basicGamedata.hpp.

5.10.2 Member Function Documentation

5.10.2.1 what()

```
std::string exceptionIllegalId::what ( ) [private]
```

What.

Returns

Info string.

Definition at line 10 of file basicGamedata.cpp.

```
10     {  
11     return "Tried to perform operations on instance with invalid id";  
12 }
```

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

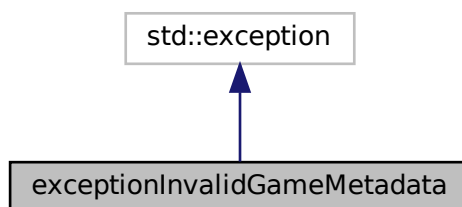
- [basicGamedata.hpp](#)
- [basicGamedata.cpp](#)

5.11 exceptionInvalidGameMetadata Class Reference

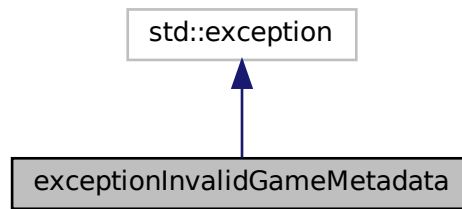
Invalid game data.

```
#include <statModifyingEntity.hpp>
```

Inheritance diagram for exceptionInvalidGameMetadata:



Collaboration diagram for exceptionInvalidGameMetadata:



Private Member Functions

- `std::string what ()`
What.

5.11.1 Detailed Description

Invalid game data.

Definition at line 18 of file `statModifyingEntity.hpp`.

5.11.2 Member Function Documentation

5.11.2.1 what()

```
std::string exceptionInvalidGameMetadata::what ( ) [private]
```

What.

Returns

Messaage.

Definition at line 10 of file `statModifyingEntity.cpp`.

```
10 {  
11     return "Invalid Game metadata";  
12 }
```

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

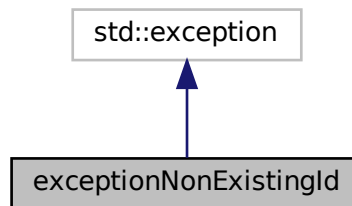
- [statModifyingEntity.hpp](#)
- [statModifyingEntity.cpp](#)

5.12 exceptionNonExistingId Class Reference

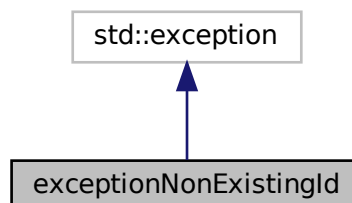
Exception.

```
#include <gameMetadata.hpp>
```

Inheritance diagram for exceptionNonExistingId:



Collaboration diagram for exceptionNonExistingId:



Public Member Functions

- virtual std::string [what](#) ()
What.

5.12.1 Detailed Description

Exception.

Definition at line 19 of file gameMetadata.hpp.

5.12.2 Member Function Documentation

5.12.2.1 what()

```
std::string exceptionNonExistingId::what ( ) [virtual]
```

What.

Returns

message.

Definition at line 11 of file gameMetadata.cpp.

```
11 { return "Not found with given ID"; }
```

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

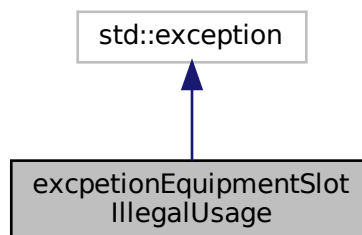
- [gameMetadata.hpp](#)
- [gameMetadata.cpp](#)

5.13 excpetionEquipmentSlotIllegalUsage Class Reference

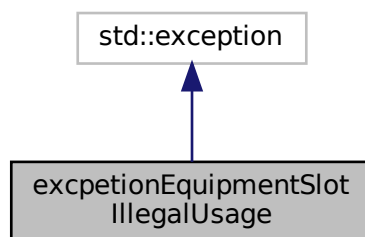
Tried to put stuff where it is not suppsloed to go.

```
#include <character.hpp>
```

Inheritance diagram for excpetionEquipmentSlotIllegalUsage:



Collaboration diagram for excpetionEquipmentSlotIllegalUsage:



Private Member Functions

- `std::string what ()`
What.

5.13.1 Detailed Description

Tried to put stuff where it is not supposed to go.

Definition at line 47 of file `character.hpp`.

5.13.2 Member Function Documentation

5.13.2.1 what()

```
std::string excpetionEquipmentSlotIllegalUsage::what ( ) [private]
```

What.

Returns

Message.

Definition at line 22 of file `character.cpp`.

```
22 {
23     return "Illegal equipment slot usage";
24 }
```

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

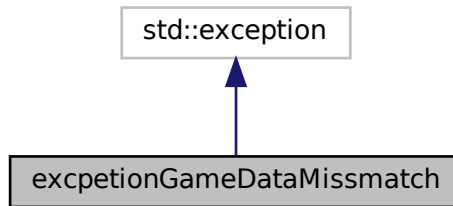
- [character.hpp](#)
- [character.cpp](#)

5.14 excpetionGameDataMismatch Class Reference

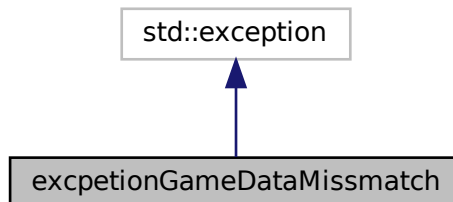
Thrown when attempted to do operation that requires 2 objects to use common [GameData](#) but different were used.

```
#include <gameData.hpp>
```

Inheritance diagram for excpetionGameDataMismatch:



Collaboration diagram for excpetionGameDataMismatch:



Private Member Functions

- `std::string what ()`
what

5.14.1 Detailed Description

Thrown when attempted to do operation that requires 2 objects to use common [GameData](#) but different were used.

Definition at line 19 of file `gameData.hpp`.

5.14.2 Member Function Documentation

5.14.2.1 what()

```
std::string excpetionGameDataMismatch::what ( ) [private]
```

what

Returns

message

Definition at line 11 of file gameData.cpp.

```
11 { return "Game data mismatch"; }
```

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

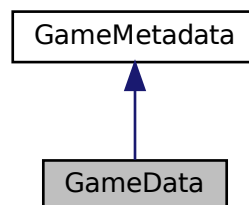
- [gameData.hpp](#)
- [gameData.cpp](#)

5.15 GameData Class Reference

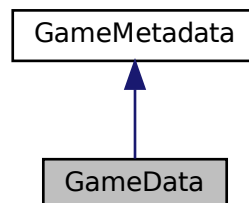
On top of what [GameMetadata](#) does. Holds items catalogue.

```
#include <gameData.hpp>
```

Inheritance diagram for GameData:



Collaboration diagram for GameData:



Public Types

- using `itemcollection_t` = `std::set< Item * >`
Collection used to hold items.
- using `stateCollcetion_t` = `std::set< State * >`
Colection used to hold states.
- using `statsCollection_t` = `std::vector< Stat * >`
Type used for stats collection.
- using `equipmentSlotsCollection_t` = `std::vector< EquipmentSlot * >`
Type used for stroing equipment slots collection.

Public Member Functions

- `~GameData ()`
Desctructor.
- `const itemcollection_t & getItems () const`
Getter.
- `const Item *const getItem (Item::id_t id) const`
Get Item.
- `void validateDataIntegrity (const StatModifyingEntity &entity) const`
Checks if Item uses this GameData insnace as it's metadata.
- `void addState (State *state)`
Adds State to collection and sets it's id.
- `const stateCollcetion_t & getStates () const`
States getter.
- `const State *const getState (State::id_t id) const`
State getter.
- `void addStat (Stat *stat)`
Add given stat.
- `void addEquipmentSlot (EquipmentSlot *eqSlot)`
Add given EquipmentSlot.
- `Stat * getStat (Stat::id_t id) const`
Getter for Stat based on id.
- `const statsCollection_t & getStats () const`
Stats getter.
- `EquipmentSlot * getEquipmentSlot (Stat::id_t id) const`
Getter for EquipmentSlot based on id.
- `const equipmentSlotsCollection_t & getEquipmentSlots () const`
Equipmentslots getter.

Private Member Functions

- `void addItem (Item *item)`
Adds item to collection and sets it's id.

Private Attributes

- friend [Item](#)
- [itemcollection_t m_items](#)
Collection of items that exist in game collection.
- [Item::id_t m_nextItemId](#) {1}
Id that will be given to next item added.
- [stateCollcetion_t m_states](#)
Collection of states that exist in game.
- [State::id_t m_nextStatId](#) {1}
Id that will be given to next state added.
- [statsCollection_t m_stats](#)
Collection of [Stat](#) added.
- [equipmentSlotsCollection_t m_equipmentSlots](#)
Collection of [EquipmentSlot](#) added.
- [BasicGameData::id_t m_nextStatId](#) {1}
Id that will be set to the next [Stat](#) added.
- [BasicGameData::id_t m_nextEquipmentSlotId](#) {1}
Id that will be set to the next [EquipmentSlot](#) added.

5.15.1 Detailed Description

On top of what [GameMetadata](#) does. Holds items catalogue.

See also

[GameMetadata](#)

Definition at line 31 of file `gameData.hpp`.

5.15.2 Member Typedef Documentation

5.15.2.1 `equipmentSlotsCollection_t`

```
using GameMetadata::equipmentSlotsCollection\_t = std::vector<EquipmentSlot *> [inherited]
```

Type used for stroing equipment slots collection.

Definition at line 41 of file `gameMetadata.hpp`.

5.15.2.2 itemcollection_t

```
using GameData::itemcollection_t = std::set<Item *>
```

Collection used to hold items.

Definition at line 36 of file gameData.hpp.

5.15.2.3 stateCollction_t

```
using GameData::stateCollction_t = std::set<State *>
```

Collection used to hold states.

Definition at line 39 of file gameData.hpp.

5.15.2.4 statsCollection_t

```
using GameMetadata::statsCollection_t = std::vector<Stat *> [inherited]
```

Type used for stats collection.

Definition at line 39 of file gameMetadata.hpp.

5.15.3 Constructor & Destructor Documentation

5.15.3.1 ~GameData()

```
GameData::~GameData ( )
```

Destructor.

Definition at line 13 of file gameData.cpp.

```
13     {
14     for (auto it : m_items) {
15         delete it;
16     }
17
18     for (auto it : m_states) {
19         delete it;
20     }
21 }
```

References `m_items`, and `m_states`.

5.15.4 Member Function Documentation

5.15.4.1 addEquipmentSlot()

```
void GameMetadata::addEquipmentSlot (
    EquipmentSlot * eqSlot ) [inherited]
```

Add given `EquipmentSlot`.

Parameters

<i>eqSlot</i>	EquipmentSlot to add.. <pre>gameMetadata->addEquipmentSlot(new EquipmentSlot("Leg", "Leg is leg")); gameMetadata->addEquipmentSlot(new EquipmentSlot("Head")); gameMetadata->addEquipmentSlot(new EquipmentSlot("Hand", "For gloves or smth"));</pre>
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

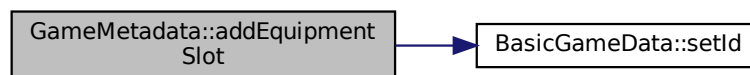
Definition at line 26 of file gameMetadata.cpp.

```
26                                     {
27     m_equipmentSlots.push_back(eqSlot);
28     eqSlot->setId(m_nextEquipmentSlotId);
29     ++m_nextEquipmentSlotId;
30 }
```

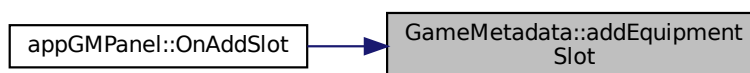
References GameMetadata::m_equipmentSlots, GameMetadata::m_nextEquipmentSlotId, and BasicGameData::setId().

Referenced by appGMPanel::OnAddSlot().

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.4.2 addItem()

```
void GameData::addItem (
    Item * item ) [private]
```

Adds item to collection and sets it's id.

Parameters

<i>item</i>	Item to add.
-------------	---------------------

```

Item *item{new Item(gameData, "Stick")};
item->addModifier(1, 3);
item->setEquipableOn(3);
Item *item2{new Item(gameData, "Sunglasses", "They protect from sun")};
item2->addModifier(2, 100);
item2->setEquipableOn(2);

```

Note

It does not validate item.

Todo LOW change excpetion to it's own exception class

Exceptions

<code>std::__throw_runtime_error</code>	"Item was not added to GameData" When item was not added to the game some reason.
-----------------------------------------	-----------------------------------------------------------------------------------

Definition at line 23 of file gameData.cpp.

```

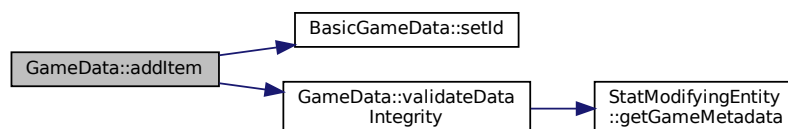
23 {
24     validateDataIntegrity(*item);
25     item->setId(m_nextItemId);
26     ++m_nextItemId;
27     auto res{m_items.insert(item)};
28     if (!res.second)
32         std::__throw_runtime_error("Item was not added to GameData");
33 }

```

References `m_items`, `m_nextItemId`, `BasicGameData::setId()`, and `validateDataIntegrity()`.

Referenced by `Item::Item()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.4.3 addStat()

```
void GameMetadata::addStat (
    Stat * stat ) [inherited]
```

Add given stat.

Parameters

stat	Stat to add. gameMetadata->addStat(new Stat("strength", "just strength")); gameMetadata->addStat(new Stat("chadness", "Only chads have this")); gameMetadata->addStat(new Stat("speed"));
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Definition at line 20 of file gameMetadata.cpp.

```
20 {
21     m_stats.push_back(stat);
22     stat->setId(m_nextStatId);
23     ++m_nextStatId;
24 }
```

References GameMetadata::m_nextStatId, GameMetadata::m_stats, and BasicGameData::setId().

Referenced by appGMPanel::OnAddStat().

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.4.4 addState()

```
void GameData::addState (
    State * state )
```

Adds **State** to collection and sets it's id.

Parameters

<i>state</i>	<p>State to add.</p> <pre>State *inz{new State(gameData, "inz")}; inz->addModifier(1, 2); inz->addModifier(2, 1);</pre>
--------------	---------------------------------------------------------------------------------------------------------------------------------------

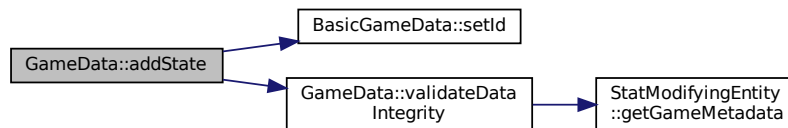
Definition at line 53 of file gameData.cpp.

```
53 {
54     validateDataIntegrity(*state);
55     m_states.insert(state);
56     state->setId(m_nextStateId);
57     ++m_nextStateId;
58 }
```

References `m_nextStateId`, `m_states`, `BasicGameData::setId()`, and `validateDataIntegrity()`.

Referenced by `State::State()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.4.5 getEquipmentSlot()

```
EquipmentSlot * GameMetadata::getEquipmentSlot (
    Stat::id_t id ) const [inherited]
```

Getter for `EquipmentSlot` based on `id`.

Parameters

<i>id</i>	Id of <code>EquipmentSlot</code> to fetch
-----------	-------------------------------------------

Returns

[EquipmentSlot](#) with given id.

Exceptions

exceptionNonExistingId	When tried to fetch instance with id that doesn't exist.
----------------------------------------	----------------------------------------------------------

Definition at line 44 of file gameMetadata.cpp.

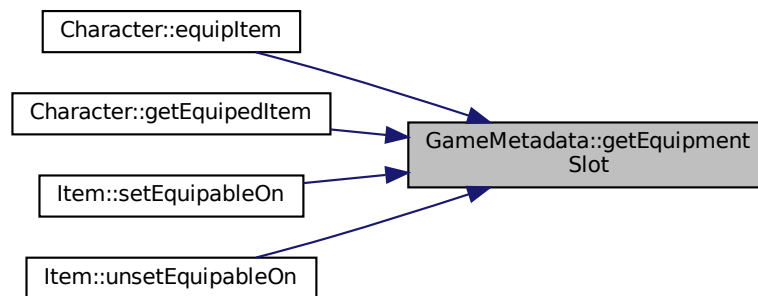
```

44                                     {
45     for (EquipmentSlot *eq : m_equipmentSlots) {
46         if (eq->m_id == id)
47             return eq;
48     }
49     throw exceptionNonExistingId();
50 }
```

References `GameMetadata::m_equipmentSlots`.

Referenced by `Character::equipItem()`, `Character::getEquippedItem()`, `Item::setEquipableOn()`, and `Item::unsetEquipableOn()`.

Here is the caller graph for this function:



5.15.4.6 getEquipmentSlots()

```

const GameMetadata::equipmentSlotsCollection\_t & GameMetadata::getEquipmentSlots ( ) const
[inherited]
```

Equipmentslots getter.

Returns

Equipment slots

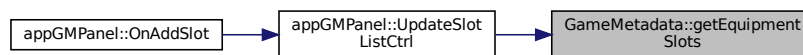
Definition at line 53 of file gameMetadata.cpp.

```
53     {
54     return m_equipmentSlots;
55 }
```

References GameMetadata::m_equipmentSlots.

Referenced by appGMPanel::UpdateSlotListCtrl().

Here is the caller graph for this function:

**5.15.4.7 getItem()**

```
const Item *const GameData::getItem (
    Item::id_t id ) const
```

Get [Item](#).

Parameters

<i>id</i>	Id of Item to fetch.
-----------	--------------------------------------

Returns

[Item](#) with given id.

Exceptions

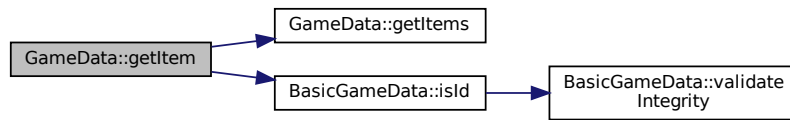
<i>exceptionNonExistingId</i>	Tried to get item that does not exist.
-----------------------------------------------	----------------------------------------

Definition at line 37 of file gameData.cpp.

```
37     {
38     auto lookup{std::find_if(getItems().begin(), getItems().end(),
39                             [=](const Item *const it) { return it->isId(id); })};
40
41     if (lookup == getItems().end())
42         throw exceptionNonExistingId();
43     return *lookup;
44 }
```

References [getItem\(\)](#), and [BasicGameData::isId\(\)](#).

Here is the call graph for this function:



5.15.4.8 getItems()

```
const GameData::itemcollection_t & GameData::getItems ( ) const
```

Getter.

Returns

`::m_items`.

Definition at line 35 of file `gameData.cpp`.

```
35 { return m_items; }
```

References `m_items`.

Referenced by `getItem()`.

Here is the caller graph for this function:



5.15.4.9 getStat()

```
Stat * GameMetadata::getStat (
    Stat::id_t id ) const [inherited]
```

Getter for `Stat` based on `id`.

Parameters

<i>id</i>	Id of Stat to fetch
-----------	-------------------------------------

Returns

[Stat](#) with given id.

Exceptions

exceptionNonExistingId	When tried to fetch instance with id that doesn't exist.
----------------------------------------	----------------------------------------------------------

Definition at line 32 of file gameMetadata.cpp.

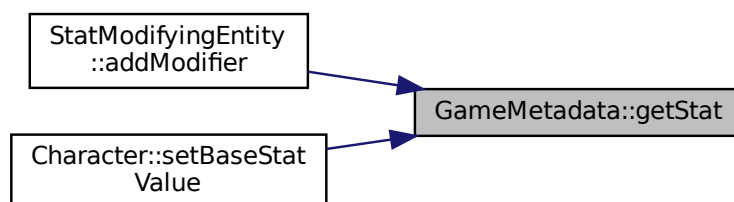
```

32                                     {
33     for (Stat *st : m_stats) {
34         if (st->m_id == id)
35             return st;
36     }
37     throw exceptionNonExistingId();
38 }
```

References `GameMetadata::m_stats`.

Referenced by `StatModifyingEntity::addModifier()`, and `Character::setBaseStatValue()`.

Here is the caller graph for this function:



5.15.4.10 getState()

```

const State *const GameData::getState (
    State::id_t id ) const
```

[State](#) getter.

Parameters

<i>id</i>	Id of State to get.
-----------	-------------------------------------

Returns

state with id asked.

