AntFarm

v0.1a

Generated by Doxygen 1.8.7

Mon May 26 2014 17:40:44

Contents

1 Main Page **1**

2 AntFarm **3**

3 Todo List **5**

4 Namespace Index **7**

4.1 Namespace List 7

5 Hierarchical Index **9**

5.1 Class Hierarchy 9

6 Class Index **11**

6.1 Class List 11

7 File Index **13**

7.1 File List 13

8 Namespace Documentation **15**

8.1 ant Namespace Reference 15

8.2 ant::AppStates Namespace Reference 17

8.2.1 Detailed Description 17

8.2.2 Enumeration Type Documentation 17

8.2.2.1 ID 17

8.3 ant::GUI Namespace Reference 17

8.4 ant::Utils Namespace Reference 17

8.4.1 Function Documentation 18

8.4.1.1 centerOrigin 18

8.4.1.2 centerOrigin 19

8.4.1.3 dot 19

8.4.1.4 getAngle 19

8.4.1.5 getDirection 19

8.4.1.6 makeSharedPtr 20

8.4.1.7 makeUniquePtr 20

8.4.1.8 normalize . . . 21
8.4.1.9 RectContains . . . 21
8.4.1.10 rotateRect . . . 21
8.4.1.11 toDegrees . . . 22
8.4.1.12 toRadians . . . 22
8.4.1.13 toString . . . 22
8.4.2 Variable Documentation . . . 23
8.4.2.1 PI . . . 23

9 Class Documentation **25**
9.1 ant::Application Class Reference . . . 25
9.1.1 Detailed Description . . . 27
9.1.2 Constructor & Destructor Documentation . . . 27
9.1.2.1 Application . . . 27
9.1.2.2 ∼Application . . . 28
9.1.3 Member Function Documentation . . . 28
9.1.3.1 loadConfig . . . 28
9.1.3.2 processInput . . . 28
9.1.3.3 registerStates . . . 29
9.1.3.4 render . . . 29
9.1.3.5 run . . . 30
9.1.3.6 update . . . 31
9.1.3.7 updateStatistics . . . 32
9.1.4 Member Data Documentation . . . 33
9.1.4.1 font . . . 33
9.1.4.2 mAssets . . . 33
9.1.4.3 mStateStack . . . 33
9.1.4.4 mStatisticsNumFrames . . . 33
9.1.4.5 mStatisticsText . . . 33
9.1.4.6 mStatisticsUpdateTime . . . 33
9.1.4.7 mWindow . . . 33
9.1.4.8 TimePerFrame . . . 33
9.1.4.9 version . . . 33
9.2 ant::AppState Class Reference . . . 33
9.2.1 Detailed Description . . . 36
9.2.2 Member Typedef Documentation . . . 37
9.2.2.1 sPtr . . . 37
9.2.2.2 uPtr . . . 37
9.2.3 Constructor & Destructor Documentation . . . 37
9.2.3.1 AppState . . . 37

9.2.3.2 ~AppState . . . 37
9.2.4 Member Function Documentation . . . 37
9.2.4.1 getContext . . . 37
9.2.4.2 handleEvent . . . 38
9.2.4.3 render . . . 38
9.2.4.4 requestStackPop . . . 38
9.2.4.5 requestStackPush . . . 39
9.2.4.6 requestStateClear . . . 40
9.2.4.7 update . . . 40
9.2.5 Member Data Documentation . . . 41
9.2.5.1 mContext . . . 41
9.2.5.2 mStack . . . 41
9.3 ant::AssetManager Class Reference . . . 41
9.3.1 Detailed Description . . . 42
9.3.2 Constructor & Destructor Documentation . . . 43
9.3.2.1 AssetManager . . . 43
9.3.2.2 ~AssetManager . . . 43
9.3.3 Member Function Documentation . . . 43
9.3.3.1 addFont . . . 43
9.3.3.2 addImage . . . 43
9.3.3.3 addTexture . . . 44
9.3.3.4 clear . . . 44
9.3.3.5 getFont . . . 44
9.3.3.6 getImage . . . 45
9.3.3.7 getTexture . . . 45
9.3.3.8 hasFont . . . 45
9.3.3.9 hasImage . . . 45
9.3.3.10 hasTexture . . . 45
9.3.3.11 loadAssets . . . 46
9.3.3.12 removeFont . . . 46
9.3.3.13 removeImage . . . 47
9.3.3.14 removeTexture . . . 48
9.3.4 Member Data Documentation . . . 48
9.3.4.1 fonts . . . 48
9.3.4.2 images . . . 48
9.3.4.3 textures . . . 48
9.4 ant::baseComponent Class Reference . . . 48
9.4.1 Detailed Description . . . 50
9.4.2 Constructor & Destructor Documentation . . . 50
9.4.2.1 baseComponent . . . 50

9.4.2.2 ~baseComponent . . . 50
9.4.3 Member Function Documentation . . . 50
9.4.3.1 getId . . . 50
9.4.3.2 getProperties . . . 51
9.4.3.3 setId . . . 51
9.4.3.4 setProperties . . . 51
9.4.4 Member Data Documentation . . . 52
9.4.4.1 id . . . 52
9.5 ant::baseEvent Class Reference . . . 52
9.5.1 Detailed Description . . . 54
9.5.2 Constructor & Destructor Documentation . . . 54
9.5.2.1 baseEvent . . . 54
9.5.2.2 ~baseEvent . . . 54
9.5.3 Member Function Documentation . . . 54
9.5.3.1 getAttributes . . . 54
9.5.3.2 getType . . . 55
9.5.3.3 setAttributes . . . 55
9.5.3.4 setType . . . 56
9.5.4 Member Data Documentation . . . 57
9.5.4.1 type . . . 57
9.6 ant::GUI::Button Class Reference . . . 57
9.6.1 Detailed Description . . . 60
9.6.2 Constructor & Destructor Documentation . . . 60
9.6.2.1 Button . . . 60
9.6.2.2 Button . . . 60
9.6.2.3 Button . . . 60
9.6.2.4 Button . . . 61
9.6.2.5 ~Button . . . 61
9.6.3 Member Function Documentation . . . 61
9.6.3.1 contains . . . 61
9.6.3.2 draw . . . 61
9.6.3.3 getAction . . . 61
9.6.3.4 getPosition . . . 61
9.6.3.5 getSize . . . 61
9.6.3.6 getSprite . . . 61
9.6.3.7 getText . . . 61
9.6.3.8 setAction . . . 61
9.6.3.9 setImage . . . 61
9.6.3.10 setPosition . . . 61
9.6.3.11 setSize . . . 62

9.6.3.12 setText . . . 62
9.6.4 Member Data Documentation . . . 62
9.6.4.1 action . . . 62
9.6.4.2 image . . . 62
9.6.4.3 position . . . 62
9.6.4.4 size . . . 62
9.6.4.5 text . . . 62
9.7 ant::collisionSystem Class Reference . . . 62
9.7.1 Detailed Description . . . 65
9.7.2 Constructor & Destructor Documentation . . . 66
9.7.2.1 collisionSystem . . . 66
9.7.2.2 ∼collisionSystem . . . 67
9.7.3 Member Function Documentation . . . 67
9.7.3.1 getEntityManager . . . 67
9.7.3.2 getEventQueue . . . 67
9.7.3.3 getName . . . 67
9.7.3.4 getRequiredComponts . . . 67
9.7.3.5 render . . . 67
9.7.3.6 setEntityManager . . . 68
9.7.3.7 setEventQueue . . . 68
9.7.3.8 setName . . . 68
9.7.3.9 setRequiredComponents . . . 68
9.7.3.10 testTerrainCollision . . . 68
9.7.3.11 update . . . 69
9.7.4 Member Data Documentation . . . 70
9.7.4.1 em . . . 70
9.7.4.2 eventQueue . . . 70
9.7.4.3 gameBounds . . . 70
9.7.4.4 gameMap . . . 70
9.7.4.5 name . . . 70
9.7.4.6 qtree . . . 70
9.7.4.7 RequiredComponents . . . 70
9.8 ant::Component< T > Class Template Reference . . . 71
9.8.1 Detailed Description . . . 72
9.8.2 Constructor & Destructor Documentation . . . 73
9.8.2.1 Component . . . 73
9.8.3 Member Function Documentation . . . 73
9.8.3.1 getId . . . 73
9.8.3.2 getProperties . . . 73
9.8.3.3 setId . . . 74