Exceptions

<i>exceptionNonExistingId</i>	Tried to get State that does not exist.
-----------------------------------------------	---------------------------------------------------------

Definition at line 64 of file gameData.cpp.

```

64                                     {
65     auto lookup{
66         std::find_if(m_states.begin(), m_states.end(),
67             [=](const State *const state) { return state->isId(id); });
68     }
69     if (lookup == m_states.end()) {
70         throw exceptionNonExistingId();
71     }
72
73     return *lookup;
74 }
```

References `m_states`.

5.15.4.11 getStates()

```
const GameData::stateCollction_t & GameData::getStates ( ) const
```

States getter.

Returns

`m_states`

Definition at line 60 of file gameData.cpp.

```

60                                     {
61     return m_states;
62 }
```

References `m_states`.

Referenced by `appGMPanel::UpdateStateListCtrl()`.

Here is the caller graph for this function:



5.15.4.12 `getStats()`

```
const GameMetadata::statsCollection_t & GameMetadata::getStats ( ) const [inherited]
```

Stats getter.

Returns

`m_stats`.

Definition at line 40 of file `gameMetadata.cpp`.

```
40 {
41     return m_stats;
42 }
```

References `GameMetadata::m_stats`.

Referenced by `appGMPanel::UpdateStatsListCtrl()`.

Here is the caller graph for this function:



5.15.4.13 `validateDataIntegrity()`

```
void GameData::validateDataIntegrity (
    const StatModifyingEntity & entity ) const
```

Checks if [Item](#) uses this [GameData](#) insnace as it's metadata.

Parameters

<i>entity</i>	Entity to check.
---------------	------------------

Exceptions

<i>excpetionGameDataMismatch</i>	When entity does not use this as it's game data.
--------------------------------------------------	--------------------------------------------------

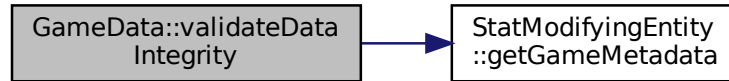
Definition at line 46 of file `gameData.cpp`.

```
46 {
47     if (this != entity.getGameMetadata())
48         throw excpetionGameDataMismatch();
49 }
```

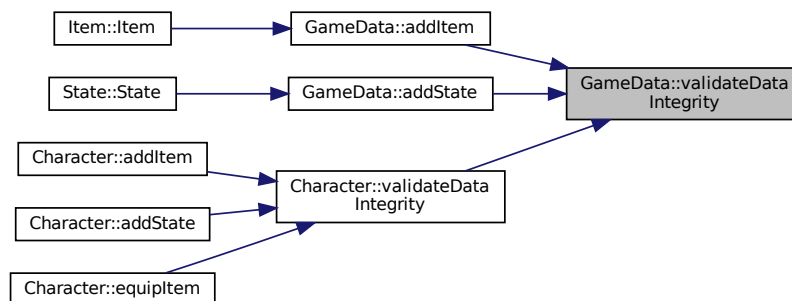
References `StatModifyingEntity::getGameMetadata()`.

Referenced by `addItem()`, `addState()`, and `Character::validateDataIntegrity()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.15.5 Member Data Documentation

5.15.5.1 Item

```
friend GameData::Item [private]
```

Definition at line 32 of file `gameData.hpp`.

5.15.5.2 m_equipmentSlots

```
equipmentSlotsCollection_t GameMetadata::m_equipmentSlots [private], [inherited]
```

Collection of [EquipmentSlot](#) added.

Definition at line 47 of file `gameMetadata.hpp`.

Referenced by `GameMetadata::addEquipmentSlot()`, `GameMetadata::getEquipmentSlot()`, `GameMetadata::getEquipmentSlots()`, and `GameMetadata::~GameMetadata()`.

5.15.5.3 m_items

```
itemcollection_t GameData::m_items [private]
```

Collection of items that exist in game collection.

Definition at line 43 of file gameData.hpp.

Referenced by addItem(), getItems(), and ~GameData().

5.15.5.4 m_nextEquipmentSlotId

```
BasicGameData::id_t GameMetadata::m_nextEquipmentSlotId {1} [private], [inherited]
```

Id that will be set to the next [EquipmentSlot](#) added.

Definition at line 52 of file gameMetadata.hpp.

Referenced by GameMetadata::addEquipmentSlot().

5.15.5.5 m_nextItemId

```
Item::id_t GameData::m_nextItemId {1} [private]
```

Id that will be given to next item added.

Definition at line 45 of file gameData.hpp.

Referenced by addItem().

5.15.5.6 m_nextStateId

```
State::id_t GameData::m_nextStateId {1} [private]
```

Id that will be given to next state added.

Definition at line 50 of file gameData.hpp.

Referenced by addState().

5.15.5.7 m_nextStatId

```
BasicGameData::id_t GameMetadata::m_nextStatId {1} [private], [inherited]
```

Id that will be set to the next [Stat](#) added.

Definition at line 50 of file gameMetadata.hpp.

Referenced by GameMetadata::addStat().

5.15.5.8 m_states

```
stateCollcetion_t GameData::m_states [private]
```

Collection of states that exist in game.

Definition at line 48 of file gameData.hpp.

Referenced by addState(), getState(), getStates(), and ~GameData().

5.15.5.9 m_stats

```
statsCollection_t GameMetadata::m_stats [private], [inherited]
```

Collection of [Stat](#) added.

Definition at line 45 of file gameMetadata.hpp.

Referenced by GameMetadata::addStat(), GameMetadata::getStat(), GameMetadata::getStats(), and GameMetadata::~~GameMetadata().

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

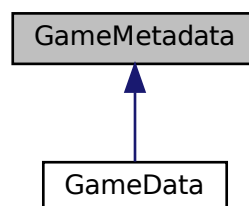
- [gameData.hpp](#)
- [gameData.cpp](#)

5.16 GameMetadata Class Reference

Holds game metadata. That is what statistics exist and what equipable slots exist.

```
#include <gameMetadata.hpp>
```

Inheritance diagram for GameMetadata:



Public Types

- using `statsCollection_t` = `std::vector< Stat * >`
Type used for stats collection.
- using `equipmentSlotsCollection_t` = `std::vector< EquipmentSlot * >`
Type used for stroing equipment slots collection.

Public Member Functions

- `~GameMetadata ()`
Destructor.
- void `addStat (Stat *stat)`
Add given stat.
- void `addEquipmentSlot (EquipmentSlot *eqSlot)`
Add given EquipmentSlot.
- `Stat * getStat (Stat::id_t id) const`
Getter for Stat based on id.
- const `statsCollection_t & getStats () const`
Stats getter.
- `EquipmentSlot * getEquipmentSlot (Stat::id_t id) const`
Getter for EquipmentSlot based on id.
- const `equipmentSlotsCollection_t & getEquipmentSlots () const`
Equipmentslots getter.

Private Attributes

- `statsCollection_t m_stats`
Collection of Stat added.
- `equipmentSlotsCollection_t m_equipmentSlots`
Collection of EquipmentSlot added.
- `BasicGameData::id_t m_nextStatId {1}`
Id that will be set to the next Stat added.
- `BasicGameData::id_t m_nextEquipmentSlotId {1}`
Id that will be set to the next EquipmentSlot added.

5.16.1 Detailed Description

Holds game metadata. That is what statistics exist and what equipable slots exist.

It will deallocate `Stat` and `EquipmentSlot` added to it so those should not be deallocated manually.

Definition at line 36 of file `gameMetadata.hpp`.

5.16.2 Member Typedef Documentation

5.16.2.1 equipmentSlotsCollection_t

```
using GameMetadata::equipmentSlotsCollection_t = std::vector<EquipmentSlot *>
```

Type used for stroing equipment slots collection.

Definition at line 41 of file gameMetadata.hpp.

5.16.2.2 statsCollection_t

```
using GameMetadata::statsCollection_t = std::vector<Stat *>
```

Type used for stats collection.

Definition at line 39 of file gameMetadata.hpp.

5.16.3 Constructor & Destructor Documentation

5.16.3.1 ~GameMetadata()

```
GameMetadata::~GameMetadata ( )
```

Destructor.

Deallocates all added equipmentslots and stats.

Definition at line 13 of file gameMetadata.cpp.

```
13 {
14     for (Stat *&it : m_stats)
15         delete it;
16     for (EquipmentSlot *&it : m_equipmentSlots)
17         delete it;
18 }
```

References `m_equipmentSlots`, and `m_stats`.

5.16.4 Member Function Documentation

5.16.4.1 addEquipmentSlot()

```
void GameMetadata::addEquipmentSlot (
    EquipmentSlot * eqSlot )
```

Add given `EquipmentSlot`.

Parameters

<i>eqSlot</i>	EquipmentSlot to add.. <pre>gameMetadata->addEquipmentSlot(new EquipmentSlot("Leg", "Leg is leg")); gameMetadata->addEquipmentSlot(new EquipmentSlot("Head")); gameMetadata->addEquipmentSlot(new EquipmentSlot("Hand", "For gloves or smth"));</pre>
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

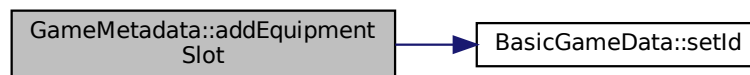
Definition at line 26 of file gameMetadata.cpp.

```
26                                     {
27     m_equipmentSlots.push_back(eqSlot);
28     eqSlot->setId(m_nextEquipmentSlotId);
29     ++m_nextEquipmentSlotId;
30 }
```

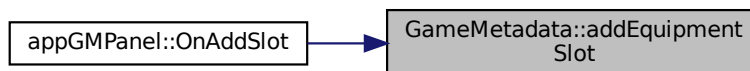
References `m_equipmentSlots`, `m_nextEquipmentSlotId`, and `BasicGameData::setId()`.

Referenced by `appGMPanel::OnAddSlot()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.16.4.2 addStat()

```
void GameMetadata::addStat (
    Stat * stat )
```

Add given stat.

Parameters

<i>stat</i>	Stat to add. <pre>gameMetadata->addStat(new Stat("strength", "just strength")); gameMetadata->addStat(new Stat("chadness", "Only chads have this")); gameMetadata->addStat(new Stat("speed"));</pre>
-------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Definition at line 20 of file gameMetadata.cpp.

```
20     {
21         m_stats.push_back(stat);
22         stat->setId(m_nextStatId);
23         ++m_nextStatId;
24     }
```

References `m_nextStatId`, `m_stats`, and `BasicGameData::setId()`.

Referenced by `appGMPanel::OnAddStat()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.16.4.3 getEquipmentSlot()

```
EquipmentSlot * GameMetadata::getEquipmentSlot (
    Stat::id_t id ) const
```

Getter for `EquipmentSlot` based on `id`.

Parameters

<i>id</i>	Id of <code>EquipmentSlot</code> to fetch
-----------	-------------------------------------------

Returns

[EquipmentSlot](#) with given id.

Exceptions

exceptionNonExistingId	When tried to fetch instance with id that doesn't exist.
----------------------------------------	----------------------------------------------------------

Definition at line 44 of file gameMetadata.cpp.

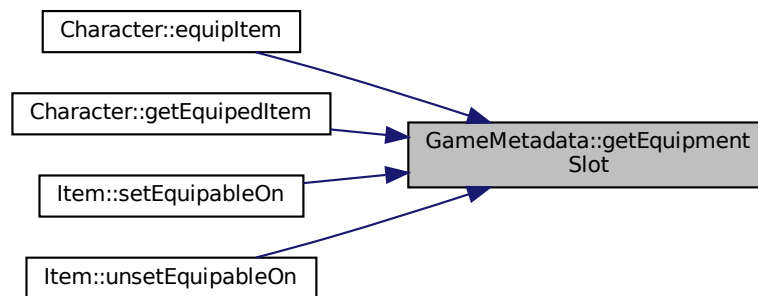
```

44                                     {
45     for (EquipmentSlot *eq : m_equipmentSlots) {
46         if (eq->m_id == id)
47             return eq;
48     }
49     throw exceptionNonExistingId();
50 }
```

References `m_equipmentSlots`.

Referenced by `Character::equipItem()`, `Character::getEquippedItem()`, `Item::setEquipableOn()`, and `Item::unsetEquipableOn()`.

Here is the caller graph for this function:



5.16.4.4 getEquipmentSlots()

```
const GameMetadata::equipmentSlotsCollection\_t & GameMetadata::getEquipmentSlots ( ) const
```

Equipmentslots getter.

Returns

Equipment slots

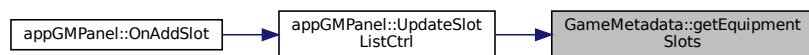
Definition at line 53 of file gameMetadata.cpp.

```
53     {
54     return m_equipmentSlots;
55 }
```

References `m_equipmentSlots`.

Referenced by `appGMPanel::UpdateSlotListCtrl()`.

Here is the caller graph for this function:

**5.16.4.5 getStat()**

```
Stat * GameMetadata::getStat (
    Stat::id_t id ) const
```

Getter for `Stat` based on id.

Parameters

<i>id</i>	Id of <code>Stat</code> to fetch
-----------	----------------------------------

Returns

`Stat` with given id.

Exceptions

<i>exceptionNonExistingId</i>	When tried to fetch instance with id that doesn't exist.
-------------------------------	----------------------------------------------------------

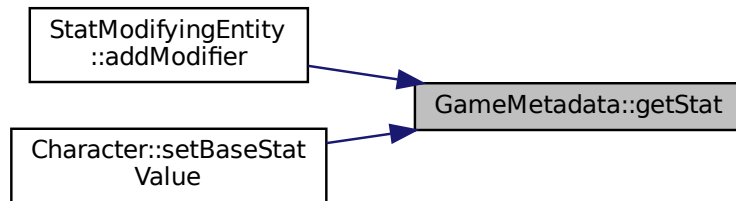
Definition at line 32 of file gameMetadata.cpp.

```
32     {
33     for (Stat *st : m_stats) {
34         if (st->m_id == id)
35             return st;
36     }
37     throw exceptionNonExistingId();
38 }
```

References `m_stats`.

Referenced by `StatModifyingEntity::addModifier()`, and `Character::setBaseStatValue()`.

Here is the caller graph for this function:



5.16.4.6 getStats()

```
const GameMetadata::statsCollection_t & GameMetadata::getStats ( ) const
```

Stats getter.

Returns

m_stats.

Definition at line 40 of file gameMetadata.cpp.

```

40                                     {
41     return m_stats;
42 }
```

References m_stats.

Referenced by appGMPanel::UpdateStatsListCtrl().

Here is the caller graph for this function:



5.16.5 Member Data Documentation

5.16.5.1 m_equipmentSlots

`equipmentSlotsCollection_t` GameMetadata::m_equipmentSlots [private]

Collection of [EquipmentSlot](#) added.

Definition at line 47 of file gameMetadata.hpp.

Referenced by `addEquipmentSlot()`, `getEquipmentSlot()`, `getEquipmentSlots()`, and `~GameMetadata()`.

5.16.5.2 m_nextEquipmentSlotId

`BasicGameData::id_t` GameMetadata::m_nextEquipmentSlotId {1} [private]

Id that will be set to the next [EquipmentSlot](#) added.

Definition at line 52 of file gameMetadata.hpp.

Referenced by `addEquipmentSlot()`.

5.16.5.3 m_nextStatId

`BasicGameData::id_t` GameMetadata::m_nextStatId {1} [private]

Id that will be set to the next [Stat](#) added.

Definition at line 50 of file gameMetadata.hpp.

Referenced by `addStat()`.

5.16.5.4 m_stats

`statsCollection_t` GameMetadata::m_stats [private]

Collection of [Stat](#) added.

Definition at line 45 of file gameMetadata.hpp.

Referenced by `addStat()`, `getStat()`, `getStats()`, and `~GameMetadata()`.

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

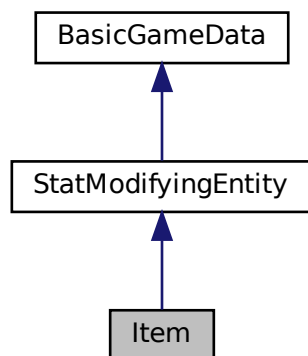
- [gameMetadata.hpp](#)
- [gameMetadata.cpp](#)

5.17 Item Class Reference

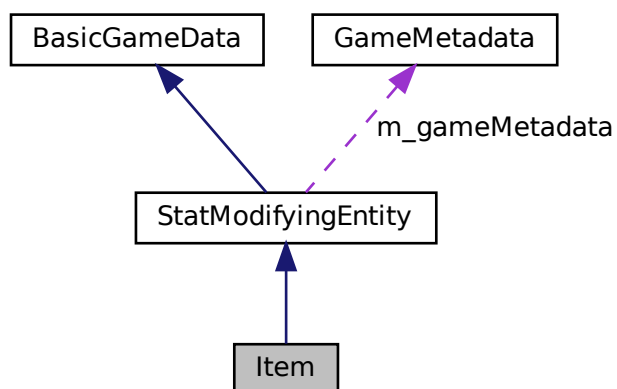
Represents an item in game.

```
#include <item.hpp>
```

Inheritance diagram for Item:



Collaboration diagram for Item:



Public Types

- using `equipableSlots_t` = `std::set< EquipmentSlot::id_t >`
Collection.

- using `modifier_t` = `std::pair< Stat::id_t, Stat::value_t >`
Represents modification of stats <stat modified, value of modification>
- using `modifiersCollection_t` = `std::vector< modifier_t >`
Collection type used to store modifiers.
- using `id_t` = `long long`
Type used for ids.

Public Member Functions

- `Item (GameData *gameData, std::string name, std::string description="")`
Constructor.
- void `setEquipableOn (EquipmentSlot::id_t equipmentSlotId)`
Make item equipable on given equipment slot.
- void `unsetEquipableOn (EquipmentSlot::id_t equipmentSlotId)`
Makes item no longer possible to equip onto given slot.
- bool `isEquipableOn (EquipmentSlot::id_t equipmentSlotId) const`
Checks whenever `Item` is equipable in given slot id.
- const `equipableSlots_t & getEquipableSlots () const`
Get equipable slots.
- void `addModifier (Stat::id_t, Stat::value_t by)`
Add modification of stats.
- `Stat::value_t getModifierValue (Stat::id_t id) const`
Modifier value getter.
- const `GameMetadata *const getGameMetadata () const`
GameMeatadata getter.
- const `modifiersCollection_t & getModifiers () const`
Modifiers getter.
- void `setName (std::string name)`
Sets name.
- void `setDescription (std::string description)`
Sets description.
- std::string `getName () const`
name getter.
- std::string `getDescription () const`
Description getter.
- `id_t getId () const`
id getter.
- bool `isId (id_t id) const`
Check if id is equal.

Static Public Attributes

- static constexpr `id_t INVALID_ID` {`std::numeric_limits<id_t>::min()`}
Value indicating that id is invalid.

Protected Member Functions

- void `setId (id_t id)`
id setter.
- void `validateIntegrity () const`
Check integrity of data.

Private Attributes

- [equipableSlots_t m_equipableOn](#)
Which equipment slots given item can be put on.
- [modifiersCollection_t m_modifiers](#)
Holds modifiers.
- `const GameMetadata *const m_gameMetadata`
Game Metadata.
- `id_t m_id {INVALID_ID}`
id.
- `std::string m_name`
name.
- `std::string m_description`
description.

5.17.1 Detailed Description

Represents an item in game.

Should be used in [GameData](#).

Definition at line 20 of file item.hpp.

5.17.2 Member Typedef Documentation

5.17.2.1 equipableSlots_t

```
using Item::equipableSlots_t = std::set<EquipmentSlot::id_t>
```

Collection.

Definition at line 24 of file item.hpp.

5.17.2.2 id_t

```
using BasicGameData::id_t = long long [inherited]
```

Type used for ids.

Definition at line 33 of file basicGamedata.hpp.

5.17.2.3 modifiersCollection_t

```
using StatModifyingEntity::modifiersCollection_t = std::vector<modifier_t> [inherited]
```

Collection type used to store modifiers.

Definition at line 38 of file statModifyingEntity.hpp.

5.17.2.4 modifier_t

```
using StatModifyingEntity::modifier_t = std::pair<Stat::id_t, Stat::value_t> [inherited]
```

Represents modification of stats <stat modified, value of modification>

Definition at line 36 of file statModifyingEntity.hpp.

5.17.3 Constructor & Destructor Documentation

5.17.3.1 Item()

```
Item::Item (
    GameData * gameData,
    std::string name,
    std::string description = "" )
```

Constructor.

Parameters

<i>gameData</i>	GameData to which item is added.
<i>name</i>	Name of item.
<i>description</i>	description.

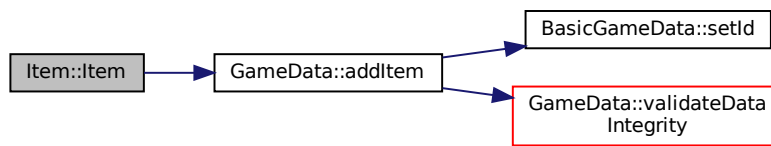
```
Item *item{new Item(gameData, "Stick")};
item->addModifier(1, 3);
item->setEquipableOn(3);
Item *item2{new Item(gameData, "Sunglasses", "They protect from sun")};
item2->addModifier(2, 100);
item2->setEquipableOn(2);
```

Definition at line 9 of file item.cpp.

```
10 : StatModifyingEntity(gameData, name, description) {
11   gameData->addItem(this);
12 }
```

References GameData::addItem().

Here is the call graph for this function:



5.17.4 Member Function Documentation

5.17.4.1 addModifier()

```

void StatModifyingEntity::addModifier (
    Stat::id_t statModified,
    Stat::value_t by ) [inherited]
  
```

Add modification of stats.

Parameters

<i>statModified</i>	modified.
<i>by</i>	Modify value.

Todo Check if entity has modifier of that [Stat](#) already.

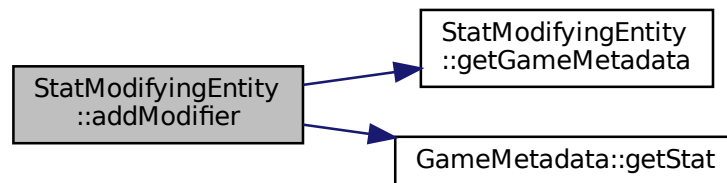
Definition at line 24 of file statModifyingEntity.cpp.

```

25 {
26
27     getGameMetadata()->getStat(statModified);
28     m_modifiers.push_back({statModified, by});
29 }
  
```

References [StatModifyingEntity::getGameMetadata\(\)](#), [GameMetadata::getStat\(\)](#), and [StatModifyingEntity::m_modifiers](#).

Here is the call graph for this function:



5.17.4.2 getDescription()

```
std::string BasicGameData::getDescription ( ) const [inherited]
```

Description getter.

Returns

Description.

Definition at line 38 of file `basicGamedata.cpp`.

```
38 { return m_description; };
```

References `BasicGameData::m_description`.

5.17.4.3 getEquipableSlots()

```
const Item::equipableSlots_t & Item::getEquipableSlots ( ) const
```

Get equipable slots.

Returns

`m_equipableOn`.

Definition at line 30 of file `item.cpp`.

```
30                                     {
31   return m_equipableOn;
32 }
```

References `m_equipableOn`.

5.17.4.4 `getGameMetadata()`

```
const GameMetadata *const StatModifyingEntity::getGameMetadata ( ) const [inherited]
```

GameMeatadata getter.

Returns

GemeMetadata used by instance.

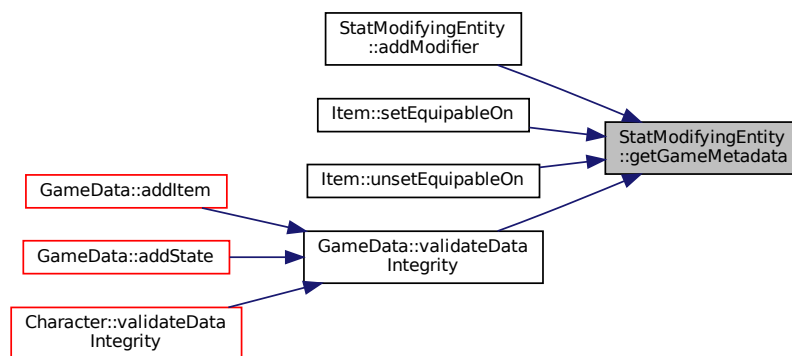
Definition at line 42 of file statModifyingEntity.cpp.

```
42 {
43     return m_gameMetadata;
44 }
```

References StatModifyingEntity::m_gameMetadata.

Referenced by StatModifyingEntity::addModifier(), setEquipableOn(), unsetEquipableOn(), and GameData::validateDataIntegrity().

Here is the caller graph for this function:



5.17.4.5 `getId()`

```
BasicGameData::id_t BasicGameData::getId ( ) const [inherited]
```

id getter.

Returns

id.

Exceptions

<code>exceptionIllegalId</code>	When tried to get id of instance that has <code>BasicGameData::INVALID_ID</code> .
---------------------------------	------------------------------------------------------------------------------------

Definition at line 40 of file `basicGamedata.cpp`.

```

40                                     {
41     if (m_id == INVALID_ID)
42         throw exceptionIllegalId();
43     return m_id;
44 };

```

References `BasicGameData::INVALID_ID`, and `BasicGameData::m_id`.

5.17.4.6 `getModifiers()`

```

const StatModifyingEntity::modifiersCollection_t & StatModifyingEntity::getModifiers ( ) const
[inherited]

```

Modifiers getter.

Returns

`::m_modifiers`

Definition at line 47 of file `statModifyingEntity.cpp`.

```

47                                     {
48     return m_modifiers;
49 }

```

References `StatModifyingEntity::m_modifiers`.

5.17.4.7 `getModifierValue()`

```

Stat::value_t StatModifyingEntity::getModifierValue (
    Stat::id_t id ) const [inherited]

```

Modifier value getter.

Parameters

<i>id</i>	Id of stat to get value of modifier of.
-----------	-----------------------------------------

Returns

Modifier value or 0 if Instance does not modify stat asked.

Definition at line 31 of file `statModifyingEntity.cpp`.

```

31                                     {
32     auto lookup{

```

```

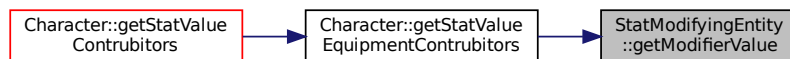
33     std::find_if(m_modifiers.begin(), m_modifiers.end(),
34                 [=](const modifier_t mod) { return mod.first == id; }));
35     // If not found modifier of asked stat.
36     if (lookup == m_modifiers.end())
37         return 0;
38
39     return lookup->second;
40 }

```

References StatModifyingEntity::m_modifiers.

Referenced by Character::getStatValueEquipmentContributors().

Here is the caller graph for this function:



5.17.4.8 getName()

```
std::string BasicGameData::getName ( ) const [inherited]
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References BasicGameData::m_name.

5.17.4.9 isEquipableOn()

```
bool Item::isEquipableOn (
    EquipmentSlot::id_t equipmentSlotId ) const
```

Checks whenever *Item* is equipable in given slot id.

Parameters

<i>equipmentSlotId</i>	Slot to check.
------------------------	----------------

Returns

True if [Item](#) is equipable in enquired slot. False if it is not.

Definition at line 24 of file item.cpp.

```

24                                     {
25     auto it{m_equipableOn.find(equipmentSlotId)};
26     const bool result{it != m_equipableOn.end()};
27     return result;
28 }
```

References `m_equipableOn`.

Referenced by `Character::equipItem()`.

Here is the caller graph for this function:

**5.17.4.10 isId()**

```

bool BasicGameData::isId (
    id_t id ) const [inherited]
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

Definition at line 48 of file basicGamedata.cpp.

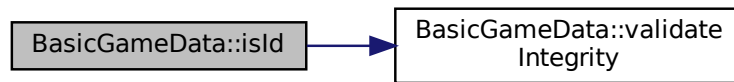
```

48                                     {
49     validateIntegrity();
50     bool res{m_id == id};
51     return res;
52 }
```

References `BasicGameData::m_id`, and `BasicGameData::validateIntegrity()`.

Referenced by `GameData::getItem()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.17.4.11 setDescription()

```
void BasicGameData::setDescription (
    std::string description ) [inherited]
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file basicGamedata.cpp.

```
32                                     {
33     m_description = description;
34 }
```

References `BasicGameData::m_description`.

5.17.4.12 setEquipableOn()

```
void Item::setEquipableOn (
    EquipmentSlot::id_t equipmentSlotId )
```

Make item equipable on given equipment slot.

Parameters

<i>equipmentSlotId</i>	On which EquipmentSlot should it be equipable.
------------------------	----------------------------------------------------------------

If it is already equipable on that slot nothing will be done.

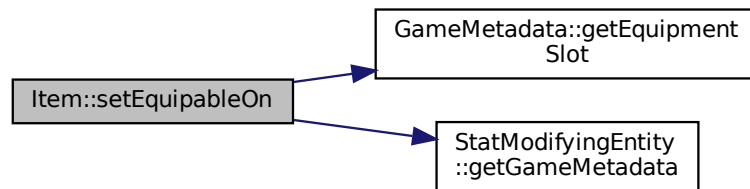
Definition at line 14 of file item.cpp.

```

14                                     {
15     getGameMetadata() -> getEquipmentSlot(equipmentSlotId);
16     m_equipableOn.insert(equipmentSlotId);
17 }
```

References [GameMetadata::getEquipmentSlot\(\)](#), [StatModifyingEntity::getGameMetadata\(\)](#), and [m_equipableOn](#).

Here is the call graph for this function:



5.17.4.13 setId()

```

void BasicGameData::setId (
    id_t id ) [protected], [inherited]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

Definition at line 17 of file basicGamedata.cpp.

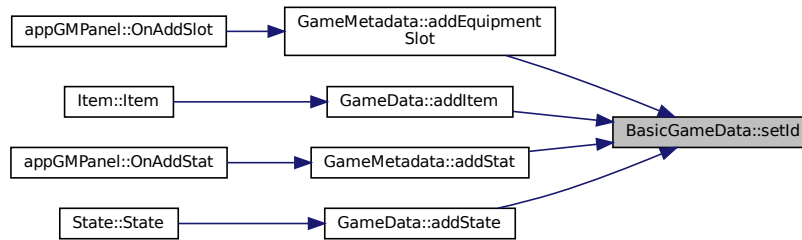
```

17                                     {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References [BasicGameData::INVALID_ID](#), and [BasicGameData::m_id](#).

Referenced by [GameMetadata::addEquipmentSlot\(\)](#), [GameData::addItem\(\)](#), [GameMetadata::addStat\(\)](#), and [GameData::addState\(\)](#).

Here is the caller graph for this function:



5.17.4.14 setName()

```
void BasicGameData::setName (
    std::string name ) [inherited]
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```
30 { m_name = name; }
```

References BasicGameData::m_name.

5.17.4.15 unsetEquipableOn()

```
void Item::unsetEquipableOn (
    EquipmentSlot::id_t equipmentSlotId )
```

Makes item no longer possible to equip onto given slot.

Parameters

<i>equipmentSlotId</i>	equipment slot id that item will be no longer equipable on.
------------------------	-------------------------------------------------------------

If it is already not equipable on that slot nothing will happen.

Definition at line 19 of file item.cpp.

```
19
```

```
{
```



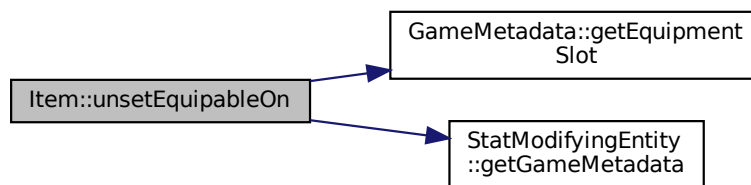
```

20  getGameMetadata\(\) -> getEquipmentSlot(equipmentSlotId);
21  m\_equipableOn.erase(equipmentSlotId);
22  }

```

References [GameMetadata::getEquipmentSlot\(\)](#), [StatModifyingEntity::getGameMetadata\(\)](#), and [m_equipableOn](#).

Here is the call graph for this function:



5.17.4.16 validateIntegrity()

```
void BasicGameData::validateIntegrity ( ) const [protected], [inherited]
```

Check integrity of data.

Checks if id is `INVALID_ID`.

Exceptions

<i>exceptionIllegalId</i>	id is illegal.
-------------------------------------------	----------------

Definition at line 22 of file `basicGamedata.cpp`.

```

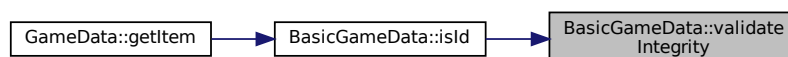
22  {
24  if (m_id == INVALID_ID) {
26      throw exceptionIllegalId();
27  }
28  }

```

References [BasicGameData::INVALID_ID](#), and [BasicGameData::m_id](#).

Referenced by [BasicGameData::isId\(\)](#).

Here is the caller graph for this function:



5.17.5 Member Data Documentation

5.17.5.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr],  
[inherited]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.17.5.2 m_description

```
std::string BasicGameData::m_description [private], [inherited]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by BasicGameData::getDescription(), and BasicGameData::setDescription().

5.17.5.3 m_equipableOn

```
equipableSlots_t Item::m_equipableOn [private]
```

Which equipment slots given item can be put on.

Definition at line 28 of file item.hpp.

Referenced by getEquipableSlots(), isEquipableOn(), setEquipableOn(), and unsetEquipableOn().

5.17.5.4 m_gameMetadata

```
const GameMetadata* const StatModifyingEntity::m_gameMetadata [private], [inherited]
```

Game Metadata.

Definition at line 45 of file statModifyingEntity.hpp.

Referenced by StatModifyingEntity::getGameMetadata().

5.17.5.5 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private], [inherited]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::isId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.17.5.6 m_modifiers

```
modifiersCollection_t StatModifyingEntity::m_modifiers [private], [inherited]
```

Holds modifiers.

Definition at line 42 of file statModifyingEntity.hpp.

Referenced by StatModifyingEntity::addModifier(), StatModifyingEntity::getModifiers(), and StatModifyingEntity::getModifierValue().

5.17.5.7 m_name

```
std::string BasicGameData::m_name [private], [inherited]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by BasicGameData::getName(), and BasicGameData::setName().

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

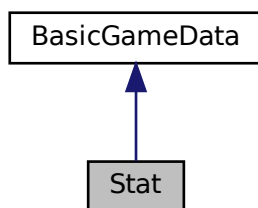
- [item.hpp](#)
- [item.cpp](#)

5.18 Stat Class Reference

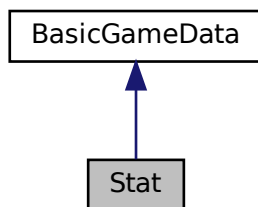
Statistics.

```
#include <stat.hpp>
```

Inheritance diagram for Stat:



Collaboration diagram for Stat:



Public Types

- using `value_t` = long long
Type of stat value.
- using `id_t` = long long
Type used for ids.

Public Member Functions

- `Stat` (std::string name, std::string description="")
Constructor.
- void `setName` (std::string name)
Sets name.

- void [setDescription](#) (std::string description)
Sets description.
- std::string [getName](#) () const
name getter.
- std::string [getDescription](#) () const
Description getter.
- [id_t](#) [getId](#) () const
id getter.
- bool [isId](#) ([id_t](#) id) const
Check if id is equal.

Static Public Attributes

- static constexpr [id_t](#) [INVALID_ID](#) {std::numeric_limits<[id_t](#)>::min()}
Value indicating that id is invalid.

Protected Member Functions

- void [setId](#) ([id_t](#) id)
id setter.
- void [validateIntegrity](#) () const
Check integrity of data.

Private Attributes

- [id_t](#) [m_id](#) {[INVALID_ID](#)}
id.
- std::string [m_name](#)
name.
- std::string [m_description](#)
description.

5.18.1 Detailed Description

Statistics.

Definition at line 14 of file stat.hpp.

5.18.2 Member Typedef Documentation

5.18.2.1 id_t

```
using BasicGameData::id_t = long long [inherited]
```

Type used for ids.

Definition at line 33 of file basicGamedata.hpp.

5.18.2.2 value_t

```
using Stat::value_t = long long
```

Type of stat value.

Definition at line 17 of file stat.hpp.

5.18.3 Constructor & Destructor Documentation

5.18.3.1 Stat()

```
Stat::Stat (
    std::string name,
    std::string description = "" )
```

Constructor.

Parameters

<i>name</i>	Name of statistics.
<i>description</i>	Optional description of statistics;

Definition at line 9 of file stat.cpp.

```
10 : BasicGameData(name, description) {}
```

5.18.4 Member Function Documentation

5.18.4.1 getDescription()

```
std::string BasicGameData::getDescription ( ) const [inherited]
```

Description getter.

Returns

Description.

Definition at line 38 of file basicGamedata.cpp.

```
38 { return m_description; };
```

References BasicGameData::m_description.

5.18.4.2 getId()

```
BasicGameData::id_t BasicGameData::getId ( ) const [inherited]
```

id getter.

Returns

id.

Exceptions

<i>exceptionIllegalId</i>	When tred to get id of instance that has BasicGameData::INVALID_ID .
---------------------------	--------------------------------------------------------------------------------------

Definition at line 40 of file basicGamedata.cpp.

```
40                                     {
41     if (m_id == INVALID_ID)
42         throw exceptionIllegalId();
43     return m_id;
44 };
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

5.18.4.3 getName()

```
std::string BasicGameData::getName ( ) const [inherited]
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References BasicGameData::m_name.

5.18.4.4 isId()

```
bool BasicGameData::isId (
    id_t id ) const [inherited]
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

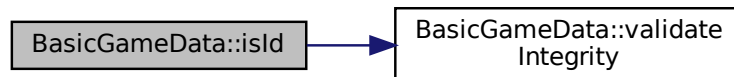
Definition at line 48 of file basicGamedata.cpp.

```
48     {  
49         validateIntegrity();  
50         bool res{m_id == id};  
51         return res;  
52     }
```

References BasicGameData::m_id, and BasicGameData::validateIntegrity().

Referenced by GameData::getItem().

Here is the call graph for this function:



Here is the caller graph for this function:



5.18.4.5 setDescription()

```
void BasicGameData::setDescription (  
    std::string description ) [inherited]
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file basicGamedata.cpp.

```

32                                     {
33     m_description = description;
34 }
```

References BasicGameData::m_description.

5.18.4.6 setId()

```

void BasicGameData::setId (
    id_t id ) [protected], [inherited]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

Definition at line 17 of file basicGamedata.cpp.

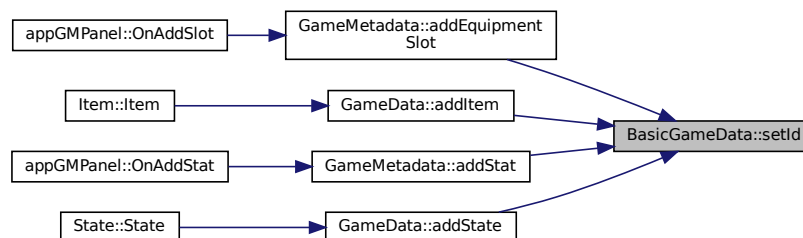
```

17                                     {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by GameMetadata::addEquipmentSlot(), GameData::addItem(), GameMetadata::addStat(), and GameData::addState().

Here is the caller graph for this function:



5.18.4.7 setName()

```
void BasicGameData::setName (  
    std::string name ) [inherited]
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```
30 { m_name = name; }
```

References BasicGameData::m_name.

5.18.4.8 validateIntegrity()

```
void BasicGameData::validateIntegrity ( ) const [protected], [inherited]
```

Check integrity of data.

Checks if id is INVALID_ID.

Exceptions

<i>exceptionIllegalId</i>	id is illegal.
-------------------------------------------	----------------

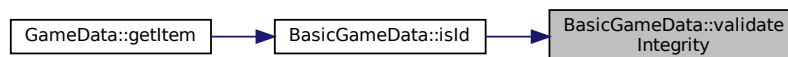
Definition at line 22 of file basicGamedata.cpp.

```
22 {
24     if (m_id == INVALID_ID) {
26         throw exceptionIllegalId();
27     }
28 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by BasicGameData::isId().

Here is the caller graph for this function:



5.18.5 Member Data Documentation

5.18.5.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr],  
[inherited]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.18.5.2 m_description

```
std::string BasicGameData::m_description [private], [inherited]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by BasicGameData::getDescription(), and BasicGameData::setDescription().

5.18.5.3 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private], [inherited]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::isId(), BasicGameData::setId(), and BasicGameData↵
::validateIntegrity().

5.18.5.4 m_name

```
std::string BasicGameData::m_name [private], [inherited]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by BasicGameData::getName(), and BasicGameData::setName().