9.8.3.4 setProperties . . . 74
9.8.4 Member Data Documentation . . . 74
9.8.4.1 properties . . . 74
9.9 ant::ComponentFactory Class Reference . . . 75
9.9.1 Detailed Description . . . 76
9.9.2 Constructor & Destructor Documentation . . . 76
9.9.2.1 ComponentFactory . . . 76
9.9.2.2 ComponentFactory . . . 77
9.9.2.3 ~ComponentFactory . . . 78
9.9.3 Member Function Documentation . . . 78
9.9.3.1 createAnimation . . . 78
9.9.3.2 createBounds . . . 78
9.9.3.3 createCounter . . . 78
9.9.3.4 createDestructable . . . 79
9.9.3.5 createIn . . . 79
9.9.3.6 createOut . . . 79
9.9.3.7 createPassage . . . 80
9.9.3.8 createSprite . . . 80
9.9.3.9 createTransform . . . 80
9.9.3.10 createVelocity . . . 80
9.9.3.11 getAssetManager . . . 81
9.9.3.12 loadAssets . . . 81
9.9.3.13 mapImage . . . 81
9.9.3.14 setAssetManager . . . 82
9.9.3.15 unloadAssets . . . 82
9.9.4 Member Data Documentation . . . 82
9.9.4.1 assets . . . 82
9.10 ant::ComponentSettings Class Reference . . . 82
9.10.1 Detailed Description . . . 84
9.10.2 Constructor & Destructor Documentation . . . 84
9.10.2.1 ComponentSettings . . . 84
9.10.2.2 ~ComponentSettings . . . 84
9.10.3 Member Function Documentation . . . 84
9.10.3.1 getSpeed . . . 84
9.10.3.2 loadSettings . . . 84
9.10.3.3 setSpeed . . . 85
9.10.4 Member Data Documentation . . . 85
9.10.4.1 animation . . . 86
9.10.4.2 bounds . . . 86
9.10.4.3 count . . . 86

9.10.4.4 dest . . . 86
9.10.4.5 DestructiblePosition . . . 86
9.10.4.6 entityName . . . 86
9.10.4.7 imageID . . . 86
9.10.4.8 maxSpeed . . . 86
9.10.4.9 minSpeed . . . 86
9.10.4.10 position . . . 86
9.10.4.11 rotation . . . 86
9.10.4.12 scale . . . 86
9.10.4.13 Speed . . . 87
9.10.4.14 spriteID . . . 87
9.11 ComponentsMask Class Reference . . . 87
9.11.1 Detailed Description . . . 88
9.11.2 Member Data Documentation . . . 88
9.11.2.1 COMPONENT_ANIMATION . . . 88
9.11.2.2 COMPONENT_BOUNDS . . . 88
9.11.2.3 COMPONENT_COUNT . . . 88
9.11.2.4 COMPONENT_DESTRUCTABLE . . . 88
9.11.2.5 COMPONENT_IN . . . 88
9.11.2.6 COMPONENT_NONE . . . 88
9.11.2.7 COMPONENT_OUT . . . 88
9.11.2.8 COMPONENT_PASSAGE . . . 88
9.11.2.9 COMPONENT_SPRITE . . . 88
9.11.2.10 COMPONENT_TRANSFORM . . . 89
9.11.2.11 COMPONENT_VELOCITY . . . 89
9.12 ant::Config Class Reference . . . 89
9.12.1 Detailed Description . . . 90
9.12.2 Member Data Documentation . . . 90
9.12.2.1 ANT_FILE . . . 90
9.12.2.2 ASSETS_GAME_JSON . . . 90
9.12.2.3 CONFIG_FILE . . . 90
9.12.2.4 GUI_FILE . . . 90
9.12.2.5 INDOOR_FILE . . . 90
9.12.2.6 LEVEL_FILE . . . 90
9.12.2.7 LEVELS_FILE . . . 90
9.12.2.8 OUTDOOR_FILE . . . 90
9.12.2.9 screenSize . . . 90
9.12.2.10 VERSION . . . 91
9.13 ant::Constructions Class Reference . . . 91
9.13.1 Detailed Description . . . 92

9.13.2 Member Data Documentation . . . 92
9.13.2.1 bridge . . . 92
9.13.2.2 Brown . . . 92
9.13.2.3 climb . . . 92
9.13.2.4 downhill . . . 93
9.13.2.5 Grey . . . 93
9.13.2.6 hole . . . 93
9.13.2.7 lightBrown . . . 93
9.13.2.8 lightGrey . . . 93
9.13.2.9 stairs . . . 93
9.13.2.10 stop . . . 93
9.13.2.11 tunnel . . . 94
9.13.2.12 uphill . . . 94
9.13.2.13 wall . . . 94
9.14 ant::constructorSystem Class Reference . . . 94
9.14.1 Detailed Description . . . 97
9.14.2 Member Typedef Documentation . . . 98
9.14.2.1 command . . . 98
9.14.3 Constructor & Destructor Documentation . . . 98
9.14.3.1 constructorSystem . . . 98
9.14.3.2 ~constructorSystem . . . 98
9.14.4 Member Function Documentation . . . 98
9.14.4.1 getEntityManager . . . 98
9.14.4.2 getEventQueue . . . 98
9.14.4.3 getName . . . 98
9.14.4.4 getRequiredComponts . . . 99
9.14.4.5 onNotify . . . 99
9.14.4.6 render . . . 99
9.14.4.7 setEntityManager . . . 99
9.14.4.8 setEventQueue . . . 100
9.14.4.9 setName . . . 100
9.14.4.10 setRequiredComponents . . . 100
9.14.4.11 update . . . 100
9.14.5 Member Data Documentation . . . 100
9.14.5.1 boundsMap . . . 100
9.14.5.2 build . . . 100
9.14.5.3 em . . . 101
9.14.5.4 eventQueue . . . 101
9.14.5.5 map . . . 101
9.14.5.6 name . . . 101

9.14.5.7 readyToBuild . . . 101
9.14.5.8 RequiredComponents . . . 101
9.15 ant::AppState::Context Struct Reference . . . 101
9.15.1 Detailed Description . . . 102
9.15.2 Constructor & Destructor Documentation . . . 102
9.15.2.1 Context . . . 102
9.15.3 Member Data Documentation . . . 102
9.15.3.1 assets . . . 102
9.15.3.2 window . . . 102
9.16 ant::Entity Class Reference . . . 102
9.16.1 Detailed Description . . . 104
9.16.2 Member Typedef Documentation . . . 105
9.16.2.1 map . . . 105
9.16.3 Constructor & Destructor Documentation . . . 105
9.16.3.1 Entity . . . 105
9.16.3.2 Entity . . . 105
9.16.3.3 Entity . . . 105
9.16.3.4 ∼Entity . . . 105
9.16.4 Member Function Documentation . . . 105
9.16.4.1 addComponent . . . 105
9.16.4.2 addState . . . 105
9.16.4.3 clear . . . 106
9.16.4.4 getComponent . . . 106
9.16.4.5 getComponents . . . 107
9.16.4.6 getMask . . . 107
9.16.4.7 getName . . . 107
9.16.4.8 getStates . . . 107
9.16.4.9 hasComponent . . . 108
9.16.4.10 is . . . 109
9.16.4.11 operator"!= . . . 109
9.16.4.12 operator== . . . 110
9.16.4.13 operator== . . . 110
9.16.4.14 removeComponent . . . 110
9.16.4.15 removeState . . . 111
9.16.4.16 setComponents . . . 112
9.16.4.17 setName . . . 112
9.16.5 Member Data Documentation . . . 112
9.16.5.1 Components . . . 112
9.16.5.2 mask . . . 113
9.16.5.3 name . . . 113