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

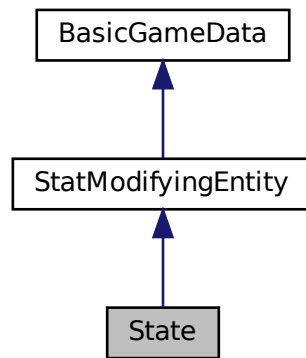
- [stat.hpp](#)
- [stat.cpp](#)

5.19 State Class Reference

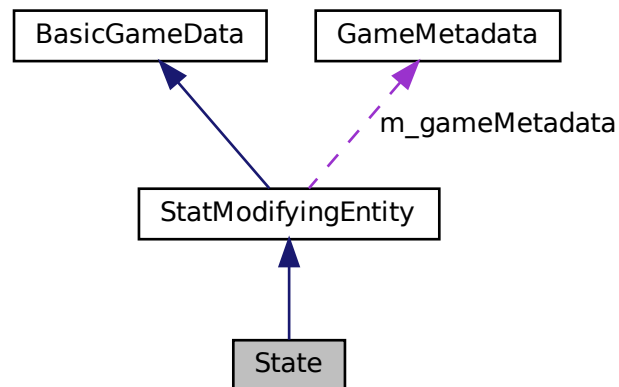
Represents [State](#) in game.

```
#include <state.hpp>
```

Inheritance diagram for State:



Collaboration diagram for State:



Public Types

- using `modifier_t` = `std::pair< Stat::id_t, Stat::value_t >`
Represents modification of stats <stat modified, value of modification>
- using `modifiersCollection_t` = `std::vector< modifier_t >`
Collection type used to store modifiers.
- using `id_t` = `long long`
Type used for ids.

Public Member Functions

- [State](#) ([GameData](#) *gameData, std::string name, std::string description="")
Constructor.
- void [addModifier](#) ([Stat::id_t](#), [Stat::value_t](#) by)
Add modification of stats.
- [Stat::value_t](#) [getModifierValue](#) ([Stat::id_t](#) id) const
Modifier value getter.
- const [GameMetadata](#) *const [getGameMetadata](#) () const
GameMeatadata getter.
- const [modifiersCollection_t](#) & [getModifiers](#) () const
Modifiers getter.
- void [setName](#) (std::string name)
Sets name.
- void [setDescription](#) (std::string description)
Sets description.
- std::string [getName](#) () const
name getter.
- std::string [getDescription](#) () const
Description getter.
- [id_t](#) [getId](#) () const
id getter.
- bool [isId](#) ([id_t](#) id) const
Check if id is equal.

Static Public Attributes

- static constexpr [id_t](#) [INVALID_ID](#) {std::numeric_limits<[id_t](#)>::min()}
Value indicating that id is invalid.

Protected Member Functions

- void [setId](#) ([id_t](#) id)
id setter.
- void [validateIntegrity](#) () const
Check itegrity of data.

Private Attributes

- [modifiersCollection_t](#) [m_modifiers](#)
Holds modifiers.
- const [GameMetadata](#) *const [m_gameMetadata](#)
Game Metadata.
- [id_t](#) [m_id](#) {[INVALID_ID](#)}
id.
- std::string [m_name](#)
name.
- std::string [m_description](#)
description.

5.19.1 Detailed Description

Reperesents [State](#) in game.

Definition at line 16 of file state.hpp.

5.19.2 Member Typedef Documentation

5.19.2.1 id_t

```
using BasicGameData::id_t = long long [inherited]
```

Type used for ids.

Definition at line 33 of file basicGamedata.hpp.

5.19.2.2 modifiersCollection_t

```
using StatModifyingEntity::modifiersCollection_t = std::vector<modifier_t> [inherited]
```

Collection type used to store modifiers.

Definition at line 38 of file statModifyingEntity.hpp.

5.19.2.3 modifier_t

```
using StatModifyingEntity::modifier_t = std::pair<Stat::id_t, Stat::value_t> [inherited]
```

Represents modification of stats <stat modified, value of modification>

Definition at line 36 of file statModifyingEntity.hpp.

5.19.3 Constructor & Destructor Documentation

5.19.3.1 State()

```
State::State (
    GameData * gameData,
    std::string name,
    std::string description = "" )
```

Constructor.

Parameters

<i>gameData</i>	Gamedata into which State will be added
<i>name</i>	Name of state.
<i>description</i>	Description of state.

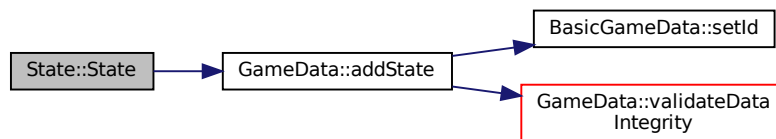
Definition at line 9 of file state.cpp.

```

10     : StatModifyingEntity(gameData, name, description) {
11     gameData->addState(this);
12 }
```

References `GameData::addState()`.

Here is the call graph for this function:



5.19.4 Member Function Documentation

5.19.4.1 addModifier()

```

void StatModifyingEntity::addModifier (
    Stat::id_t statModified,
    Stat::value_t by ) [inherited]
```

Add modification of stats.

Parameters

<i>statModified</i>	modified.
<i>by</i>	Modify value.

Todo Check if entity has modifier of that [Stat](#) already.

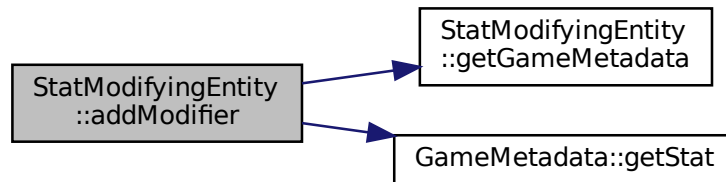
Definition at line 24 of file statModifyingEntity.cpp.

```

25                                     {
26
27     getGameMetadata()->getStat(statModified);
28     m_modifiers.push_back({statModified, by});
29 }
```


References StatModifyingEntity::getGameMetadata(), GameMetadata::getStat(), and StatModifyingEntity::m_↵ modifiers.

Here is the call graph for this function:



5.19.4.2 getDescription()

```
std::string BasicGameData::getDescription ( ) const [inherited]
```

Description getter.

Returns

Description.

Definition at line 38 of file basicGamedata.cpp.

```
38 { return m_description; };
```

References BasicGameData::m_description.

5.19.4.3 getGameMetadata()

```
const GameMetadata *const StatModifyingEntity::getGameMetadata ( ) const [inherited]
```

GameMeatadata getter.

Returns

GameMetadata used by instance.

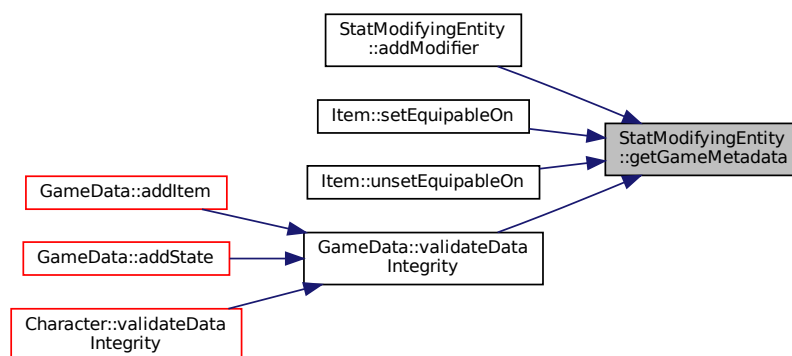
Definition at line 42 of file statModifyingEntity.cpp.

```
42                                     {
43     return m_gameMetadata;
44 }
```

References StatModifyingEntity::m_gameMetadata.

Referenced by StatModifyingEntity::addModifier(), Item::setEquipableOn(), Item::unsetEquipableOn(), and GameData::validateDataIntegrity().

Here is the caller graph for this function:

**5.19.4.4 getId()**

```
BasicGameData::id_t BasicGameData::getId ( ) const [inherited]
```

id getter.

Returns

id.

Exceptions

<i>exceptionIllegalId</i>	When tried to get id of instance that has BasicGameData::INVALID_ID .
-------------------------------------------	---------------------------------------------------------------------------------------

Definition at line 40 of file basicGamedata.cpp.

```
40                                     {
41     if (m_id == INVALID_ID)
42         throw exceptionIllegalId();
43     return m_id;
44 };
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

5.19.4.5 getModifiers()

```
const StatModifyingEntity::modifiersCollection_t & StatModifyingEntity::getModifiers ( ) const
[inherited]
```

Modifiers getter.

Returns

::m_modifiers

Definition at line 47 of file statModifyingEntity.cpp.

```
47                                     {
48     return m_modifiers;
49 }
```

References StatModifyingEntity::m_modifiers.

5.19.4.6 getModifierValue()

```
Stat::value_t StatModifyingEntity::getModifierValue (
    Stat::id_t id ) const [inherited]
```

Modifier value getter.

Parameters

<i>id</i>	Id of stat to get value of modifier of.
-----------	-----------------------------------------

Returns

Modifier value or 0 if Instance does not modify stat asked.

Definition at line 31 of file statModifyingEntity.cpp.

```
31                                     {
32     auto lookup{
33         std::find_if(m_modifiers.begin(), m_modifiers.end(),
34             [=](const modifier_t mod) { return mod.first == id; });
35     // If not found modifier of asked stat.
36     if (lookup == m_modifiers.end())
37         return 0;
38     return lookup->second;
39 }
40 }
```

References StatModifyingEntity::m_modifiers.

Referenced by Character::getStatValueEquipmentContributors().

Here is the caller graph for this function:



5.19.4.7 getName()

```
std::string BasicGameData::getName ( ) const [inherited]
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References BasicGameData::m_name.

5.19.4.8 isId()

```
bool BasicGameData::isId (
    id_t id ) const [inherited]
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

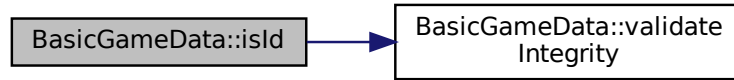
Definition at line 48 of file basicGamedata.cpp.

```
48 {
49     validateIntegrity();
50     bool res{m_id == id};
51     return res;
52 }
```

References BasicGameData::m_id, and BasicGameData::validateIntegrity().

Referenced by `GameData::getItem()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.19.4.9 setDescription()

```
void BasicGameData::setDescription (
    std::string description ) [inherited]
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file `basicGamedata.cpp`.

```
32 {
33     m_description = description;
34 }
```

References `BasicGameData::m_description`.

5.19.4.10 setId()

```
void BasicGameData::setId (
    id_t id ) [protected], [inherited]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

Definition at line 17 of file basicGamedata.cpp.

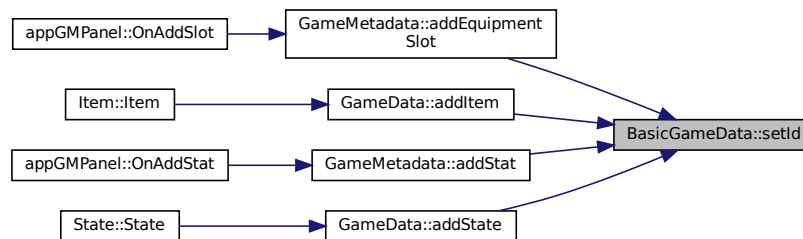
```

17         {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by GameMetadata::addEquipmentSlot(), GameData::addItem(), GameMetadata::addStat(), and GameData::addState().

Here is the caller graph for this function:



5.19.4.11 setName()

```

void BasicGameData::setName (
    std::string name ) [inherited]
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```

30 { m_name = name; }
```

References BasicGameData::m_name.

5.19.4.12 validateIntegrity()

```

void BasicGameData::validateIntegrity ( ) const [protected], [inherited]
```

Check integrity of data.

Checks if id is INVALID_ID.

Exceptions

<code>exceptionIllegalId</code>	id is illegal.
-------------------------------------------------	----------------

Definition at line 22 of file basicGamedata.cpp.

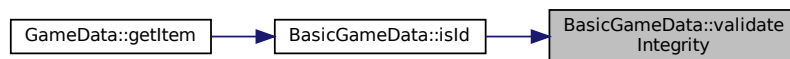
```

22
24     if (m_id == INVALID_ID) {
26         throw exceptionIllegalId();
27     }
28 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by BasicGameData::isId().

Here is the caller graph for this function:



5.19.5 Member Data Documentation

5.19.5.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr],
[inherited]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.19.5.2 m_description

```
std::string BasicGameData::m_description [private], [inherited]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by BasicGameData::getDescription(), and BasicGameData::setDescription().

5.19.5.3 m_gameMetadata

```
const GameMetadata* const StatModifyingEntity::m_gameMetadata [private], [inherited]
```

Game Metadata.

Definition at line 45 of file statModifyingEntity.hpp.

Referenced by StatModifyingEntity::getGameMetadata().

5.19.5.4 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private], [inherited]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::isId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.19.5.5 m_modifiers

```
modifiersCollection_t StatModifyingEntity::m_modifiers [private], [inherited]
```

Holds modifiers.

Definition at line 42 of file statModifyingEntity.hpp.

Referenced by StatModifyingEntity::addModifier(), StatModifyingEntity::getModifiers(), and StatModifyingEntity::getModifierValue().

5.19.5.6 m_name

```
std::string BasicGameData::m_name [private], [inherited]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by BasicGameData::getName(), and BasicGameData::setName().

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

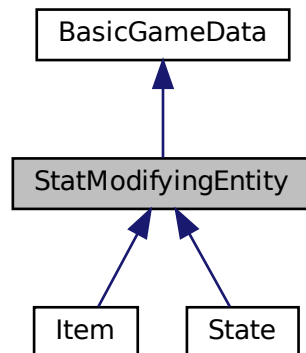
- [state.hpp](#)
- [state.cpp](#)

5.20 StatModifyingEntity Class Reference

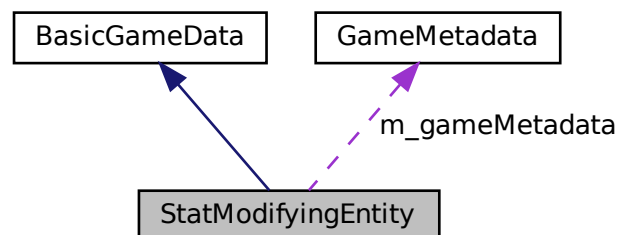
Represents collection of stat modifiers.

```
#include <statModifyingEntity.hpp>
```

Inheritance diagram for StatModifyingEntity:



Collaboration diagram for StatModifyingEntity:



Public Types

- using `modifier_t` = `std::pair< Stat::id_t, Stat::value_t >`
Represents modification of stats <stat modified, value of modification>
- using `modifiersCollection_t` = `std::vector< modifier_t >`
Collection type used to store modifiers.
- using `id_t` = `long long`
Type used for ids.

Public Member Functions

- [StatModifyingEntity](#) (const [GameMetadata](#) *const gameMetadata, std::string name, std::string description="")
Constructor.
- void [addModifier](#) ([Stat::id_t](#), [Stat::value_t](#) by)
Add modification of stats.
- [Stat::value_t](#) [getModifierValue](#) ([Stat::id_t](#) id) const
Modifier value getter.
- const [GameMetadata](#) *const [getGameMetadata](#) () const
GameMeatadata getter.
- const [modifiersCollection_t](#) & [getModifiers](#) () const
Modifiers getter.
- void [setName](#) (std::string name)
Sets name.
- void [setDescription](#) (std::string description)
Sets description.
- std::string [getName](#) () const
name getter.
- std::string [getDescription](#) () const
Description getter.
- [id_t](#) [getId](#) () const
id getter.
- bool [isId](#) ([id_t](#) id) const
Check if id is equal.

Static Public Attributes

- static constexpr [id_t](#) [INVALID_ID](#) {std::numeric_limits<[id_t](#)>::min()}
Value indicating that id is invalid.

Protected Member Functions

- void [setId](#) ([id_t](#) id)
id setter.
- void [validateIntegrity](#) () const
Check itegrity of data.

Private Attributes

- [modifiersCollection_t](#) [m_modifiers](#)
Holds modifiers.
- const [GameMetadata](#) *const [m_gameMetadata](#)
Game Metadata.
- [id_t](#) [m_id](#) {[INVALID_ID](#)}
id.
- std::string [m_name](#)
name.
- std::string [m_description](#)
description.

5.20.1 Detailed Description

Represents collection of stat modifiers.

It modifies Stats so it needs access to information about what stats do exist. Associates instance with game↔ Metadata.

Definition at line 33 of file statModifyingEntity.hpp.

5.20.2 Member Typedef Documentation

5.20.2.1 id_t

```
using BasicGameData::id_t = long long [inherited]
```

Type used for ids.

Definition at line 33 of file basicGamedata.hpp.

5.20.2.2 modifiersCollection_t

```
using StatModifyingEntity::modifiersCollection_t = std::vector<modifier_t>
```

Collection type used to store modifiers.

Definition at line 38 of file statModifyingEntity.hpp.

5.20.2.3 modifier_t

```
using StatModifyingEntity::modifier_t = std::pair<Stat::id_t, Stat::value_t>
```

Represents modification of stats <stat modified, value of modification>

Definition at line 36 of file statModifyingEntity.hpp.

5.20.3 Constructor & Destructor Documentation

5.20.3.1 StatModifyingEntity()

```
StatModifyingEntity::StatModifyingEntity (
    const GameMetadata *const gameMetadata,
    std::string name,
    std::string description = "" )
```

Constructor.

Parameters

<i>gameMetadata</i>	gameMetadata that instance is about.
<i>name</i>	Name
<i>description</i>	Description.

Exceptions

<i>exceptionInvalidGameMetadata</i>	
-----------------------------------------------------	--

Definition at line 14 of file statModifyingEntity.cpp.

```

17     : BasicGameData(name, description), m_gameMetadata(gameMetadata) {
19     if (gameMetadata == nullptr)
20         throw exceptionInvalidGameMetadata();
21 }
```

5.20.4 Member Function Documentation

5.20.4.1 addModifier()

```

void StatModifyingEntity::addModifier (
    Stat::id_t statModified,
    Stat::value_t by )
```

Add modification of stats.

Parameters

<i>statModified</i>	modified.
<i>by</i>	Modify value.

Todo Check if entity has modifier of that [Stat](#) already.

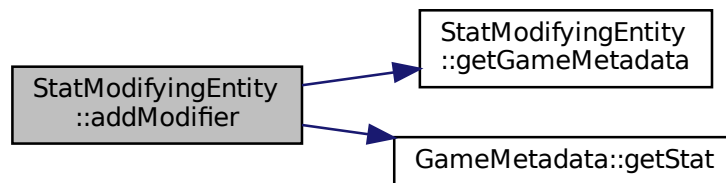
Definition at line 24 of file statModifyingEntity.cpp.

```

25                                     {
26
27     getGameMetadata()->getStat(statModified);
28     m_modifiers.push_back({statModified, by});
29 }
```

References [getGameMetadata\(\)](#), [GameMetadata::getStat\(\)](#), and [m_modifiers](#).

Here is the call graph for this function:



5.20.4.2 getDescription()

```
std::string BasicGameData::getDescription ( ) const [inherited]
```

Description getter.

Returns

Description.

Definition at line 38 of file `basicGamedata.cpp`.

```
38 { return m_description; };
```

References `BasicGameData::m_description`.

5.20.4.3 getGameMetadata()

```
const GameMetadata *const StatModifyingEntity::getGameMetadata ( ) const
```

GameMeatadata getter.

Returns

GameMetadata used by instance.

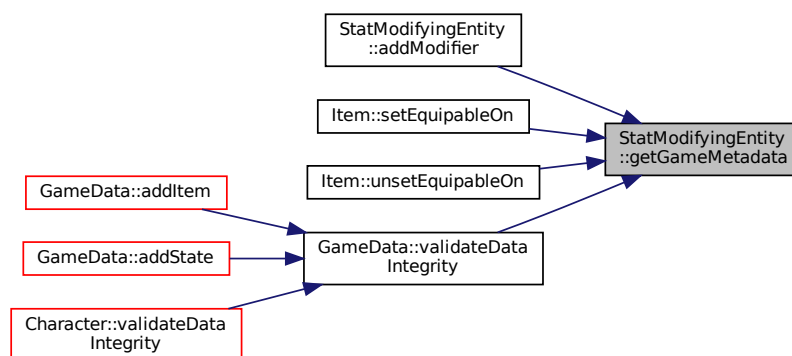
Definition at line 42 of file statModifyingEntity.cpp.

```
42                                     {
43     return m_gameMetadata;
44 }
```

References m_gameMetadata.

Referenced by addModifier(), Item::setEquipableOn(), Item::unsetEquipableOn(), and GameData::validateDataIntegrity().

Here is the caller graph for this function:

**5.20.4.4 getId()**

```
BasicGameData::id_t BasicGameData::getId ( ) const [inherited]
```

id getter.

Returns

id.

Exceptions

<i>exceptionIllegalId</i>	When tried to get id of instance that has BasicGameData::INVALID_ID .
-------------------------------------------	---------------------------------------------------------------------------------------

Definition at line 40 of file basicGamedata.cpp.

```
40                                     {
41     if (m_id == INVALID_ID)
42         throw exceptionIllegalId();
43     return m_id;
44 };
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

5.20.4.5 getModifiers()

```
const StatModifyingEntity::modifiersCollection_t & StatModifyingEntity::getModifiers ( ) const
```

Modifiers getter.