9.16.5.4 states . . . 113
9.17 ant::EntityFactory Class Reference . . . 113
9.17.1 Detailed Description . . . 114
9.17.2 Constructor & Destructor Documentation . . . 114
9.17.2.1 EntityFactory . . . 114
9.17.2.2 EntityFactory . . . 115
9.17.2.3 ~EntityFactory . . . 116
9.17.3 Member Function Documentation . . . 116
9.17.3.1 createEntity . . . 116
9.17.3.2 createEntity . . . 117
9.17.3.3 getAssetManager . . . 118
9.17.3.4 getComponentFactory . . . 118
9.17.3.5 loadAssets . . . 118
9.17.3.6 setAssetManager . . . 118
9.17.3.7 setComponentFactory . . . 119
9.17.3.8 unloadAssets . . . 119
9.17.4 Member Data Documentation . . . 119
9.17.4.1 Ant . . . 119
9.17.4.2 componentFactory . . . 119
9.17.4.3 Door . . . 119
9.17.4.4 InDoor . . . 120
9.17.4.5 level . . . 120
9.17.4.6 OutDoor . . . 120
9.18 ant::EntityManager Class Reference . . . 120
9.18.1 Detailed Description . . . 122
9.18.2 Member Typedef Documentation . . . 122
9.18.2.1 iterator . . . 122
9.18.2.2 list . . . 122
9.18.3 Constructor & Destructor Documentation . . . 122
9.18.3.1 EntityManager . . . 122
9.18.3.2 ~EntityManager . . . 122
9.18.4 Member Function Documentation . . . 123
9.18.4.1 addEntities . . . 123
9.18.4.2 addEntity . . . 123
9.18.4.3 begin . . . 123
9.18.4.4 end . . . 123
9.18.4.5 getEntities . . . 123
9.18.4.6 getEntity . . . 123
9.18.4.7 removeEntity . . . 124
9.18.4.8 removeEntity . . . 124

9.18.4.9 removeEntity . . . 124
9.18.4.10 Transfer . . . 124
9.18.4.11 Transfer . . . 124
9.18.5 Member Data Documentation . . . 125
9.18.5.1 entities . . . 125
9.19 ant::Event< T > Class Template Reference . . . 125
9.19.1 Detailed Description . . . 126
9.19.2 Constructor & Destructor Documentation . . . 127
9.19.2.1 Event . . . 127
9.19.3 Member Function Documentation . . . 127
9.19.3.1 getAttributes . . . 127
9.19.3.2 getType . . . 127
9.19.3.3 setAttributes . . . 128
9.19.3.4 setType . . . 128
9.19.4 Member Data Documentation . . . 128
9.19.4.1 attributes . . . 128
9.20 ant::EventQueue Class Reference . . . 128
9.20.1 Detailed Description . . . 129
9.20.2 Constructor & Destructor Documentation . . . 130
9.20.2.1 EventQueue . . . 130
9.20.2.2 ~EventQueue . . . 130
9.20.3 Member Function Documentation . . . 130
9.20.3.1 clear . . . 130
9.20.3.2 front . . . 130
9.20.3.3 isEmpty . . . 130
9.20.3.4 pop . . . 130
9.20.3.5 push . . . 130
9.20.3.6 size . . . 131
9.20.4 Member Data Documentation . . . 131
9.20.4.1 eventQueue . . . 131
9.21 EventType Class Reference . . . 131
9.21.1 Detailed Description . . . 132
9.21.2 Member Data Documentation . . . 132
9.21.2.1 CHANGE_COMMAND . . . 132
9.21.2.2 CHANGE_NENTITIES . . . 132
9.21.2.3 CHANGE_OVERTIME . . . 132
9.21.2.4 COLLISION_EVENT . . . 132
9.21.2.5 LEVEL_COMPLETE . . . 132
9.21.2.6 LEVEL_FAILED . . . 132
9.21.2.7 OUT_MAP . . . 133

9.21.2.8 SELECT_ENTITY . . . 133
9.21.2.9 TERRAIN_COLLISION . . . 133
9.22 ant::GameEventDispatcher Class Reference . . . 133
9.22.1 Detailed Description . . . 134
9.22.2 Constructor & Destructor Documentation . . . 134
9.22.2.1 GameEventDispatcher . . . 134
9.22.2.2 ~GameEventDispatcher . . . 134
9.22.3 Member Function Documentation . . . 135
9.22.3.1 DispatchEvent . . . 135
9.22.4 Member Data Documentation . . . 136
9.22.4.1 ChangeCommand . . . 136
9.22.4.2 ClickEvents . . . 136
9.22.4.3 LevelComplete . . . 136
9.22.4.4 LevelFailed . . . 136
9.22.4.5 onCollision . . . 136
9.22.4.6 outMap . . . 137
9.22.4.7 spawnEvents . . . 137
9.23 ant::GameOverState Class Reference . . . 137
9.23.1 Detailed Description . . . 139
9.23.2 Member Typedef Documentation . . . 139
9.23.2.1 sPtr . . . 139
9.23.2.2 uPtr . . . 140
9.23.3 Constructor & Destructor Documentation . . . 140
9.23.3.1 GameOverState . . . 140
9.23.4 Member Function Documentation . . . 140
9.23.4.1 getContext . . . 140
9.23.4.2 handleEvent . . . 141
9.23.4.3 loadConfig . . . 141
9.23.4.4 render . . . 142
9.23.4.5 requestStackPop . . . 142
9.23.4.6 requestStackPush . . . 143
9.23.4.7 requestStateClear . . . 144
9.23.4.8 update . . . 144
9.23.5 Member Data Documentation . . . 145
9.23.5.1 font . . . 145
9.23.5.2 mText . . . 145
9.24 ant::GameState Class Reference . . . 145
9.24.1 Detailed Description . . . 149
9.24.2 Member Typedef Documentation . . . 149
9.24.2.1 sPtr . . . 149

9.24.2.2 uPtr . . . 149
9.24.3 Constructor & Destructor Documentation . . . 149
9.24.3.1 GameState . . . 149
9.24.3.2 ~GameState . . . 149
9.24.4 Member Function Documentation . . . 150
9.24.4.1 getContext . . . 150
9.24.4.2 handleEvent . . . 150
9.24.4.3 loadConfig . . . 151
9.24.4.4 loadGUIConf . . . 151
9.24.4.5 onNotify . . . 151
9.24.4.6 render . . . 152
9.24.4.7 requestStackPop . . . 152
9.24.4.8 requestStackPush . . . 152
9.24.4.9 requestStateClear . . . 153
9.24.4.10 update . . . 154
9.24.5 Member Data Documentation . . . 154
9.24.5.1 assets . . . 154
9.24.5.2 buttons . . . 154
9.24.5.3 currentLevel . . . 154
9.24.5.4 eventQueue . . . 154
9.24.5.5 font . . . 154
9.24.5.6 gameEventDispatcher . . . 154
9.24.5.7 GameSpeed . . . 155
9.24.5.8 level . . . 155
9.24.5.9 totalLevels . . . 155
9.24.5.10 win . . . 155
9.25 ant::gravitySystem Class Reference . . . 155
9.25.1 Detailed Description . . . 158
9.25.2 Constructor & Destructor Documentation . . . 158
9.25.2.1 gravitySystem . . . 158
9.25.2.2 gravitySystem . . . 159
9.25.2.3 ~gravitySystem . . . 160
9.25.3 Member Function Documentation . . . 160
9.25.3.1 getEntityManager . . . 160
9.25.3.2 getEventQueue . . . 160
9.25.3.3 getGravity . . . 160
9.25.3.4 getName . . . 160
9.25.3.5 getRequiredComponts . . . 160
9.25.3.6 render . . . 161
9.25.3.7 setEntityManager . . . 161

9.25.3.8 setEventQueue . . . 161
9.25.3.9 setGravity . . . 161
9.25.3.10 setName . . . 161
9.25.3.11 setRequiredComponents . . . 161
9.25.3.12 update . . . 161
9.25.4 Member Data Documentation . . . 162
9.25.4.1 em . . . 162
9.25.4.2 eventQueue . . . 162
9.25.4.3 Gravity . . . 162
9.25.4.4 name . . . 162
9.25.4.5 RequiredComponents . . . 162
9.26 ant::Level Class Reference . . . 162
9.26.1 Detailed Description . . . 164
9.26.2 Constructor & Destructor Documentation . . . 164
9.26.2.1 Level . . . 164
9.26.2.2 Level . . . 165
9.26.2.3 ~Level . . . 166
9.26.3 Member Function Documentation . . . 166
9.26.3.1 getAssetManager . . . 166
9.26.3.2 getEventQueue . . . 166
9.26.3.3 getWorldFactory . . . 166
9.26.3.4 init . . . 166
9.26.3.5 loadLevel . . . 167
9.26.3.6 render . . . 168
9.26.3.7 setAssetManager . . . 168
9.26.3.8 setEventQueue . . . 168
9.26.3.9 setWorldFactory . . . 168
9.26.3.10 size . . . 168
9.26.3.11 update . . . 169
9.26.4 Member Data Documentation . . . 169
9.26.4.1 eventQueue . . . 169
9.26.4.2 gameEventDispatcher . . . 169
9.26.4.3 levels . . . 169
9.26.4.4 worldFactory . . . 169
9.27 ant::MenuState Class Reference . . . 169
9.27.1 Detailed Description . . . 172
9.27.2 Member Typedef Documentation . . . 173
9.27.2.1 sPtr . . . 173
9.27.2.2 uPtr . . . 173
9.27.3 Member Enumeration Documentation . . . 173

9.27.3.1 OptionNames . . . 173
9.27.4 Constructor & Destructor Documentation . . . 173
9.27.4.1 MenuState . . . 173
9.27.5 Member Function Documentation . . . 173
9.27.5.1 getContext . . . 174
9.27.5.2 handleEvent . . . 174
9.27.5.3 loadConfig . . . 175
9.27.5.4 render . . . 176
9.27.5.5 requestStackPop . . . 176
9.27.5.6 requestStackPush . . . 177
9.27.5.7 requestStateClear . . . 178
9.27.5.8 update . . . 178
9.27.5.9 updateOptionText . . . 179
9.27.6 Member Data Documentation . . . 179
9.27.6.1 font . . . 179
9.27.6.2 mBackground . . . 179
9.27.6.3 mOptionIndex . . . 179
9.27.6.4 mOptions . . . 179
9.27.6.5 strBackground . . . 179
9.28 ant::movementSystem Class Reference . . . 180
9.28.1 Detailed Description . . . 182
9.28.2 Constructor & Destructor Documentation . . . 182
9.28.2.1 movementSystem . . . 182
9.28.2.2 ~movementSystem . . . 182
9.28.3 Member Function Documentation . . . 183
9.28.3.1 getEntityManager . . . 183
9.28.3.2 getEventQueue . . . 183
9.28.3.3 getName . . . 183
9.28.3.4 getRequiredComponts . . . 183
9.28.3.5 onNotify . . . 183
9.28.3.6 render . . . 184
9.28.3.7 setEntityManager . . . 184
9.28.3.8 setEventQueue . . . 184
9.28.3.9 setName . . . 184
9.28.3.10 setRequiredComponents . . . 184
9.28.3.11 update . . . 184
9.28.4 Member Data Documentation . . . 185
9.28.4.1 em . . . 185
9.28.4.2 eventQueue . . . 185
9.28.4.3 name . . . 185

9.28.4.4 RequiredComponents . . . 185
9.29 ant::Observer Class Reference . . . 185
9.29.1 Detailed Description . . . 186
9.29.2 Constructor & Destructor Documentation . . . 186
9.29.2.1 Observer . . . 186
9.29.2.2 ~Observer . . . 186
9.29.3 Member Function Documentation . . . 187
9.29.3.1 onNotify . . . 187
9.30 ant::outSystem Class Reference . . . 188
9.30.1 Detailed Description . . . 190
9.30.2 Constructor & Destructor Documentation . . . 190
9.30.2.1 outSystem . . . 190
9.30.2.2 outSystem . . . 191
9.30.2.3 ~outSystem . . . 192
9.30.3 Member Function Documentation . . . 192
9.30.3.1 getEntityManager . . . 192
9.30.3.2 getEventQueue . . . 192
9.30.3.3 getName . . . 192
9.30.3.4 getRequiredComponts . . . 192
9.30.3.5 onNotify . . . 192
9.30.3.6 render . . . 193
9.30.3.7 setEntityManager . . . 193
9.30.3.8 setEventQueue . . . 193
9.30.3.9 setName . . . 193
9.30.3.10 setRequiredComponents . . . 193
9.30.3.11 update . . . 193
9.30.4 Member Data Documentation . . . 193
9.30.4.1 em . . . 194
9.30.4.2 eventQueue . . . 194
9.30.4.3 name . . . 194
9.30.4.4 RequiredComponents . . . 194
9.30.4.5 savedEntities . . . 194
9.30.4.6 totalEntities . . . 194
9.31 ant::PauseState Class Reference . . . 194
9.31.1 Detailed Description . . . 197
9.31.2 Member Typedef Documentation . . . 198
9.31.2.1 sPtr . . . 198
9.31.2.2 uPtr . . . 198
9.31.3 Constructor & Destructor Documentation . . . 198
9.31.3.1 PauseState . . . 198

9.31.4 Member Function Documentation . . . 198
9.31.4.1 getContext . . . 198
9.31.4.2 handleEvent . . . 199
9.31.4.3 loadConfig . . . 199
9.31.4.4 render . . . 200
9.31.4.5 requestStackPop . . . 200
9.31.4.6 requestStackPush . . . 201
9.31.4.7 requestStateClear . . . 202
9.31.4.8 update . . . 202
9.31.5 Member Data Documentation . . . 203
9.31.5.1 font . . . 203
9.31.5.2 mShowText . . . 203
9.31.5.3 mText . . . 203
9.31.5.4 mTextEffectTime . . . 203
9.32 ant::StateStack::PendingChange Struct Reference . . . 203
9.32.1 Detailed Description . . . 204
9.32.2 Constructor & Destructor Documentation . . . 204
9.32.2.1 PendingChange . . . 204
9.32.3 Member Data Documentation . . . 204
9.32.3.1 action . . . 204
9.32.3.2 stateID . . . 204
9.33 ant::Utils::Quadtree Class Reference . . . 204
9.33.1 Detailed Description . . . 206
9.33.2 Member Typedef Documentation . . . 206
9.33.2.1 list . . . 206
9.33.3 Constructor & Destructor Documentation . . . 206
9.33.3.1 Quadtree . . . 206
9.33.3.2 ~Quadtree . . . 207
9.33.3.3 Quadtree . . . 207
9.33.4 Member Function Documentation . . . 207
9.33.4.1 clear . . . 207
9.33.4.2 getIndex . . . 207
9.33.4.3 insert . . . 208
9.33.4.4 render . . . 209
9.33.4.5 retrieve . . . 209
9.33.4.6 split . . . 210
9.33.5 Member Data Documentation . . . 211
9.33.5.1 bounds . . . 211
9.33.5.2 entities . . . 211
9.33.5.3 level . . . 211