Returns

::m_modifiers

Definition at line 47 of file statModifyingEntity.cpp.

```
47 {
48     return m_modifiers;
49 }
```

References m_modifiers.

5.20.4.6 getModifierValue()

```
Stat::value_t StatModifyingEntity::getModifierValue (
    Stat::id_t id ) const
```

Modifier value getter.

Parameters

<i>id</i>	Id of stat to get value of modifier of.
-----------	-----------------------------------------

Returns

Modifier value or 0 if Instance does not modify stat asked.

Definition at line 31 of file statModifyingEntity.cpp.

```
31 {
32     auto lookup{
33         std::find_if(m_modifiers.begin(), m_modifiers.end(),
34             [=](const modifier_t mod) { return mod.first == id; });
35     // If not found modifier of asked stat.
36     if (lookup == m_modifiers.end())
37         return 0;
38     return lookup->second;
39 }
40 }
```

References m_modifiers.

Referenced by Character::getStatValueEquipmentContributors().

Here is the caller graph for this function:



5.20.4.7 getName()

```
std::string BasicGameData::getName ( ) const [inherited]
```

name getter.

Returns

name

Definition at line 36 of file basicGamedata.cpp.

```
36 { return m_name; }
```

References BasicGameData::m_name.

5.20.4.8 isId()

```
bool BasicGameData::isId (
    id_t id ) const [inherited]
```

Check if id is equal.

Parameters

<i>id</i>	Id to check.
-----------	--------------

Returns

True if id is same as parameter.

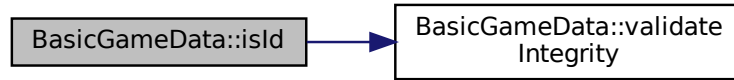
Definition at line 48 of file basicGamedata.cpp.

```
48 {
49     validateIntegrity();
50     bool res{m_id == id};
51     return res;
52 }
```

References BasicGameData::m_id, and BasicGameData::validateIntegrity().

Referenced by `GameData::getItem()`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.20.4.9 setDescription()

```
void BasicGameData::setDescription (
    std::string description ) [inherited]
```

Sets description.

Parameters

<i>description</i>	Description to be set.
--------------------	------------------------

Definition at line 32 of file `basicGamedata.cpp`.

```
32 {
33     m_description = description;
34 }
```

References `BasicGameData::m_description`.

5.20.4.10 setId()

```
void BasicGameData::setId (
    id_t id ) [protected], [inherited]
```

id setter.

Parameters

<i>id</i>	Id to set. If id to be set is INVALID_ID this will not be set.
-----------	----------------------------------------------------------------

Definition at line 17 of file basicGamedata.cpp.

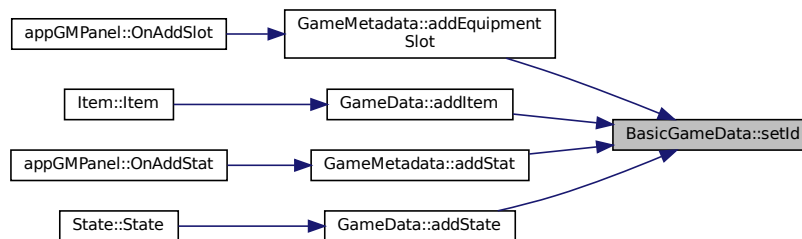
```

17         {
18     if (id == INVALID_ID)
19         return;
20     m_id = id;
21 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by GameMetadata::addEquipmentSlot(), GameData::addItem(), GameMetadata::addStat(), and GameData::addState().

Here is the caller graph for this function:

**5.20.4.11 setName()**

```

void BasicGameData::setName (
    std::string name ) [inherited]
```

Sets name.

Parameters

<i>name</i>	name to be set.
-------------	-----------------

Definition at line 30 of file basicGamedata.cpp.

```

30 { m_name = name; }
```

References BasicGameData::m_name.

5.20.4.12 validateIntegrity()

```

void BasicGameData::validateIntegrity ( ) const [protected], [inherited]
```

Check integrity of data.

Checks if id is INVALID_ID.

Exceptions

<code>exceptionIllegalId</code>	id is illegal.
-------------------------------------------------	----------------

Definition at line 22 of file basicGamedata.cpp.

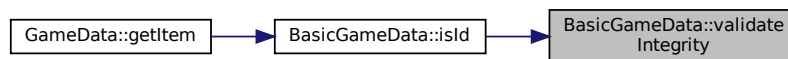
```

22
24     if (m_id == INVALID_ID) {
26         throw exceptionIllegalId();
27     }
28 }
```

References BasicGameData::INVALID_ID, and BasicGameData::m_id.

Referenced by BasicGameData::isId().

Here is the caller graph for this function:



5.20.5 Member Data Documentation

5.20.5.1 INVALID_ID

```
constexpr id_t BasicGameData::INVALID_ID {std::numeric_limits<id_t>::min()} [static], [constexpr],
[inherited]
```

Value indicating that id is invalid.

Definition at line 35 of file basicGamedata.hpp.

Referenced by BasicGameData::getId(), BasicGameData::setId(), and BasicGameData::validateIntegrity().

5.20.5.2 m_description

```
std::string BasicGameData::m_description [private], [inherited]
```

description.

Definition at line 43 of file basicGamedata.hpp.

Referenced by BasicGameData::getDescription(), and BasicGameData::setDescription().

5.20.5.3 m_gameMetadata

```
const GameMetadata* const StatModifyingEntity::m_gameMetadata [private]
```

Game Metadata.

Definition at line 45 of file statModifyingEntity.hpp.

Referenced by `getGameMetadata()`.

5.20.5.4 m_id

```
id_t BasicGameData::m_id {INVALID_ID} [private], [inherited]
```

id.

Definition at line 39 of file basicGamedata.hpp.

Referenced by `BasicGameData::getId()`, `BasicGameData::isId()`, `BasicGameData::setId()`, and `BasicGameData::validateIntegrity()`.

5.20.5.5 m_modifiers

```
modifiersCollection_t StatModifyingEntity::m_modifiers [private]
```

Holds modifiers.

Definition at line 42 of file statModifyingEntity.hpp.

Referenced by `addModifier()`, `getModifiers()`, and `getModifierValue()`.

5.20.5.6 m_name

```
std::string BasicGameData::m_name [private], [inherited]
```

name.

Definition at line 41 of file basicGamedata.hpp.

Referenced by `BasicGameData::getName()`, and `BasicGameData::setName()`.

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

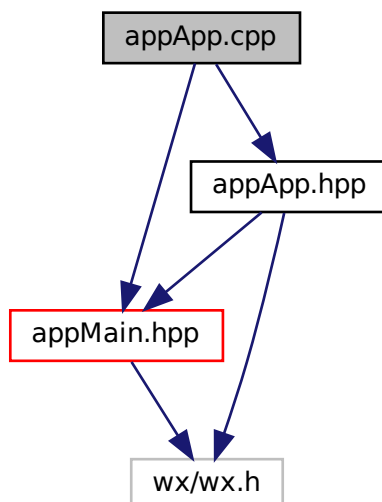
- [statModifyingEntity.hpp](#)
- [statModifyingEntity.cpp](#)

Chapter 6

File Documentation

6.1 appApp.cpp File Reference

```
#include "appApp.hpp"  
#include "appMain.hpp"  
Include dependency graph for appApp.cpp:
```



Functions

- [wxIMPLEMENT_APP](#) (`appApp`)

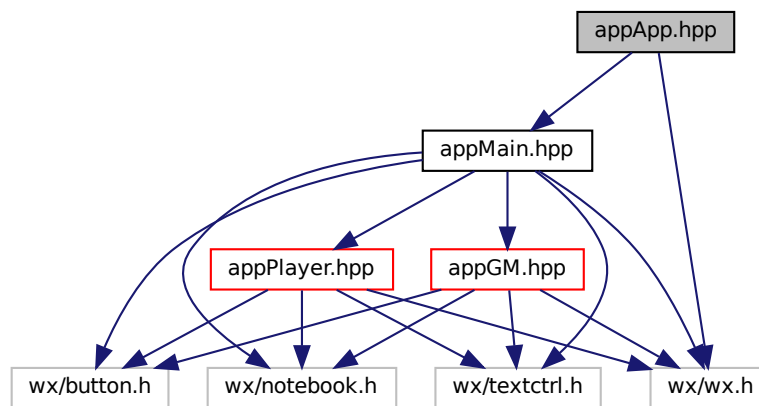
6.1.1 Function Documentation

6.1.1.1 wxIMPLEMENT_APP()

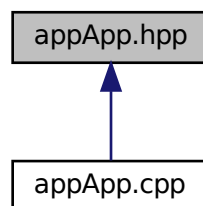
```
wxIMPLEMENT_APP (
    appApp )
```

6.2 appApp.hpp File Reference

```
#include "appMain.hpp"
#include <wx/wx.h>
Include dependency graph for appApp.hpp:
```



This graph shows which files directly or indirectly include this file:



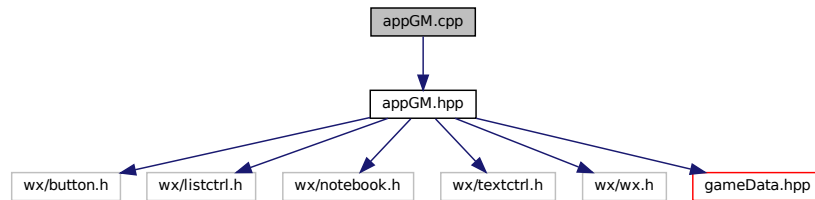
Classes

- class `appApp`

6.3 appGM.cpp File Reference

```
#include "appGM.hpp"
```

Include dependency graph for appGM.cpp:



Functions

- [wxBEGIN_EVENT_TABLE](#) ([appGMPanel](#), wxPanel) [wxEND_EVENT_TABLE](#)() [appGMPanel](#)

6.3.1 Function Documentation

6.3.1.1 wxBEGIN_EVENT_TABLE()

```

wxBEGIN_EVENT_TABLE (
    appGMPanel ,
    wxPanel )

```

Definition at line 3 of file appGM.cpp.

```

11 : wxPanel(parent, wxID_ANY), m_gamedata(new GameData) {
12   gmNotebook = new wxNotebook(this, wxID_ANY);
13
14   addElementPanel = new wxPanel(gmNotebook, wxID_ANY);
15   addElementSizer = new wxBoxSizer(wxVERTICAL);
16
17   addStatButton = new wxButton(addElementPanel, ID_ADD_STAT, "Add stat");
18   addEquipmentButton =
19     new wxButton(addElementPanel, ID_ADD_EQUIPMENT, "Add item");
20   addSlotButton = new wxButton(addElementPanel, ID_ADD_SLOT, "Add slot");
21   addStateButton = new wxButton(addElementPanel, ID_ADD_STATE, "Add state");
22
23   addElementSizer->Add(addStatButton, 0, wxALL, 5);
24   addElementSizer->Add(addEquipmentButton, 0, wxALL, 5);
25   addElementSizer->Add(addSlotButton, 0, wxALL, 5);
26   addElementSizer->Add(addStateButton, 0, wxALL, 5);
27
28   addElementPanel->SetSizer(addElementSizer);
29
30   statsPanel = new wxPanel(gmNotebook, wxID_ANY);
31   statsSizer = new wxBoxSizer(wxVERTICAL);
32   statsListCtrl = new wxListCtrl(statsPanel, wxID_ANY, wxDefaultPosition,
33     wxDefaultSize, wxLC_REPORT | wxLC_SINGLE_SEL);
34   statsListCtrl->InsertColumn(0, "Name");
35   statsListCtrl->InsertColumn(1, "Description");
36   statsListCtrl->SetColumnWidth(0, 150);
37   statsListCtrl->SetColumnWidth(1, 400);
38   statsSizer->Add(statsListCtrl, 1, wxEXPAND | wxALL, 5);
39   statsPanel->SetSizer(statsSizer);
40
41   eqPanel = new wxPanel(gmNotebook, wxID_ANY);

```

```

42  eqSizer = new wxBoxSizer(wxVERTICAL);
43  eqListCtrl = new wxListCtrl(eqPanel, wxID_ANY, wxDefaultPosition,
44                               wxDefaultSize, wxLC_REPORT | wxLC_SINGLE_SEL);
45  eqListCtrl->InsertColumn(0, "Name");
46  eqListCtrl->SetColumnWidth(0, 150);
47  eqSizer->Add(eqListCtrl, 1, wxEXPAND | wxALL, 5);
48  eqPanel->SetSizer(eqSizer);
49
50  slotPanel = new wxPanel(gmNotebook, wxID_ANY);
51  slotSizer = new wxBoxSizer(wxVERTICAL);
52  slotListCtrl = new wxListCtrl(slotPanel, wxID_ANY, wxDefaultPosition,
53                               wxDefaultSize, wxLC_REPORT | wxLC_SINGLE_SEL);
54  slotListCtrl->InsertColumn(0, "Name");
55  slotListCtrl->SetColumnWidth(0, 150);
56  slotSizer->Add(slotListCtrl, 1, wxEXPAND | wxALL, 5);
57  slotPanel->SetSizer(slotSizer);
58
59  statePanel = new wxPanel(gmNotebook, wxID_ANY);
60  stateSizer = new wxBoxSizer(wxVERTICAL);
61  stateListCtrl = new wxListCtrl(statePanel, wxID_ANY, wxDefaultPosition,
62                               wxDefaultSize, wxLC_REPORT | wxLC_SINGLE_SEL);
63  stateListCtrl->InsertColumn(0, "Name");
64  stateListCtrl->SetColumnWidth(0, 150);
65  stateSizer->Add(stateListCtrl, 1, wxEXPAND | wxALL, 5);
66  statePanel->SetSizer(stateSizer);
67
68  gmNotebook->AddPage(addElementPanel, "Add");
69  gmNotebook->AddPage(statsPanel, "Stats");
70  gmNotebook->AddPage(eqPanel, "Items");
71  gmNotebook->AddPage(slotPanel, "Slots");
72  gmNotebook->AddPage(statePanel, "States");
73
74  wxBoxSizer *mainSizer = new wxBoxSizer(wxVERTICAL);
75  mainSizer->Add(gmNotebook, 1, wxEXPAND | wxALL, 5);
76
77  SetSizer(mainSizer);
78 }

```

References ID_ADD_EQUIPMENT, ID_ADD_SLOT, ID_ADD_STAT, and ID_ADD_STATE.

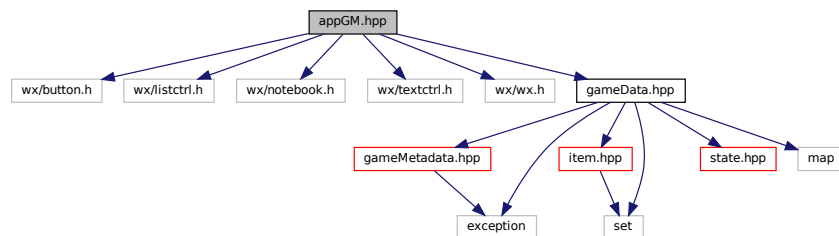
6.4 appGM.hpp File Reference

```

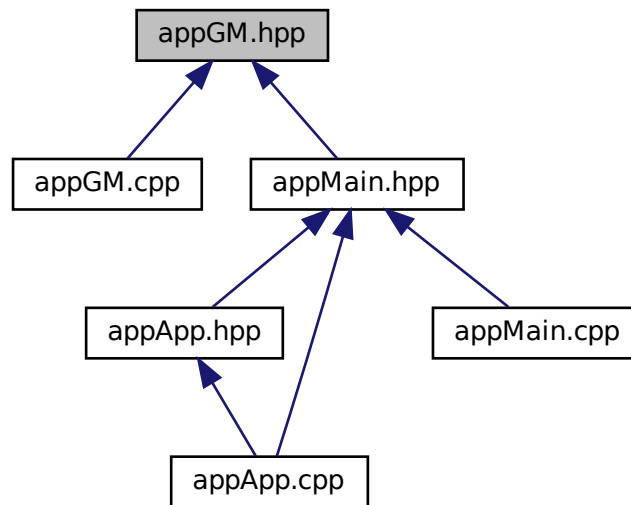
#include <wx/button.h>
#include <wx/listctrl.h>
#include <wx/notebook.h>
#include <wx/textctrl.h>
#include <wx/wx.h>
#include "gameData.hpp"

```

Include dependency graph for appGM.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [appGMPanel](#)

Enumerations

- enum { [ID_ADD_STAT](#) = wxID_HIGHEST + 1 , [ID_ADD_EQUIPMENT](#) , [ID_ADD_SLOT](#) , [ID_ADD_STATE](#) }

6.4.1 Enumeration Type Documentation

6.4.1.1 anonymous enum

anonymous enum

Enumerator

ID_ADD_STAT	
ID_ADD_EQUIPMENT	
ID_ADD_SLOT	
ID_ADD_STATE	

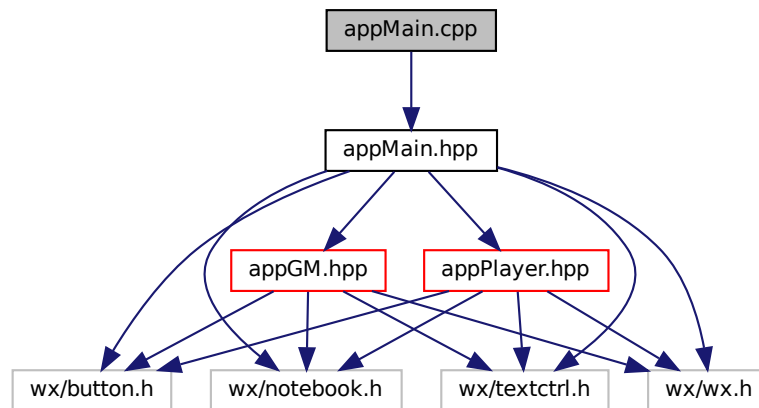
Definition at line 11 of file appGM.hpp.

```
11 { ID_ADD_STAT = wxID_HIGHEST + 1, ID_ADD_EQUIPMENT, ID_ADD_SLOT, ID_ADD_STATE };
```

6.5 appMain.cpp File Reference

```
#include "appMain.hpp"
```

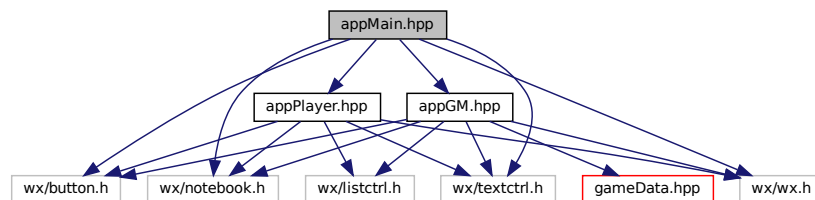
Include dependency graph for appMain.cpp:



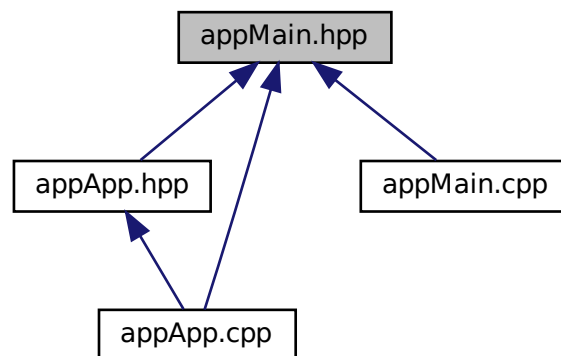
6.6 appMain.hpp File Reference

```
#include "appGM.hpp"
#include "appPlayer.hpp"
#include <wx/button.h>
#include <wx/notebook.h>
#include <wx/textctrl.h>
#include <wx/wx.h>
```

Include dependency graph for appMain.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [appFrame](#)

Enumerations

- enum { [ID_ADD_CHARACTER](#) = 1 }

6.6.1 Enumeration Type Documentation

6.6.1.1 anonymous enum

anonymous enum

Enumerator

ID_ADD_CHARACTER	
------------------	--

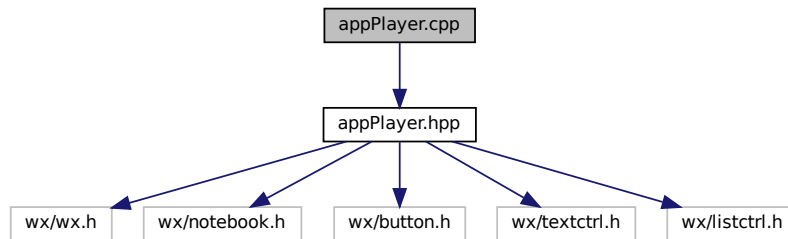
Definition at line 33 of file appMain.hpp.