9.33.5.4 MAX_CAPACITY . . . 211
9.33.5.5 MAX_LEVEL . . . 211
9.33.5.6 nodes . . . 211
9.34 ant::renderSystem Class Reference . . . 211
9.34.1 Detailed Description . . . 214
9.34.2 Constructor & Destructor Documentation . . . 214
9.34.2.1 renderSystem . . . 214
9.34.2.2 ~renderSystem . . . 214
9.34.3 Member Function Documentation . . . 214
9.34.3.1 getEntityManager . . . 214
9.34.3.2 getEventQueue . . . 215
9.34.3.3 getName . . . 215
9.34.3.4 getRequiredComponts . . . 215
9.34.3.5 render . . . 215
9.34.3.6 setEntityManager . . . 215
9.34.3.7 setEventQueue . . . 215
9.34.3.8 setName . . . 216
9.34.3.9 setRequiredComponents . . . 217
9.34.3.10 update . . . 217
9.34.4 Member Data Documentation . . . 217
9.34.4.1 em . . . 217
9.34.4.2 eventQueue . . . 217
9.34.4.3 name . . . 217
9.34.4.4 RequiredComponents . . . 217
9.35 ant::spawnSystem Class Reference . . . 217
9.35.1 Detailed Description . . . 220
9.35.2 Constructor & Destructor Documentation . . . 220
9.35.2.1 spawnSystem . . . 220
9.35.2.2 ~spawnSystem . . . 221
9.35.3 Member Function Documentation . . . 221
9.35.3.1 getEntityManager . . . 221
9.35.3.2 getEventQueue . . . 221
9.35.3.3 getName . . . 221
9.35.3.4 getRequiredComponts . . . 221
9.35.3.5 getStates . . . 222
9.35.3.6 onNotify . . . 222
9.35.3.7 render . . . 222
9.35.3.8 setEntityManager . . . 222
9.35.3.9 setEventQueue . . . 222
9.35.3.10 setName . . . 222

9.35.3.11 setRequiredComponents . . . 223
9.35.3.12 setStates . . . 223
9.35.3.13 update . . . 223
9.35.4 Member Data Documentation . . . 223
9.35.4.1 clock . . . 223
9.35.4.2 createdEntities . . . 223
9.35.4.3 elapsedTime . . . 224
9.35.4.4 em . . . 224
9.35.4.5 entityFactory . . . 224
9.35.4.6 eventQueue . . . 224
9.35.4.7 name . . . 224
9.35.4.8 nEntities . . . 224
9.35.4.9 overTime . . . 224
9.35.4.10 RequiredComponents . . . 224
9.35.4.11 spawnPoint . . . 224
9.35.4.12 states . . . 224
9.36 States Class Reference . . . 224
9.36.1 Detailed Description . . . 225
9.36.2 Member Data Documentation . . . 225
9.36.2.1 BUILDING . . . 225
9.36.2.2 CLIMBING . . . 225
9.36.2.3 FALLING . . . 226
9.36.2.4 GROUND . . . 226
9.36.2.5 NONE . . . 226
9.36.2.6 ONFIRE . . . 226
9.36.2.7 SAVED . . . 226
9.36.2.8 UNSAVED . . . 226
9.37 ant::StateStack Class Reference . . . 226
9.37.1 Detailed Description . . . 228
9.37.2 Member Enumeration Documentation . . . 229
9.37.2.1 Action . . . 229
9.37.3 Constructor & Destructor Documentation . . . 229
9.37.3.1 StateStack . . . 229
9.37.3.2 ~StateStack . . . 229
9.37.4 Member Function Documentation . . . 229
9.37.4.1 applyPendingChanges . . . 229
9.37.4.2 clearStates . . . 230
9.37.4.3 createState . . . 230
9.37.4.4 handleEvent . . . 230
9.37.4.5 isEmpty . . . 231

9.37.4.6 popState . . . 231
9.37.4.7 pushState . . . 231
9.37.4.8 registerState . . . 232
9.37.4.9 registerState . . . 232
9.37.4.10 render . . . 233
9.37.4.11 update . . . 233
9.37.5 Member Data Documentation . . . 233
9.37.5.1 mContext . . . 233
9.37.5.2 mFactories . . . 234
9.37.5.3 mPendingList . . . 234
9.37.5.4 mStack . . . 234
9.38 ant::Subject Class Reference . . . 234
9.38.1 Detailed Description . . . 235
9.38.2 Constructor & Destructor Documentation . . . 235
9.38.2.1 Subject . . . 235
9.38.2.2 ∼Subject . . . 235
9.38.3 Member Function Documentation . . . 235
9.38.3.1 addObserver . . . 235
9.38.3.2 notifyObservers . . . 235
9.38.3.3 removeObserver . . . 236
9.38.4 Member Data Documentation . . . 236
9.38.4.1 Observers . . . 236
9.39 ant::System Class Reference . . . 236
9.39.1 Detailed Description . . . 238
9.39.2 Constructor & Destructor Documentation . . . 238
9.39.2.1 System . . . 238
9.39.2.2 System . . . 239
9.39.2.3 System . . . 240
9.39.2.4 ∼System . . . 240
9.39.3 Member Function Documentation . . . 240
9.39.3.1 getEntityManager . . . 240
9.39.3.2 getEventQueue . . . 240
9.39.3.3 getName . . . 240
9.39.3.4 getRequiredComponts . . . 241
9.39.3.5 render . . . 241
9.39.3.6 setEntityManager . . . 241
9.39.3.7 setEventQueue . . . 241
9.39.3.8 setName . . . 241
9.39.3.9 setRequiredComponents . . . 241
9.39.3.10 update . . . 241

9.39.4 Member Data Documentation . . . 242
9.39.4.1 em . . . 242
9.39.4.2 eventQueue . . . 242
9.39.4.3 name . . . 242
9.39.4.4 RequiredComponents . . . 242
9.40 ant::SystemFactory Class Reference . . . 242
9.40.1 Detailed Description . . . 244
9.40.2 Constructor & Destructor Documentation . . . 244
9.40.2.1 SystemFactory . . . 244
9.40.2.2 SystemFactory . . . 244
9.40.2.3 ~SystemFactory . . . 244
9.40.3 Member Function Documentation . . . 245
9.40.3.1 createCollisionSystem . . . 245
9.40.3.2 createConstructorSystem . . . 246
9.40.3.3 createGravitySystem . . . 246
9.40.3.4 createMovementSystem . . . 246
9.40.3.5 createOutSystem . . . 247
9.40.3.6 createRenderSystem . . . 247
9.40.3.7 createSpawnSystem . . . 247
9.40.3.8 getEventQueue . . . 247
9.40.3.9 getGameEventDispatcher . . . 247
9.40.3.10 setEventQueue . . . 248
9.40.3.11 setGameEventDispatcher . . . 249
9.40.4 Member Data Documentation . . . 249
9.40.4.1 eventQueue . . . 249
9.40.4.2 gameEventDispatcher . . . 249
9.41 ant::SystemManager Class Reference . . . 249
9.41.1 Detailed Description . . . 251
9.41.2 Member Typedef Documentation . . . 251
9.41.2.1 map . . . 251
9.41.3 Constructor & Destructor Documentation . . . 251
9.41.3.1 SystemManager . . . 251
9.41.3.2 ~SystemManager . . . 251
9.41.4 Member Function Documentation . . . 251
9.41.4.1 addSystem . . . 251
9.41.4.2 getSystem . . . 251
9.41.4.3 getSystems . . . 252
9.41.4.4 removeSystem . . . 252
9.41.4.5 render . . . 252
9.41.4.6 setEntityManager . . . 252