```
33 { ID\_ADD\_CHARACTER = 1 };
```

6.7 appPlayer.cpp File Reference

```
#include "appPlayer.hpp"
```

Include dependency graph for appPlayer.cpp:



Functions

- [wxBEGIN_EVENT_TABLE](#) (appPlayerPanel, wxPanel) [wxEND_EVENT_TABLE\(\)](#) [appPlayerPanel](#)

6.7.1 Function Documentation

6.7.1.1 wxBEGIN_EVENT_TABLE()

```

wxBEGIN_EVENT_TABLE (
    appPlayerPanel ,
    wxPanel )
  
```

Definition at line 3 of file appPlayer.cpp.

```

12     : wxPanel(parent, wxID_ANY)
13 {
14     playerNotebook = new wxNotebook(this, wxID_ANY);
15
16     statsPanel = new wxPanel(playerNotebook, wxID_ANY);
17     statsSizer = new wxBoxSizer(wxVERTICAL);
18     statsListCtrl = new wxListCtrl(statsPanel, wxID_ANY, wxDefaultPosition, wxDefaultSize, wxLC_REPORT |
19     wxLC_SINGLE_SEL);
20     statsListCtrl->InsertColumn(0, "Name");
21     statsListCtrl->InsertColumn(1, "Description");
22     statsListCtrl->InsertColumn(2, "Value");
23     statsListCtrl->SetColumnWidth(0, 150);
24     statsListCtrl->SetColumnWidth(1, 400);
25     statsListCtrl->SetColumnWidth(2, 50);
26     statsSizer->Add(statsListCtrl, 1, wxEXPAND | wxALL, 5);
27     statsPanel->SetSizer(statsSizer);
28
29     equippedEQPanel = new wxPanel(playerNotebook, wxID_ANY);
30     equippedEQSizer = new wxBoxSizer(wxVERTICAL);
31     equippedEQListCtrl = new wxListCtrl(equippedEQPanel, wxID_ANY, wxDefaultPosition, wxDefaultSize,
32     wxLC_REPORT | wxLC_SINGLE_SEL);
33     equippedEQListCtrl->InsertColumn(0, "Name");
34     equippedEQListCtrl->InsertColumn(1, "Description");
35     equippedEQListCtrl->SetColumnWidth(0, 150);
36     equippedEQListCtrl->SetColumnWidth(1, 400);
37     equippedEQListCtrl->Add(equippedEQListCtrl, 1, wxEXPAND | wxALL, 5);
38
39     // Dodajemy przycisk do panelu equippedEQPanel
40     unequipEquipmentButton = new wxButton(equippedEQPanel, ID_CHAR_UNEQUIP_EQUIPMENT, "Unequip");
41     equippedEQSizer->Add(unequipEquipmentButton, 0, wxALIGN_RIGHT | wxALL, 5);
42
43     equippedEQPanel->SetSizer(equippedEQSizer);
44
45     eqPanel = new wxPanel(playerNotebook, wxID_ANY);
  
```

```

44     eqSizer = new wxBoxSizer(wxVERTICAL);
45     eqListCtrl = new wxListCtrl(eqPanel, wxID_ANY, wxDefaultPosition, wxDefaultSize, wxLC_REPORT |
46                               wxLC_SINGLE_SEL);
47     eqListCtrl->InsertColumn(0, "Name");
48     eqListCtrl->InsertColumn(1, "Description");
49     eqListCtrl->SetColumnWidth(0, 150);
50     eqListCtrl->SetColumnWidth(1, 400);
51     eqSizer->Add(eqListCtrl, 1, wxEXPAND | wxALL, 5);
52
53     // Dodajemy przycisk do panelu eqPanel
54     equipEquipmentButton = new wxButton(eqPanel, ID_CHAR_EQUIP_EQUIPMENT, "Equip");
55     eqSizer->Add(equipEquipmentButton, 0, wxALIGN_RIGHT | wxALL, 5);
56
57     eqPanel->SetSizer(eqSizer);
58
59     statesPanel = new wxPanel(playerNotebook, wxID_ANY);
60     statesSizer = new wxBoxSizer(wxVERTICAL);
61     statesListCtrl = new wxListCtrl(statesPanel, wxID_ANY, wxDefaultPosition, wxDefaultSize, wxLC_REPORT
62                                   | wxLC_SINGLE_SEL);
63     statesListCtrl->InsertColumn(0, "Name");
64     statesListCtrl->InsertColumn(1, "Description");
65     statesListCtrl->SetColumnWidth(0, 150);
66     statesListCtrl->SetColumnWidth(1, 400);
67     statesSizer->Add(statesListCtrl, 1, wxEXPAND | wxALL, 5);
68     statesPanel->SetSizer(statesSizer);
69
70     addPanel = new wxPanel(playerNotebook, wxID_ANY);
71     addPanelSizer = new wxBoxSizer(wxVERTICAL);
72     addStatButton = new wxButton(addPanel, ID_CHAR_ADD_STAT, "Add Stat");
73     addEquipmentButton = new wxButton(addPanel, ID_CHAR_ADD_EQUIPMENT, "Add Equipment");
74     addStateButton = new wxButton(addPanel, ID_CHAR_ADD_STATE, "Add State");
75     addPanelSizer->Add(addStatButton, 0, wxALIGN_LEFT | wxALL, 5);
76     addPanelSizer->Add(addEquipmentButton, 0, wxALIGN_LEFT | wxALL, 5);
77     addPanelSizer->Add(addStateButton, 0, wxALIGN_LEFT | wxALL, 5);
78     addPanel->SetSizer(addPanelSizer);
79
80     playerNotebook->AddPage(statsPanel, "Stats");
81     playerNotebook->AddPage(equippedEQPanel, "Equipped EQ");
82     playerNotebook->AddPage(eqPanel, "EQ");
83     playerNotebook->AddPage(statesPanel, "States");
84     playerNotebook->AddPage(addPanel, "Add");
85
86     mainSizer = new wxBoxSizer(wxVERTICAL);
87     mainSizer->Add(playerNotebook, 1, wxEXPAND | wxALL, 5);
88     SetSizerAndFit(mainSizer);
89 }

```

References ID_CHAR_ADD_EQUIPMENT, ID_CHAR_ADD_STAT, ID_CHAR_ADD_STATE, ID_CHAR_EQUIP_EQUIPMENT, and ID_CHAR_UNEQUIP_EQUIPMENT.

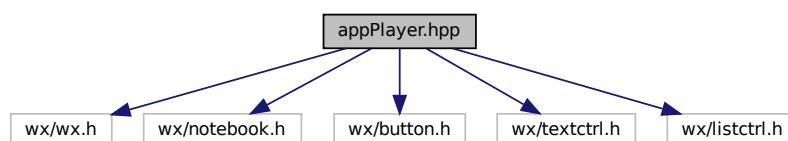
6.8 appPlayer.hpp File Reference

```

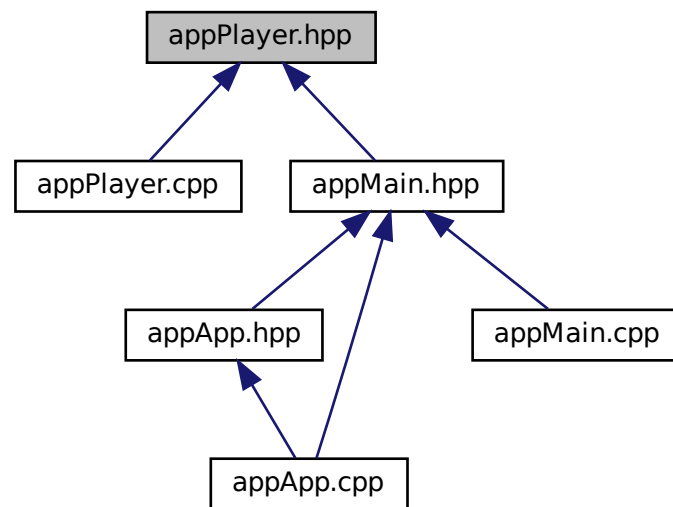
#include <wx/wx.h>
#include <wx/notebook.h>
#include <wx/button.h>
#include <wx/textctrl.h>
#include <wx/listctrl.h>

```

Include dependency graph for appPlayer.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [appPlayerPanel](#)

Enumerations

- enum {
[ID_CHAR_ADD_STAT](#) = wxID_HIGHEST + 1 , [ID_CHAR_ADD_EQUIPMENT](#) , [ID_CHAR_ADD_STATE](#) ,
[ID_CHAR_EQUIP_EQUIPMENT](#) ,
[ID_CHAR_UNEQUIP_EQUIPMENT](#) }

6.8.1 Enumeration Type Documentation

6.8.1.1 anonymous enum

anonymous enum

Enumerator

ID_CHAR_ADD_STAT	
ID_CHAR_ADD_EQUIPMENT	
ID_CHAR_ADD_STATE	
ID_CHAR_EQUIP_EQUIPMENT	
ID_CHAR_UNEQUIP_EQUIPMENT	

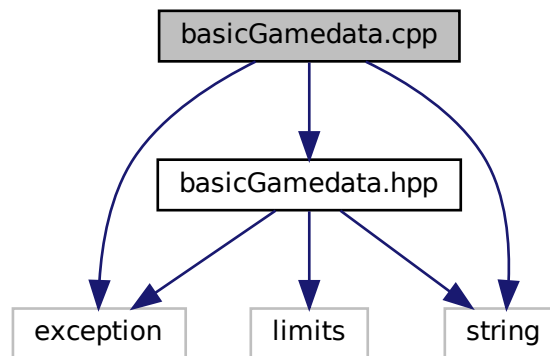
Definition at line 10 of file appPlayer.hpp.

```
11 {  
12     ID_CHAR_ADD_STAT = wxID_HIGHEST + 1,  
13     ID_CHAR_ADD_EQUIPMENT,  
14     ID_CHAR_ADD_STATE,  
15     ID_CHAR_EQUIP_EQUIPMENT,  
16     ID_CHAR_UNEQUIP_EQUIPMENT  
17 };
```

6.9 basicGamedata.cpp File Reference

[BasicGameData](#) implementation.

```
#include "basicGamedata.hpp"  
#include <exception>  
#include <string>  
Include dependency graph for basicGamedata.cpp:
```



6.9.1 Detailed Description

[BasicGameData](#) implementation.

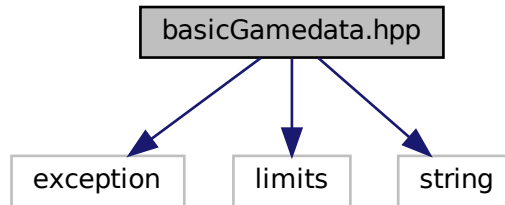
6.10 basicGamedata.hpp File Reference

[BasicGameData](#) interface.

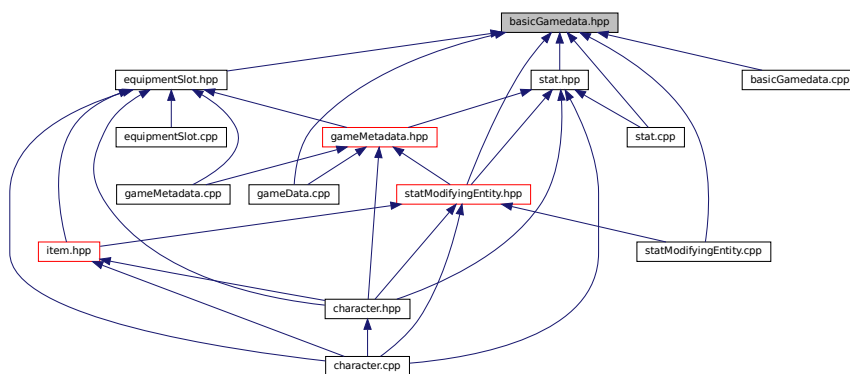
```
#include <exception>  
#include <limits>
```

```
#include <string>
```

Include dependency graph for basicGamedata.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [exceptionIllegalId](#)
Illegal id exception.
- class [BasicGameData](#)
Basic information about game.

6.10.1 Detailed Description

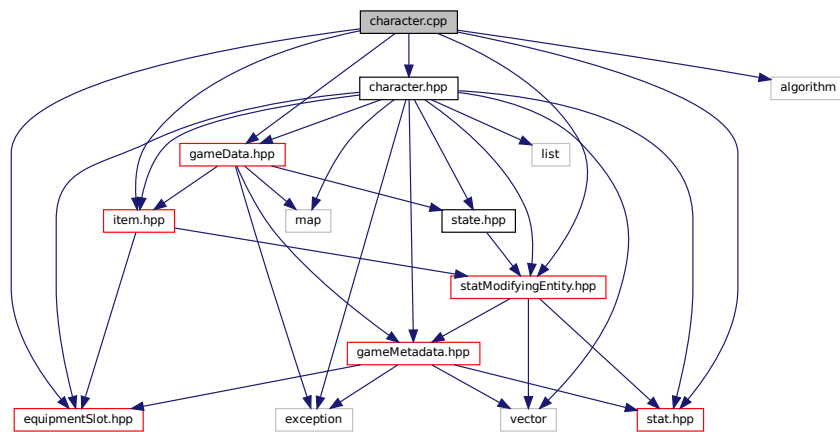
[BasicGameData](#) interface.

6.11 character.cpp File Reference

[Character](#) implementation.

```
#include "character.hpp"
#include "equipmentSlot.hpp"
#include "gameData.hpp"
#include "item.hpp"
#include "stat.hpp"
#include "statModifyingEntity.hpp"
#include <algorithm>
```

Include dependency graph for character.cpp:



6.11.1 Detailed Description

[Character](#) implementation.

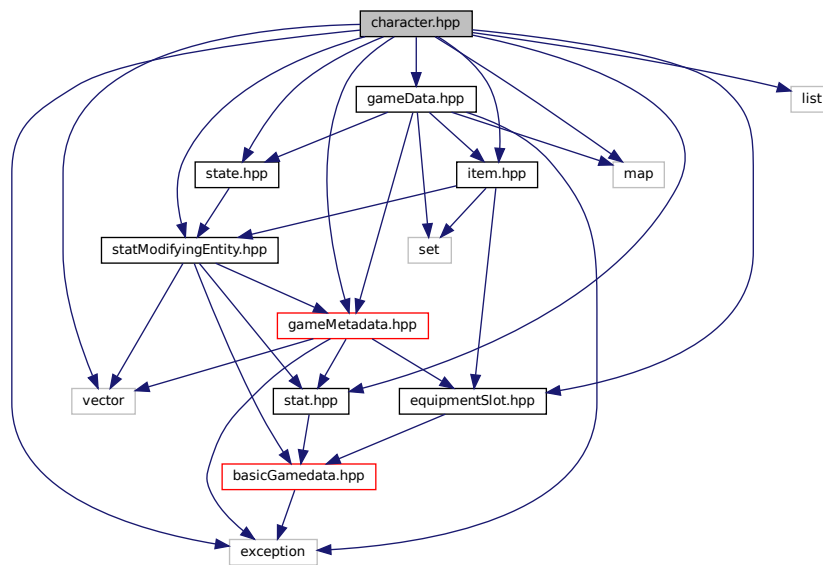
6.12 character.hpp File Reference

[Character](#) interface.

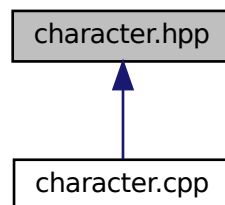
```
#include "equipmentSlot.hpp"
#include "gameData.hpp"
#include "gameMetadata.hpp"
#include "item.hpp"
#include "stat.hpp"
#include "statModifyingEntity.hpp"
#include "state.hpp"
#include <exception>
#include <list>
#include <map>
```

```
#include <vector>
```

Include dependency graph for character.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [exceptionEquipmentSlotOccupied](#)
Tried to perform operation on occupied already slot.
- class [exceptionEquipmentSlotUnused](#)
Tried to extract data from unused slot.
- class [excpetionEquipmentSlotIllegalUsage](#)
Tried to put stuff where it is not suppoed to go.
- class [Character](#)
Represents character.

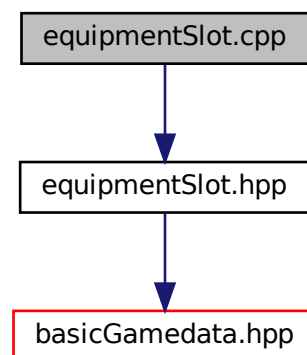
6.12.1 Detailed Description

[Character](#) interface.

6.13 equipmentSlot.cpp File Reference

[EquipmentSlot](#) implementation.

```
#include "equipmentSlot.hpp"  
Include dependency graph for equipmentSlot.cpp:
```



6.13.1 Detailed Description

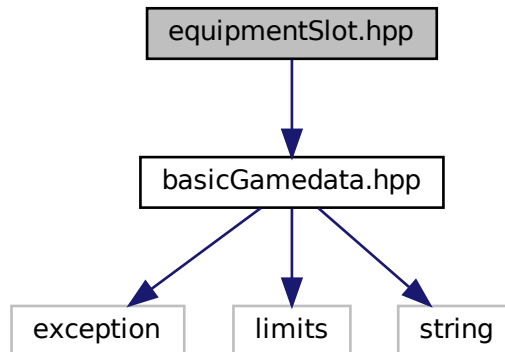
[EquipmentSlot](#) implementation.

6.14 equipmentSlot.hpp File Reference

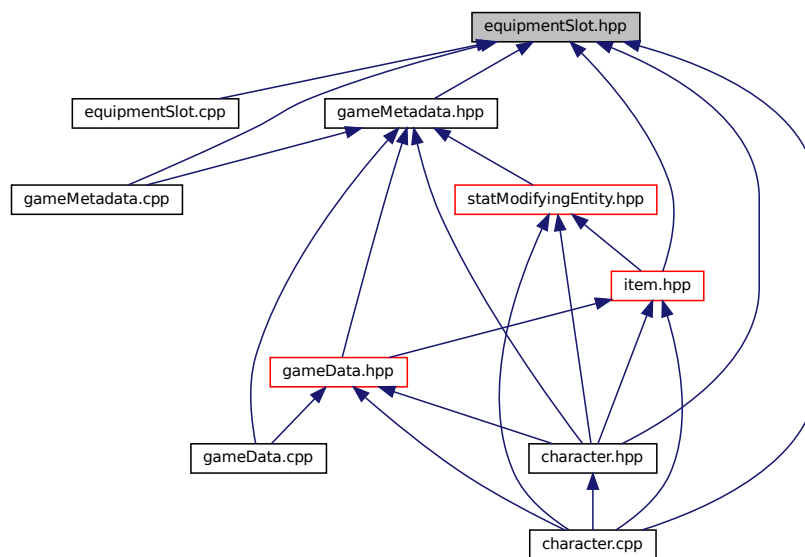
[EquipmentSlot](#) interface.

```
#include "basicGamedata.hpp"
```

Include dependency graph for equipmentSlot.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [EquipmentSlot](#)
Equipment slot representation.

6.14.1 Detailed Description

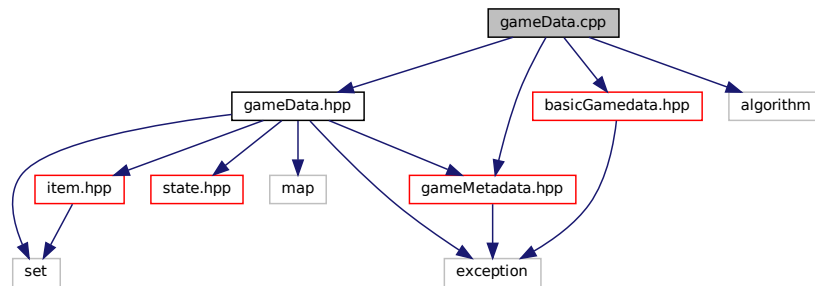
[EquipmentSlot](#) interface.

6.15 gameData.cpp File Reference

[GameData](#) implementation.