9.41.4.7 setEventQueue . . . 252
9.41.4.8 setSystems . . . 253
9.41.4.9 update . . . 253
9.41.5 Member Data Documentation . . . 253
9.41.5.1 systems . . . 253
9.42 ant::TitleState Class Reference . . . 253
9.42.1 Detailed Description . . . 256
9.42.2 Member Typedef Documentation . . . 257
9.42.2.1 sPtr . . . 257
9.42.2.2 uPtr . . . 257
9.42.3 Constructor & Destructor Documentation . . . 257
9.42.3.1 TitleState . . . 257
9.42.4 Member Function Documentation . . . 257
9.42.4.1 getContext . . . 257
9.42.4.2 handleEvent . . . 258
9.42.4.3 loadConfig . . . 258
9.42.4.4 render . . . 259
9.42.4.5 requestStackPop . . . 259
9.42.4.6 requestStackPush . . . 260
9.42.4.7 requestStateClear . . . 261
9.42.4.8 update . . . 261
9.42.5 Member Data Documentation . . . 262
9.42.5.1 background . . . 262
9.42.5.2 font . . . 262
9.42.5.3 mBackground . . . 262
9.42.5.4 mShowText . . . 262
9.42.5.5 mText . . . 262
9.42.5.6 mTextEffectTime . . . 262
9.42.5.7 mTitle . . . 262
9.43 ant::World Class Reference . . . 262
9.43.1 Detailed Description . . . 264
9.43.2 Constructor & Destructor Documentation . . . 264
9.43.2.1 World . . . 264
9.43.2.2 World . . . 264
9.43.2.3 World . . . 265
9.43.2.4 ~World . . . 265
9.43.3 Member Function Documentation . . . 265
9.43.3.1 getEntityManager . . . 265
9.43.3.2 getEventQueue . . . 265
9.43.3.3 getGameEventDispatcher . . . 265

9.43.3.4 getId . . . 266
9.43.3.5 getSystemManager . . . 266
9.43.3.6 render . . . 266
9.43.3.7 setEntityManager . . . 266
9.43.3.8 setEventQueue . . . 266
9.43.3.9 setGameEventDispatcher . . . 266
9.43.3.10 setId . . . 267
9.43.3.11 setSystemManager . . . 267
9.43.3.12 update . . . 267
9.43.4 Member Data Documentation . . . 267
9.43.4.1 entityManager . . . 267
9.43.4.2 eventQueue . . . 267
9.43.4.3 gameEventDispatcher . . . 267
9.43.4.4 id . . . 267
9.43.4.5 systemManager . . . 268
9.44 ant::WorldFactory Class Reference . . . 268
9.44.1 Detailed Description . . . 269
9.44.2 Constructor & Destructor Documentation . . . 269
9.44.2.1 WorldFactory . . . 269
9.44.2.2 WorldFactory . . . 270
9.44.2.3 ~WorldFactory . . . 271
9.44.3 Member Function Documentation . . . 271
9.44.3.1 create . . . 271
9.44.3.2 getAssetManager . . . 272
9.44.3.3 getEntityFactory . . . 272
9.44.3.4 getEventQueue . . . 272
9.44.3.5 getGameEventDispatcher . . . 272
9.44.3.6 loadAssets . . . 272
9.44.3.7 setAssetManager . . . 273
9.44.3.8 setEntityFactory . . . 273
9.44.3.9 setEventQueue . . . 273
9.44.3.10 setGameEventDispatcher . . . 273
9.44.3.11 unloadAssets . . . 273
9.44.4 Member Data Documentation . . . 274
9.44.4.1 entityFactory . . . 274
9.44.4.2 eventQueue . . . 274
9.44.4.3 gameEventDispatcher . . . 274
9.44.4.4 lvlID . . . 274
9.44.4.5 systemFactory . . . 274
9.45 ant::WorldManager Class Reference . . . 274

9.45.1 Detailed Description . . . 276
9.45.2 Member Typedef Documentation . . . 276
9.45.2.1 iterator . . . 276
9.45.2.2 map . . . 276
9.45.3 Constructor & Destructor Documentation . . . 276
9.45.3.1 WorldManager . . . 276
9.45.3.2 ∼WorldManager . . . 276
9.45.4 Member Function Documentation . . . 276
9.45.4.1 addWorld . . . 276
9.45.4.2 clear . . . 277
9.45.4.3 getWorld . . . 277
9.45.4.4 getWorlds . . . 277
9.45.4.5 removeWorld . . . 277
9.45.4.6 render . . . 277
9.45.4.7 setWorlds . . . 278
9.45.4.8 size . . . 278
9.45.4.9 update . . . 278
9.45.5 Member Data Documentation . . . 278
9.45.5.1 worlds . . . 278

10 File Documentation **279**
10.1 Application.cpp File Reference . . . 279
10.2 Application.hpp File Reference . . . 279
10.3 AppState.cpp File Reference . . . 280
10.4 AppState.hpp File Reference . . . 281
10.5 AppStates.hpp File Reference . . . 281
10.6 AssetManager.cpp File Reference . . . 282
10.7 AssetManager.hpp File Reference . . . 282
10.8 Button.cpp File Reference . . . 283
10.9 Button.hpp File Reference . . . 283
10.10 collisionSystem.cpp File Reference . . . 284
10.11 collisionSystem.hpp File Reference . . . 285
10.12 Component.cpp File Reference . . . 286
10.13 Component.hpp File Reference . . . 286
10.14 ComponentFactory.cpp File Reference . . . 287
10.15 ComponentFactory.hpp File Reference . . . 287
10.16 ComponentMask.hpp File Reference . . . 289
10.17 ComponentSettings.cpp File Reference . . . 289
10.18 ComponentSettings.hpp File Reference . . . 290
10.19 Config.cpp File Reference . . . 290

10.20 Config.hpp File Reference . . . 291
10.21 Constructions.cpp File Reference . . . 292
10.22 Constructions.hpp File Reference . . . 292
10.23 constructorSystem.cpp File Reference . . . 293
10.24 constructorSystem.hpp File Reference . . . 294
10.25 Entity.cpp File Reference . . . 295
10.26 Entity.hpp File Reference . . . 295
10.27 EntityFactory.cpp File Reference . . . 296
10.28 EntityFactory.hpp File Reference . . . 296
10.29 EntityManager.cpp File Reference . . . 297
10.30 EntityManager.hpp File Reference . . . 298
10.31 Event.cpp File Reference . . . 299
10.32 Event.hpp File Reference . . . 299
10.33 EventQueue.cpp File Reference . . . 300
10.34 EventQueue.hpp File Reference . . . 301
10.35 EventType.hpp File Reference . . . 302
10.36 GameEventDispatcher.cpp File Reference . . . 303
10.37 GameEventDispatcher.hpp File Reference . . . 303
10.38 GameOverState.cpp File Reference . . . 305
10.39 GameOverState.hpp File Reference . . . 305
10.40 GameState.cpp File Reference . . . 306
10.41 GameState.hpp File Reference . . . 306
10.42 gravitySystem.cpp File Reference . . . 307
10.43 gravitySystem.hpp File Reference . . . 308
10.44 Level.cpp File Reference . . . 309
10.45 Level.hpp File Reference . . . 310
10.46 main.cpp File Reference . . . 311
10.46.1 Function Documentation . . . 311
10.46.1.1 main . . . 311
10.47 mainpage.dox File Reference . . . 311
10.48 Math.cpp File Reference . . . 311
10.49 Math.hpp File Reference . . . 312
10.50 MenuState.cpp File Reference . . . 314
10.51 MenuState.hpp File Reference . . . 314
10.52 movementSystem.cpp File Reference . . . 315
10.53 movementSystem.hpp File Reference . . . 315
10.54 Observer.hpp File Reference . . . 316
10.55 outSystem.cpp File Reference . . . 317
10.56 outSystem.hpp File Reference . . . 318
10.57 PauseState.cpp File Reference . . . 319