```
#include "gameData.hpp"
#include "basicGamedata.hpp"
#include "gameMetadata.hpp"
#include <algorithm>
```

Include dependency graph for gameData.cpp:



6.15.1 Detailed Description

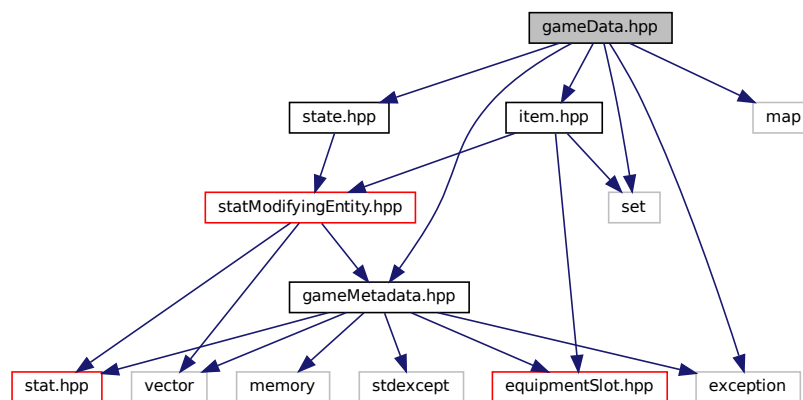
[GameData](#) implementation.

6.16 gameData.hpp File Reference

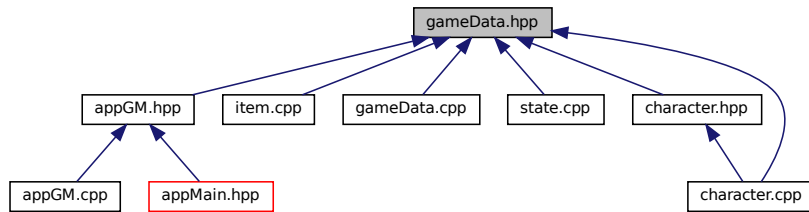
[GameData](#) interface.

```
#include "gameMetadata.hpp"
#include "item.hpp"
#include "state.hpp"
#include <exception>
#include <map>
#include <set>
```

Include dependency graph for gameData.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [excpetionGameDataMismatch](#)
Thrown when attempted to do operation that requires 2 objects to use common [GameData](#) but different were used.
- class [GameData](#)
On top of what [GameMetadata](#) does. Holds items cataloge.

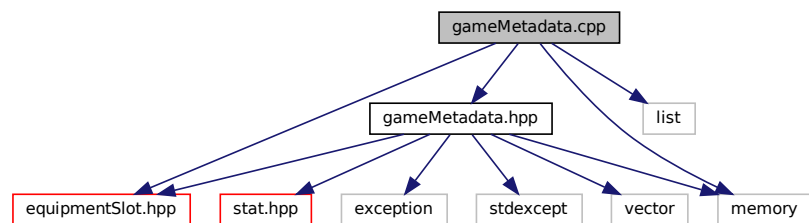
6.16.1 Detailed Description

[GameData](#) interface.

6.17 gameMetadata.cpp File Reference

[GameMetadata](#) implementation.

```
#include "gameMetadata.hpp"
#include "equipmentsSlot.hpp"
#include <list>
#include <memory>
Include dependency graph for gameMetadata.cpp:
```



6.17.1 Detailed Description

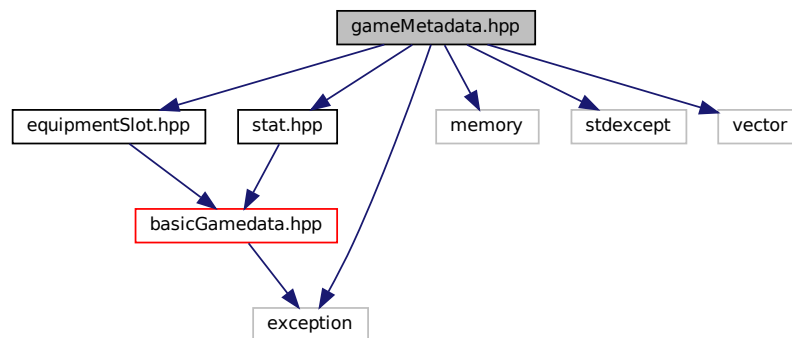
[GameMetadata](#) implementation.

6.18 gameMetadata.hpp File Reference

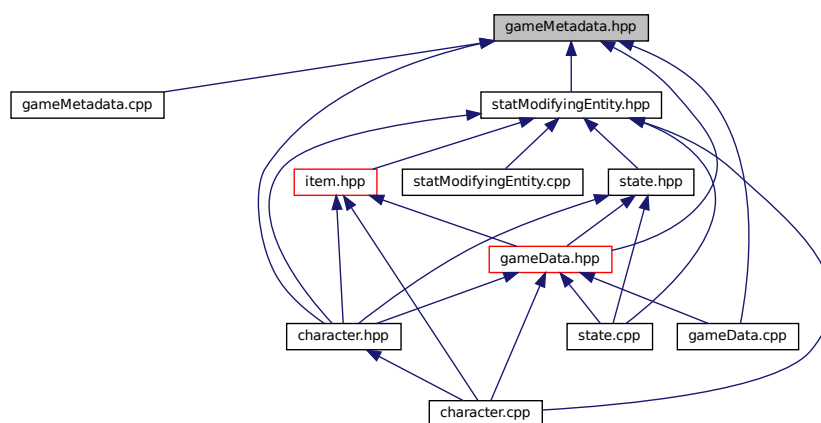
[GameMetadata](#) interface.

```
#include "equipmentSlot.hpp"
#include "stat.hpp"
#include <exception>
#include <memory>
#include <stdexcept>
#include <vector>
```

Include dependency graph for gameMetadata.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [exceptionNonExistingId](#)
Exception.
- class [GameMetadata](#)
Holds game metadata. That is what statistics exist and what equipable slots exist.

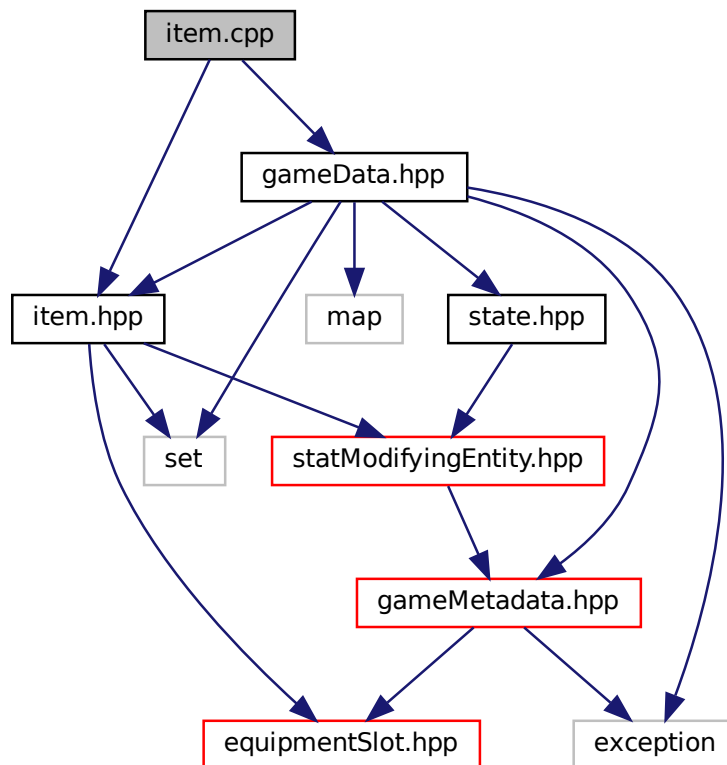
6.18.1 Detailed Description

[GameMetadata](#) interface.

6.19 item.cpp File Reference

[Item](#) implementation.

```
#include "item.hpp"
#include "gameData.hpp"
Include dependency graph for item.cpp:
```



6.19.1 Detailed Description

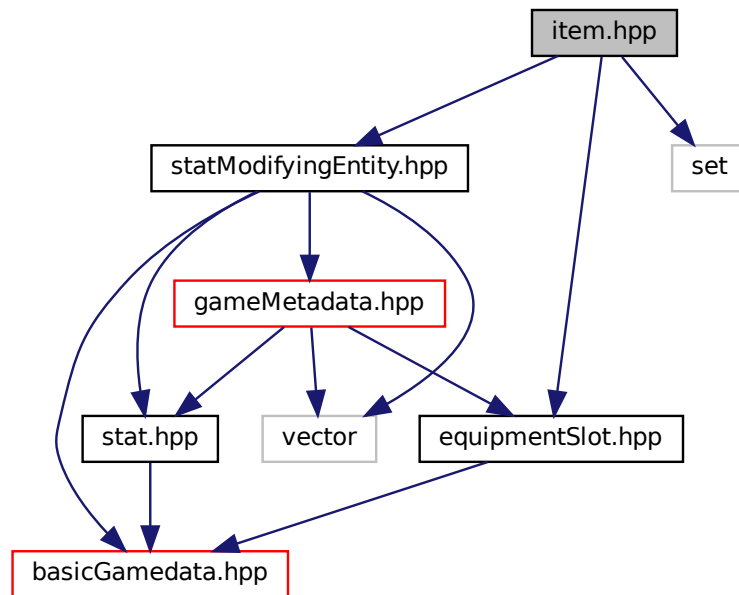
[Item](#) implementation.

6.20 item.hpp File Reference

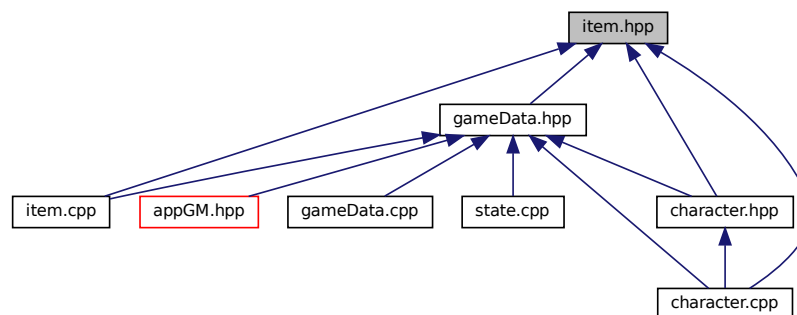
[Item](#) interface.

```
#include "equipmentSlot.hpp"
#include "statModifyingEntity.hpp"
#include <set>
```

Include dependency graph for item.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Item](#)

Represents an item in game.

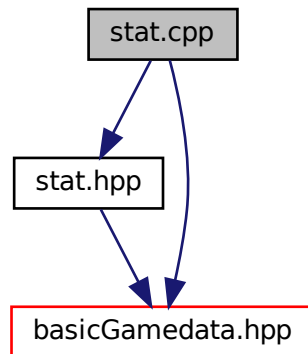
6.20.1 Detailed Description

[Item](#) interface.

6.21 stat.cpp File Reference

[Stat](#) implementation.

```
#include "stat.hpp"  
#include "basicGamedata.hpp"  
Include dependency graph for stat.cpp:
```



6.21.1 Detailed Description

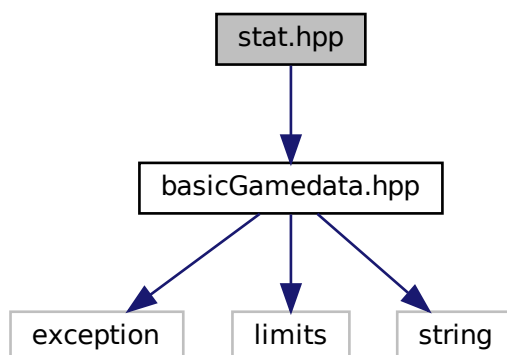
[Stat](#) implementation.

6.22 stat.hpp File Reference

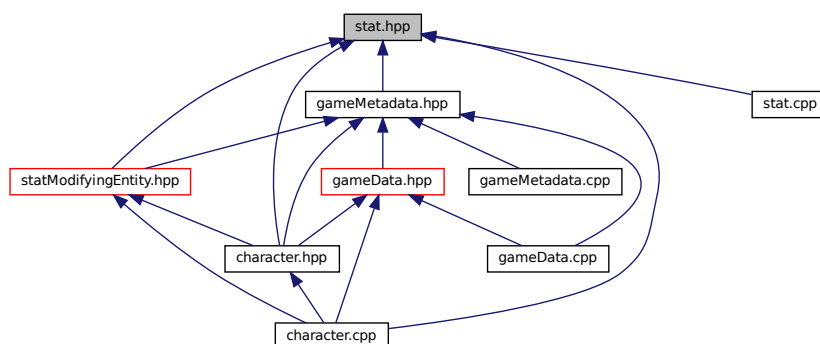
[Stat](#) interface.

```
#include "basicGamedata.hpp"
```

Include dependency graph for stat.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Stat](#)
Statistics.

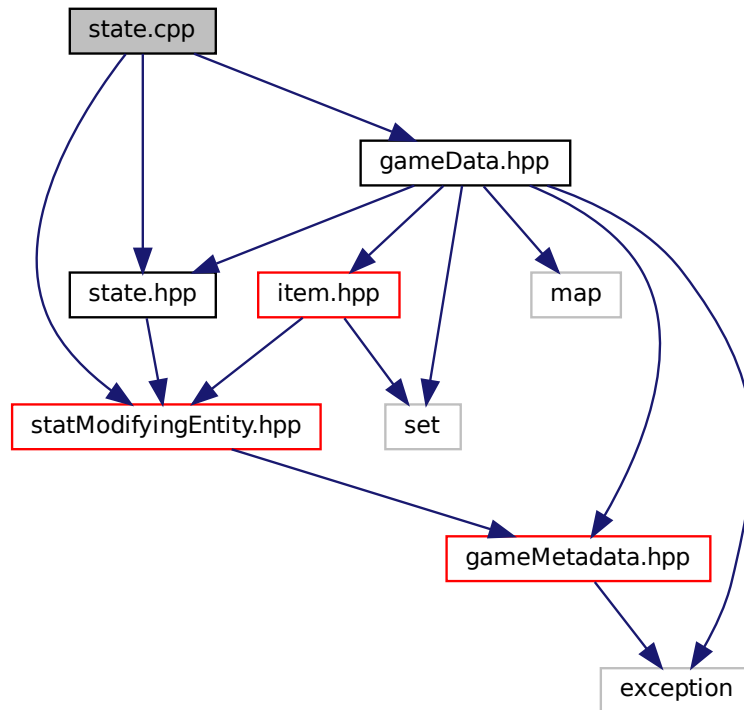
6.22.1 Detailed Description

[Stat](#) interface.

6.23 state.cpp File Reference

[State](#) implementation.

```
#include "state.hpp"
#include "gameData.hpp"
#include "statModifyingEntity.hpp"
Include dependency graph for state.cpp:
```



6.23.1 Detailed Description

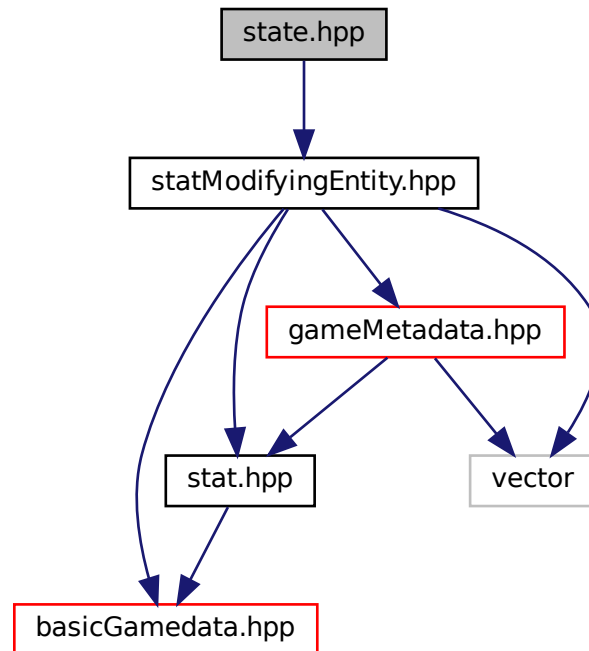
[State](#) implementation.

6.24 state.hpp File Reference

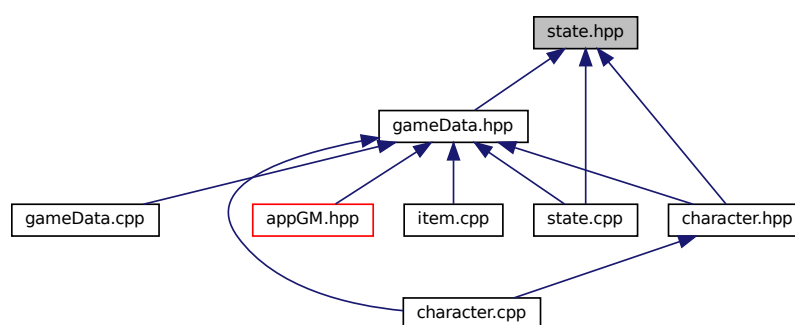
[State](#) interface.

```
#include "statModifyingEntity.hpp"
```

Include dependency graph for state.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [State](#)
Represents [State](#) in game.

6.24.1 Detailed Description

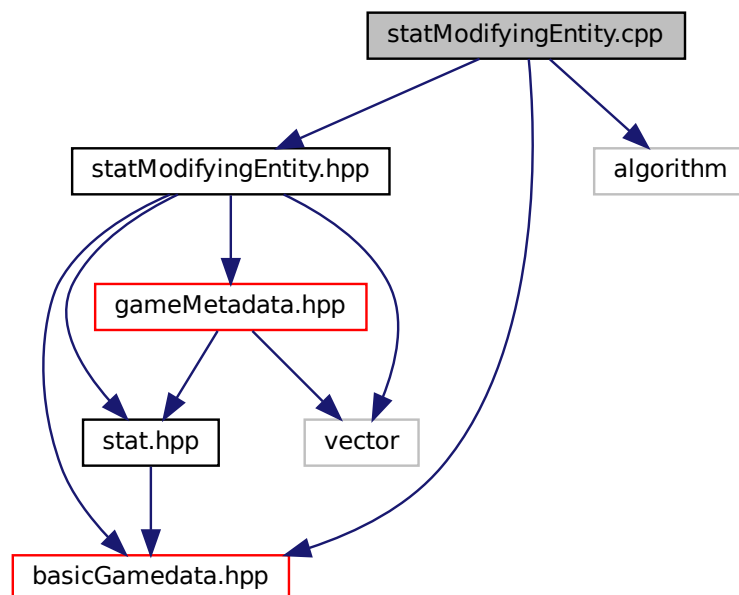
[State](#) interface.

6.25 statModifyingEntity.cpp File Reference

[StatModifyingEntity](#) implementation.

```
#include "statModifyingEntity.hpp"
#include "basicGamedata.hpp"
#include <algorithm>
```

Include dependency graph for statModifyingEntity.cpp:



6.25.1 Detailed Description

[StatModifyingEntity](#) implementation.

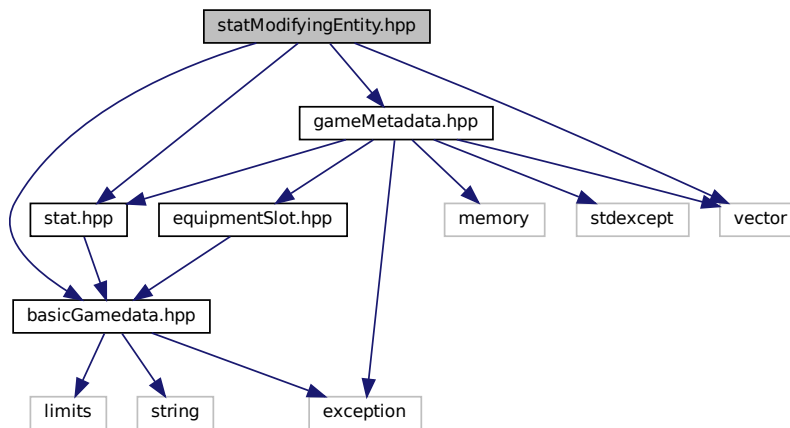
6.26 statModifyingEntity.hpp File Reference

[StatModifyingEntity](#) interface.

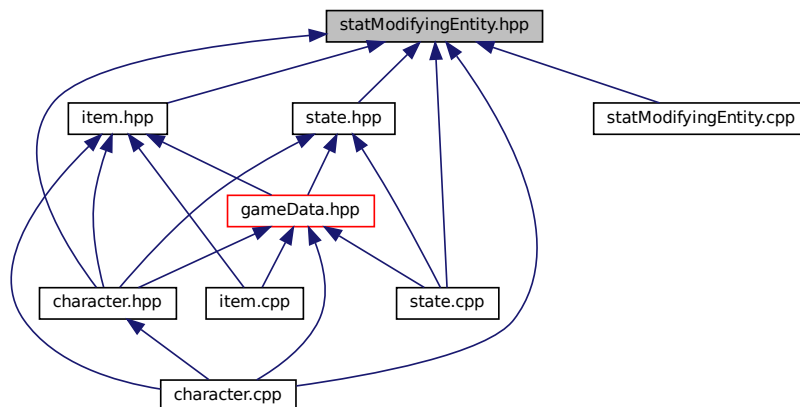
```
#include "basicGamedata.hpp"
#include "gameMetadata.hpp"
#include "stat.hpp"
```

```
#include <vector>
```

Include dependency graph for statModifyingEntity.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [exceptionInvalidGameMetadata](#)
Invalid game data.
- class [StatModifyingEntity](#)
Represents collection of stat modifiers.

6.26.1 Detailed Description

[StatModifyingEntity](#) interface.

Index

- ~GameData
 - GameData, [80](#)
- ~GameMetadata
 - GameMetadata, [95](#)
- addCharPanel
 - appFrame, [13](#)
- addElementPanel
 - appGMPanel, [21](#)
- addElementSizer
 - appGMPanel, [21](#)
- addEquipmentButton
 - appGMPanel, [21](#)
 - appPlayerPanel, [28](#)
- addEquipmentSlot
 - GameData, [80](#)
 - GameMetadata, [95](#)
- addItem
 - Character, [43](#)
 - GameData, [81](#)
- addModifier
 - Item, [106](#)
 - State, [130](#)
 - StatModifyingEntity, [142](#)
- addPanel
 - appPlayerPanel, [28](#)
- addPanelSizer
 - appPlayerPanel, [28](#)
- addSlotButton
 - appGMPanel, [21](#)
- addStat
 - GameData, [82](#)
 - GameMetadata, [96](#)
- addStatButton
 - appGMPanel, [21](#)
 - appPlayerPanel, [28](#)
- addState
 - Character, [44](#)
 - GameData, [83](#)
- addStateButton
 - appGMPanel, [21](#)
 - appPlayerPanel, [28](#)
- appApp, [9](#)
 - OnInit, [10](#)
- appApp.cpp, [151](#)
 - wxIMPLEMENT_APP, [151](#)
- appApp.hpp, [152](#)
- appFrame, [10](#)
 - addCharPanel, [13](#)
 - appFrame, [11](#)
 - charNameTextCtrl, [13](#)
 - gmPanel, [13](#)
 - notebook, [13](#)
 - OnAddCharacter, [12](#)
- appGM.cpp, [153](#)
 - wxBEGIN_EVENT_TABLE, [153](#)
- appGM.hpp, [154](#)
 - ID_ADD_EQUIPMENT, [155](#)
 - ID_ADD_SLOT, [155](#)
 - ID_ADD_STAT, [155](#)
 - ID_ADD_STATE, [155](#)
- appGMPanel, [14](#)
 - addElementPanel, [21](#)
 - addElementSizer, [21](#)
 - addEquipmentButton, [21](#)
 - addSlotButton, [21](#)
 - addStatButton, [21](#)
 - addStateButton, [21](#)
 - appGMPanel, [15](#)
 - eqListCtrl, [22](#)
 - eqPanel, [22](#)
 - eqSizer, [22](#)
 - gmNotebook, [22](#)
 - m_gamedata, [22](#)
 - OnAddEquipment, [16](#)
 - OnAddSlot, [16](#)
 - OnAddStat, [17](#)
 - OnAddState, [17](#)
 - slotListCtrl, [22](#)
 - slotPanel, [23](#)
 - slotSizer, [23](#)
 - stateListCtrl, [23](#)
 - statePanel, [23](#)
 - stateSizer, [23](#)
 - statsListCtrl, [23](#)
 - statsPanel, [24](#)
 - statsSizer, [24](#)
 - UpdateEqListCtrl, [18](#)
 - UpdateSlotListCtrl, [18](#)
 - UpdateStateListCtrl, [19](#)
 - UpdateStatsListCtrl, [20](#)
 - wxDECLARE_EVENT_TABLE, [20](#)
- appMain.cpp, [156](#)
- appMain.hpp, [156](#)
 - ID_ADD_CHARACTER, [157](#)
- appPlayer.cpp, [157](#)
 - wxBEGIN_EVENT_TABLE, [158](#)
- appPlayer.hpp, [159](#)
 - ID_CHAR_ADD_EQUIPMENT, [160](#)

- ID_CHAR_ADD_STAT, 160
- ID_CHAR_ADD_STATE, 160
- ID_CHAR_EQUIP_EQUIPMENT, 160
- ID_CHAR_UNEQUIP_EQUIPMENT, 160
- appPlayerPanel, 24
 - addEquipmentButton, 28
 - addPanel, 28
 - addPanelSizer, 28
 - addStatButton, 28
 - addStateButton, 28
 - appPlayerPanel, 26
 - eqListCtrl, 29
 - eqPanel, 29
 - eqSizer, 29
 - equipEquipmentButton, 29
 - equippedEQListCtrl, 29
 - equippedEQPanel, 29
 - equippedEQSizer, 30
 - mainSizer, 30
 - OnAddEquipment, 26
 - OnAddStat, 26
 - OnAddState, 26
 - OnEquipEquipment, 26
 - OnUnequipEquipment, 27
 - playerNotebook, 30
 - statesListCtrl, 30
 - statesPanel, 30
 - statesSizer, 30
 - statsListCtrl, 31
 - statsPanel, 31
 - statsSizer, 31
 - unequipEquipmentButton, 31
 - UpdateEQListCtrl, 27
 - UpdateEquippedEQListCtrl, 27
 - UpdateStatesListCtrl, 27
 - UpdateStatsListCtrl, 27
 - wxDECLARE_EVENT_TABLE, 28
- BasicGameData, 32
 - BasicGameData, 33
 - GameData, 38
 - GameMetadata, 38
 - getDescription, 34
 - getId, 34
 - getName, 35
 - id_t, 33
 - INVALID_ID, 38
 - isId, 35
 - m_description, 39
 - m_id, 39
 - m_name, 39
 - setDescription, 36
 - setId, 36
 - setName, 37
 - validateIntegrity, 37
- basicGamedata.cpp, 161
- basicGamedata.hpp, 161
- Character, 40
 - addItem, 43
 - addState, 44
 - Character, 43
 - equiplItem, 45
 - equipment_t, 42
 - getBaseStatValue, 46
 - getEquippedItem, 47, 48
 - getEquipment, 49
 - getGameData, 49
 - getInventory, 50
 - getStates, 50
 - getStatValue, 51
 - getStatValueContributors, 52
 - getStatValueEquipmentContributors, 52
 - getStatValueStatesContributors, 53
 - inventory_t, 42
 - isEquipmentSlotUsed, 54
 - itemQuantity_t, 42
 - m_baseStatValues, 57
 - m_equipment, 57
 - m_gameData, 57
 - m_inventory, 58
 - m_states, 58
 - setBaseStatValue, 55
 - states_t, 42
 - statValueContributors_t, 42
 - statValues_t, 43
 - validateDataIntegrity, 56
- character.cpp, 163
- character.hpp, 163
- charNameTextCtrl
 - appFrame, 13
- eqListCtrl
 - appGMPanel, 22
 - appPlayerPanel, 29
- eqPanel
 - appGMPanel, 22
 - appPlayerPanel, 29
- eqSizer
 - appGMPanel, 22
 - appPlayerPanel, 29
- equipableSlots_t
 - Item, 104
- equipEquipmentButton
 - appPlayerPanel, 29
- equiplItem
 - Character, 45
- equipment_t
 - Character, 42
- EquipmentSlot, 59
 - EquipmentSlot, 61
 - getDescription, 61
 - getId, 61
 - getName, 62
 - id_t, 60
 - INVALID_ID, 65
 - isId, 62
 - m_description, 65

- m_id, 66
 - m_name, 66
 - setDescription, 63
 - setId, 63
 - setName, 64
 - validateIntegrity, 65
- equipmentSlot.cpp, 165
- equipmentSlot.hpp, 165
- equippedEQListCtrl
 - appPlayerPanel, 29
- equippedEQPanel
 - appPlayerPanel, 29
- equippedEQSizer
 - appPlayerPanel, 30
- equipmentSlotsCollection_t
 - GameData, 79
 - GameMetadata, 94
- exceptionEquipmentSlotOccupied, 67
 - what, 68
- exceptionEquipmentSlotUnused, 68
 - what, 69
- exceptionIllegalId, 70
 - what, 71
- exceptionInvalidGameMetadata, 71
 - what, 72
- exceptionNonExistingId, 73
 - what, 74
- exceptionEquipmentSlotIllegalUsage, 74
 - what, 75
- exceptionGameDataMismatch, 76
 - what, 77
- GameData, 77
 - ~GameData, 80
 - addEquipmentSlot, 80
 - addItem, 81
 - addStat, 82
 - addState, 83
 - BasicGameData, 38
 - equipmentSlotsCollection_t, 79
 - getEquipmentSlot, 84
 - getEquipmentSlots, 85
 - getItem, 86
 - getItems, 87
 - getStat, 87
 - getState, 88
 - getStates, 89
 - getStats, 89
 - Item, 91
 - itemcollection_t, 79
 - m_equipmentSlots, 91
 - m_items, 91
 - m_nextEquipmentSlotId, 92
 - m_nextItemId, 92
 - m_nextStatId, 92
 - m_nextStatId, 92
 - m_states, 93
 - m_stats, 93
 - stateCollection_t, 80
 - statsCollection_t, 80
 - validateDataIntegrity, 90
- gameData.cpp, 167
- gameData.hpp, 167
- GameMetadata, 93
 - ~GameMetadata, 95
 - addEquipmentSlot, 95
 - addStat, 96
 - BasicGameData, 38
 - equipmentSlotsCollection_t, 94
 - getEquipmentSlot, 97
 - getEquipmentSlots, 98
 - getStat, 99
 - getStats, 100
 - m_equipmentSlots, 100
 - m_nextEquipmentSlotId, 101
 - m_nextStatId, 101
 - m_stats, 101
 - statsCollection_t, 95
- gameMetadata.cpp, 168
- gameMetadata.hpp, 169
- getBaseStatValue
 - Character, 46
- getDescription
 - BasicGameData, 34
 - EquipmentSlot, 61
 - Item, 107
 - Stat, 120
 - State, 131
 - StatModifyingEntity, 143
- getEquipableSlots
 - Item, 107
- getEquippedItem
 - Character, 47, 48
- getEquipment
 - Character, 49
- getEquipmentSlot
 - GameData, 84
 - GameMetadata, 97
- getEquipmentSlots
 - GameData, 85
 - GameMetadata, 98
- getGameData
 - Character, 49
- getGameMetadata
 - Item, 107
 - State, 131
 - StatModifyingEntity, 143
- getId
 - BasicGameData, 34
 - EquipmentSlot, 61
 - Item, 108
 - Stat, 121
 - State, 132
 - StatModifyingEntity, 144
- getInventory
 - Character, 50
- getItem

- GameData, 86
- getItems
 - GameData, 87
- getModifiers
 - Item, 109
 - State, 133
 - StatModifyingEntity, 145
- getModifierValue
 - Item, 109
 - State, 133
 - StatModifyingEntity, 145
- getName
 - BasicGameData, 35
 - EquipmentSlot, 62
 - Item, 110
 - Stat, 121
 - State, 134
 - StatModifyingEntity, 146
- getStat
 - GameData, 87
 - GameMetadata, 99
- getState
 - GameData, 88
- getStates
 - Character, 50
 - GameData, 89
- getStats
 - GameData, 89
 - GameMetadata, 100
- getStatValue
 - Character, 51
- getStatValueContributors
 - Character, 52
- getStatValueEquipmentContributors
 - Character, 52
- getStatValueStatesContributors
 - Character, 53
- gmNotebook
 - appGMPanel, 22
- gmPanel
 - appFrame, 13
- ID_ADD_CHARACTER
 - appMain.hpp, 157
- ID_ADD_EQUIPMENT
 - appGM.hpp, 155
- ID_ADD_SLOT
 - appGM.hpp, 155
- ID_ADD_STAT
 - appGM.hpp, 155
- ID_ADD_STATE
 - appGM.hpp, 155
- ID_CHAR_ADD_EQUIPMENT
 - appPlayer.hpp, 160
- ID_CHAR_ADD_STAT
 - appPlayer.hpp, 160
- ID_CHAR_ADD_STATE
 - appPlayer.hpp, 160
- ID_CHAR_EQUIP_EQUIPMENT
 - appPlayer.hpp, 160
- ID_CHAR_UNEQUIP_EQUIPMENT
 - appPlayer.hpp, 160
- id_t
 - BasicGameData, 33
 - EquipmentSlot, 60
 - Item, 104
 - Stat, 119
 - State, 129
 - StatModifyingEntity, 141
- INVALID_ID
 - BasicGameData, 38
 - EquipmentSlot, 65
 - Item, 116
 - Stat, 125
 - State, 137
 - StatModifyingEntity, 149
- inventory_t
 - Character, 42
- isEquipableOn
 - Item, 110
- isEquipmentSlotUsed
 - Character, 54
- isId
 - BasicGameData, 35
 - EquipmentSlot, 62
 - Item, 111
 - Stat, 121
 - State, 134
 - StatModifyingEntity, 146
- Item, 102
 - addModifier, 106
 - equipableSlots_t, 104
 - GameData, 91
 - getDescription, 107
 - getEquipableSlots, 107
 - getGameMetadata, 107
 - getId, 108
 - getModifiers, 109
 - getModifierValue, 109
 - getName, 110
 - id_t, 104
 - INVALID_ID, 116
 - isEquipableOn, 110
 - isId, 111
 - Item, 105
 - m_description, 116
 - m_equipableOn, 116
 - m_gameMetadata, 116
 - m_id, 116
 - m_modifiers, 117
 - m_name, 117
 - modifiersCollection_t, 104
 - modifier_t, 105
 - setDescription, 112
 - setEquipableOn, 112
 - setId, 113
 - setName, 114

- unsetEquipableOn, 114
 - validateIntegrity, 115
- item.cpp, 170
- item.hpp, 171
- itemcollection_t
 - GameData, 79
- itemQuantity_t
 - Character, 42
- m_baseStatValues
 - Character, 57
- m_description
 - BasicGameData, 39
 - EquipmentSlot, 65
 - Item, 116
 - Stat, 126
 - State, 137
 - StatModifyingEntity, 149
- m_equipableOn
 - Item, 116
- m_equipment
 - Character, 57
- m_equipmentSlots
 - GameData, 91
 - GameMetadata, 100
- m_gameData
 - Character, 57
- m_gamedata
 - appGMPanel, 22
- m_gameMetadata
 - Item, 116
 - State, 137
 - StatModifyingEntity, 149
- m_id
 - BasicGameData, 39
 - EquipmentSlot, 66
 - Item, 116
 - Stat, 126
 - State, 138
 - StatModifyingEntity, 150
- m_inventory
 - Character, 58
- m_items
 - GameData, 91
- m_modifiers
 - Item, 117
 - State, 138
 - StatModifyingEntity, 150
- m_name
 - BasicGameData, 39
 - EquipmentSlot, 66
 - Item, 117
 - Stat, 126
 - State, 138
 - StatModifyingEntity, 150
- m_nextEquipmentSlotId
 - GameData, 92
 - GameMetadata, 101
- m_nextItemId
 - GameData, 92
- m_nextStatId
 - GameData, 92
 - GameMetadata, 101
- m_states
 - Character, 58
 - GameData, 93
- m_stats
 - GameData, 93
 - GameMetadata, 101
- mainSizer
 - appPlayerPanel, 30
- modifiersCollection_t
 - Item, 104
 - State, 129
 - StatModifyingEntity, 141
- modifier_t
 - Item, 105
 - State, 129
 - StatModifyingEntity, 141
- notebook
 - appFrame, 13
- OnAddCharacter
 - appFrame, 12
- OnAddEquipment
 - appGMPanel, 16
 - appPlayerPanel, 26
- OnAddSlot
 - appGMPanel, 16
- OnAddStat
 - appGMPanel, 17
 - appPlayerPanel, 26
- OnAddState
 - appGMPanel, 17
 - appPlayerPanel, 26
- OnEquipEquipment
 - appPlayerPanel, 26
- OnInit
 - appApp, 10
- OnUnequipEquipment
 - appPlayerPanel, 27
- playerNotebook
 - appPlayerPanel, 30
- setBaseStatValue
 - Character, 55
- setDescription
 - BasicGameData, 36
 - EquipmentSlot, 63
 - Item, 112
 - Stat, 122
 - State, 135
 - StatModifyingEntity, 147
- setEquipableOn

- Item, 112
- setId
 - BasicGameData, 36
 - EquipmentSlot, 63
 - Item, 113
 - Stat, 123
 - State, 135
 - StatModifyingEntity, 147
- setName
 - BasicGameData, 37
 - EquipmentSlot, 64
 - Item, 114
 - Stat, 123
 - State, 136
 - StatModifyingEntity, 148
- slotListCtrl
 - appGMPanel, 22
- slotPanel
 - appGMPanel, 23
- slotSizer
 - appGMPanel, 23
- Stat, 118
 - getDescription, 120
 - getId, 121
 - getName, 121
 - id_t, 119
 - INVALID_ID, 125
 - isId, 121
 - m_description, 126
 - m_id, 126
 - m_name, 126
 - setDescription, 122
 - setId, 123
 - setName, 123
 - Stat, 120
 - validateIntegrity, 125
 - value_t, 120
- stat.cpp, 172
- stat.hpp, 172
- State, 127
 - addModifier, 130
 - getDescription, 131
 - getGameMetadata, 131
 - getId, 132
 - getModifiers, 133
 - getModifierValue, 133
 - getName, 134
 - id_t, 129
 - INVALID_ID, 137
 - isId, 134
 - m_description, 137
 - m_gameMetadata, 137
 - m_id, 138
 - m_modifiers, 138
 - m_name, 138
 - modifiersCollection_t, 129
 - modifier_t, 129
 - setDescription, 135
 - setId, 135
 - setName, 136
 - State, 129
 - validateIntegrity, 136
- state.cpp, 174
- state.hpp, 174
- stateCollction_t
 - GameData, 80
- stateListCtrl
 - appGMPanel, 23
- statePanel
 - appGMPanel, 23
- states_t
 - Character, 42
- stateSizer
 - appGMPanel, 23
- statesListCtrl
 - appPlayerPanel, 30
- statesPanel
 - appPlayerPanel, 30
- statesSizer
 - appPlayerPanel, 30
- StatModifyingEntity, 139
 - addModifier, 142
 - getDescription, 143
 - getGameMetadata, 143
 - getId, 144
 - getModifiers, 145
 - getModifierValue, 145
 - getName, 146
 - id_t, 141
 - INVALID_ID, 149
 - isId, 146
 - m_description, 149
 - m_gameMetadata, 149
 - m_id, 150
 - m_modifiers, 150
 - m_name, 150
 - modifiersCollection_t, 141
 - modifier_t, 141
 - setDescription, 147
 - setId, 147
 - setName, 148
 - StatModifyingEntity, 141
 - validateIntegrity, 148
- statModifyingEntity.cpp, 176
- statModifyingEntity.hpp, 176
- statsCollection_t
 - GameData, 80
 - GameMetadata, 95
- statsListCtrl
 - appGMPanel, 23
 - appPlayerPanel, 31
- statsPanel
 - appGMPanel, 24
 - appPlayerPanel, 31
- statsSizer
 - appGMPanel, 24

- appPlayerPanel, [31](#)
- statValueContributors_t
 - Character, [42](#)
- statValues_t
 - Character, [43](#)
- unequipEquipmentButton
 - appPlayerPanel, [31](#)
- unsetEquipableOn
 - Item, [114](#)
- UpdateEQListCtrl
 - appPlayerPanel, [27](#)
- UpdateEqListCtrl
 - appGMPanel, [18](#)
- UpdateEquippedEQListCtrl
 - appPlayerPanel, [27](#)
- UpdateSlotListCtrl
 - appGMPanel, [18](#)
- UpdateStateListCtrl
 - appGMPanel, [19](#)
- UpdateStatesListCtrl
 - appPlayerPanel, [27](#)
- UpdateStatsListCtrl
 - appGMPanel, [20](#)
 - appPlayerPanel, [27](#)
- validateDataIntegrity
 - Character, [56](#)
 - GameData, [90](#)
- validateIntegrity
 - BasicGameData, [37](#)
 - EquipmentSlot, [65](#)
 - Item, [115](#)
 - Stat, [125](#)
 - State, [136](#)
 - StatModifyingEntity, [148](#)
- value_t
 - Stat, [120](#)
- what
 - exceptionEquipmentSlotOccupied, [68](#)
 - exceptionEquipmentSlotUnused, [69](#)
 - exceptionIllegalId, [71](#)
 - exceptionInvalidGameMetadata, [72](#)
 - exceptionNonExistingId, [74](#)
 - exceptionEquipmentSlotIllegalUsage, [75](#)
 - exceptionGameDataMismatch, [77](#)
- wxBEGIN_EVENT_TABLE
 - appGM.cpp, [153](#)
 - appPlayer.cpp, [158](#)
- wxDECLARE_EVENT_TABLE
 - appGMPanel, [20](#)
 - appPlayerPanel, [28](#)
- wxIMPLEMENT_APP
 - appApp.cpp, [151](#)