10.58 PauseState.hpp File Reference . . . 320
10.59 Quadtree.cpp File Reference . . . 320
10.60 Quadtree.hpp File Reference . . . 321
10.61 README.md File Reference . . . 322
10.62 renderSystem.cpp File Reference . . . 322
10.63 renderSystem.hpp File Reference . . . 323
10.64 spawnSystem.cpp File Reference . . . 324
10.65 spawnSystem.hpp File Reference . . . 325
10.66 States.hpp File Reference . . . 326
10.67 StateStack.cpp File Reference . . . 326
10.68 StateStack.hpp File Reference . . . 327
10.69 String.cpp File Reference . . . 327
10.70 String.hpp File Reference . . . 328
10.71 Subject.cpp File Reference . . . 329
10.72 Subject.hpp File Reference . . . 330
10.73 System.cpp File Reference . . . 332
10.74 System.hpp File Reference . . . 332
10.75 SystemFactory.cpp File Reference . . . 333
10.76 SystemFactory.hpp File Reference . . . 334
10.77 SystemManager.cpp File Reference . . . 335
10.78 SystemManager.hpp File Reference . . . 335
10.79 TitleState.cpp File Reference . . . 336
10.80 TitleState.hpp File Reference . . . 337
10.81 utility.cpp File Reference . . . 337
10.82 utility.hpp File Reference . . . 338
10.83 World.cpp File Reference . . . 339
10.84 World.hpp File Reference . . . 340
10.85 WorldFactory.cpp File Reference . . . 341
10.86 WorldFactory.hpp File Reference . . . 341
10.87 WorldManager.cpp File Reference . . . 342
10.88 WorldManager.hpp File Reference . . . 343

Chapter 1

Main Page

Juego sobre hormigas atrapadas que deben escapar.Para pasar al siguiente nivel las hormigas tienen que llegar a la puerta de salida. Las hormigas se moverolas, al colisionar con una pared estas rebotaran y caminaran en la misma direcciro en sentido contrario. Con los controles y acciones de las hormigas tendremos que conseguir que estas lleguen a la puerta de salida intentando perder el menor nmero de unidades posibles, como mmo tendremos que salvar a una hormiga para poder pasar al siguiente nivel.

Author

Cristian Glez `Cristian.glez.m@gmail.com`
Xavi Diaz `xdiazcarrasco@iesesteveterradas.cat`

Version

0.1

Chapter 2

AntFarm

AntFarm is a game about ants that are trapped in some place and need to get to the other door to escape.

Requires CMake and one of the following compilers:

- GCC 4.8 or newer;
- Clang 3.3 or newer (Xcode 5.0 or newer on OS X);
- Visual Studio 2010 or newer;

Installing

AntFarm requires the following dependencies:

- `SFML` 2.1 or newer
- `JsonBox` 0.4.4 or newer

Building

Use CMake to build the project.

- select the root directory of JsonBox
- select the root directory of SFML
- press configure and then generate Project.

Documentation

The documentation can be found on "doc" or be generated with doxygen(it needs to be configured)

Chapter 3

Todo List

Member ant::baseComponent::baseComponent (long int id)

posible cambio en el id de long int a bitset<NBits>

Member ant::movementSystem::update (sf::Time dt)

agregar al calculo delta time

Class ant::Utils::Quadtree

Posible cambio por Quadtree<T> tarda 1ms mas.

Chapter 4

Namespace Index

4.1 Namespace List

Here is a list of all namespaces with brief descriptions:

ant . . . . . ??

ant::AppStates

Tiene los ID de los estados de la aplicacion . . . . . ??

ant::GUI . . . . . ??

ant::Utils . . . . . ??

Chapter 5

Hierarchical Index

5.1 Class Hierarchy

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

ant::Application . . . . . ??
ant::AppState . . . . . ??
  ant::GameOverState . . . . . ??
  ant::GameState . . . . . ??
  ant::MenuState . . . . . ??
  ant::PauseState . . . . . ??
  ant::TitleState . . . . . ??
ant::AssetManager . . . . . ??
ant::baseComponent . . . . . ??
  ant::Component< T > . . . . . ??
ant::baseEvent . . . . . ??
  ant::Event< T > . . . . . ??
ant::ComponentFactory . . . . . ??
ant::ComponentSettings . . . . . ??
ComponentsMask . . . . . ??
ant::Config . . . . . ??
ant::Constructions . . . . . ??
ant::AppState::Context . . . . . ??
Drawable
  ant::GUI::Button . . . . . ??
ant::Entity . . . . . ??
ant::EntityFactory . . . . . ??
ant::EntityManager . . . . . ??
ant::EventQueue . . . . . ??
EventType . . . . . ??
ant::GameEventDispatcher . . . . . ??
ant::Level . . . . . ??
ant::Observer . . . . . ??
  ant::constructorSystem . . . . . ??
  ant::GameState . . . . . ??
  ant::movementSystem . . . . . ??
  ant::outSystem . . . . . ??
  ant::spawnSystem . . . . . ??
ant::StateStack::PendingChange . . . . . ??
ant::Utils::Quadtree . . . . . ??
States . . . . . ??
ant::StateStack . . . . . ??

ant::Subject . . . . . ??
ant::System . . . . . ??
ant::collisionSystem . . . . . ??
ant::constructorSystem . . . . . ??
ant::gravitySystem . . . . . ??
ant::movementSystem . . . . . ??
ant::outSystem . . . . . ??
ant::renderSystem . . . . . ??
ant::spawnSystem . . . . . ??
ant::SystemFactory . . . . . ??
ant::SystemManager . . . . . ??
ant::World . . . . . ??
ant::WorldFactory . . . . . ??
ant::WorldManager . . . . . ??

Chapter 6

Class Index

6.1 Class List

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

ant::Application
Clase de aplicacion . . . . . ??
ant::AppState
Clase abstracta de estado de aplicacion . . . . . ??
ant::AssetManager
Clase para tener todos los assets(imagenes,texturas,etc) del juego . . . . . ??
ant::baseComponent
Clase base de componente . . . . . ??
ant::baseEvent
Clase Base de eventos . . . . . ??
ant::GUI::Button
Clase de un simple boton . . . . . ??
ant::collisionSystem
Sistema que comprueba las colisiones entre entidades . . . . . ??
ant::Component< T >
Clase especializada de baseComponent . . . . . ??
ant::ComponentFactory
Clase para crear los componentes de las entidades . . . . . ??
ant::ComponentSettings
Clase para la configuracion de los componentes en EntityFactory . . . . . ??
ComponentsMask
Tipos de componentes . . . . . ??
ant::Config
Clase de configuracion con constantes de ficheros . . . . . ??
ant::Constructions
Clase estatica con los comandos para las hormigas . . . . . ??
ant::constructorSystem
Sistema encargado de que las hormigas construyan etc . . . . . ??
ant::AppState::Context
Clase para compartir datos esenciales a los estados . . . . . ??
ant::Entity
Clase para tener los componentes, solo puede tener un componente de cada tipo . . . . . ??
ant::EntityFactory
Clase para crear las entidades del juego . . . . . ??
ant::EntityManager
Clase Para manejar las entidades . . . . . ??
ant::Event< T >
Clase Especializada de evento . . . . . ??

ant::EventQueue
Clase donde se almacenan los eventos para ser procesados despues . . . . . ??
EventType
Tipos de eventos . . . . . ??
ant::GameEventDispatcher
Clase encargada de processar eventos . . . . . ??
ant::GameOverState
Muestra el mensaje de final de juego . . . . . ??
ant::GameState
Estado de juego, Inicia el juego . . . . . ??
ant::gravitySystem
Sistema para aplicar la gravedad a las entidades . . . . . ??
ant::Level
Clase que tiene el nivel del juego . . . . . ??
ant::MenuState
Clase de estado del menu . . . . . ??
ant::movementSystem
Sistema encargado de hacer que las entidades se muevan . . . . . ??
ant::Observer
Interfaz para los observadores . . . . . ??
ant::outSystem
Sistema encargado de quitar a las entidades que chocan con las entidades con componente OUT . . . . . ??
ant::PauseState
Estado de pausa . . . . . ??
ant::StateStack::PendingChange . . . . . ??
ant::Utils::Quadtree
Clase para reducir el numero de test de collisiones . . . . . ??
ant::renderSystem
Sistema encargado de mostrar las entidades que tengan los componentes necesarios . . . . ??
ant::spawnSystem
Sistema encargado de crear las entidades sobre el tiempo establecido . . . . . ??
States
Estados de las entidades . . . . . ??
ant::StateStack
Clase para almacenar los estados del juego . . . . . ??
ant::Subject
Tema al cual se suscriben los observadores Observer . . . . . ??
ant::System
Clase base para los sistemas . . . . . ??
ant::SystemFactory
Clase para crear sistemas . . . . . ??
ant::SystemManager
Clase contenedor para los systemas . . . . . ??
ant::TitleState
Muestra el titulo de la aplicacion . . . . . ??
ant::World
Clase para crear escenarios . . . . . ??
ant::WorldFactory
Clase para crear mundos . . . . . ??
ant::WorldManager
Clase Contenedor para los mundos . . . . . ??

Chapter 7

File Index

7.1 File List

Here is a list of all files with brief descriptions:

Application.cpp . . . . . ??
Application.hpp . . . . . ??
AppState.cpp . . . . . ??
AppState.hpp . . . . . ??
AppStates.hpp . . . . . ??
AssetManager.cpp . . . . . ??
AssetManager.hpp . . . . . ??
Button.cpp . . . . . ??
Button.hpp . . . . . ??
collisionSystem.cpp . . . . . ??
collisionSystem.hpp . . . . . ??
Component.cpp . . . . . ??
Component.hpp . . . . . ??
ComponentFactory.cpp . . . . . ??
ComponentFactory.hpp . . . . . ??
ComponentMask.hpp . . . . . ??
ComponentSettings.cpp . . . . . ??
ComponentSettings.hpp . . . . . ??
Config.cpp . . . . . ??
Config.hpp . . . . . ??
Constructions.cpp . . . . . ??
Constructions.hpp . . . . . ??
constructorSystem.cpp . . . . . ??
constructorSystem.hpp . . . . . ??
Entity.cpp . . . . . ??
Entity.hpp . . . . . ??
EntityFactory.cpp . . . . . ??
EntityFactory.hpp . . . . . ??
EntityManager.cpp . . . . . ??
EntityManager.hpp . . . . . ??
Event.cpp . . . . . ??
Event.hpp . . . . . ??
EventQueue.cpp . . . . . ??
EventQueue.hpp . . . . . ??
EventType.hpp . . . . . ??
GameEventDispatcher.cpp . . . . . ??
GameEventDispatcher.hpp . . . . . ??
GameOverState.cpp . . . . . ??

GameOverState.hpp . . . . . ??
GameState.cpp . . . . . ??
GameState.hpp . . . . . ??
gravitySystem.cpp . . . . . ??
gravitySystem.hpp . . . . . ??
Level.cpp . . . . . ??
Level.hpp . . . . . ??
main.cpp . . . . . ??
Math.cpp . . . . . ??
Math.hpp . . . . . ??
MenuState.cpp . . . . . ??
MenuState.hpp . . . . . ??
movementSystem.cpp . . . . . ??
movementSystem.hpp . . . . . ??
Observer.hpp . . . . . ??
outSystem.cpp . . . . . ??
outSystem.hpp . . . . . ??
PauseState.cpp . . . . . ??
PauseState.hpp . . . . . ??
Quadtree.cpp . . . . . ??
Quadtree.hpp . . . . . ??
renderSystem.cpp . . . . . ??
renderSystem.hpp . . . . . ??
spawnSystem.cpp . . . . . ??
spawnSystem.hpp . . . . . ??
States.hpp . . . . . ??
StateStack.cpp . . . . . ??
StateStack.hpp . . . . . ??
String.cpp . . . . . ??
String.hpp . . . . . ??
Subject.cpp . . . . . ??
Subject.hpp . . . . . ??
System.cpp . . . . . ??
System.hpp . . . . . ??
SystemFactory.cpp . . . . . ??
SystemFactory.hpp . . . . . ??
SystemManager.cpp . . . . . ??
SystemManager.hpp . . . . . ??
TitleState.cpp . . . . . ??
TitleState.hpp . . . . . ??
utility.cpp . . . . . ??
utility.hpp . . . . . ??
World.cpp . . . . . ??
World.hpp . . . . . ??
WorldFactory.cpp . . . . . ??
WorldFactory.hpp . . . . . ??
WorldManager.cpp . . . . . ??
WorldManager.hpp . . . . . ??

Chapter 8

Namespace Documentation

8.1 ant Namespace Reference

Namespaces

- AppStates

  Tiene los ID de los estados de la aplicacion.
- GUI
- Utils

Classes

- class Application

  Clase de aplicacion.
- class AppState

  Clase abstracta de estado de aplicacion.
- class AssetManager

  Clase para tener todos los assets(imagenes,texturas,etc) del juego.
- class baseComponent

  Clase base de componente.
- class baseEvent

  Clase Base de eventos.
- class collisionSystem

  Sistema que comprueba las colisiones entre entidades.
- class Component

  Clase especializada de baseComponent.
- class ComponentFactory

  Clase para crear los componentes de las entidades.
- class ComponentSettings

  Clase para la configuracion de los componentes en EntityFactory.
- class Config

  Clase de configuracion con constantes de ficheros.
- class Constructions

  Clase estatica con los comandos para las hormigas.
- class constructorSystem

  Sistema encargado de que las hormigas construyan etc.
- class Entity

Clase para tener los componentes, solo puede tener un componente de cada tipo.

- class EntityFactory

  Clase para crear las entidades del juego.
- class EntityManager

  Clase Para manejar las entidades.
- class Event

  Clase Especializada de evento.
- class EventQueue

  Clase donde se almacenan los eventos para ser procesados despues.
- class GameEventDispatcher

  Clase encargada de processar eventos.
- class GameOverState

  Muestra el mensaje de final de juego.
- class GameState

  Estado de juego, Inicia el juego.
- class gravitySystem

  Sistema para aplicar la gravedad a las entidades.
- class Level

  Clase que tiene el nivel del juego.
- class MenuState

  Clase de estado del menu.
- class movementSystem

  Sistema encargado de hacer que las entidades se muevan.
- class Observer

  Interfaz para los observadores.
- class outSystem

  Sistema encargado de quitar a las entidades que chocan con las entidades con componente OUT.
- class PauseState

  Estado de pausa.
- class renderSystem

  Sistema encargado de mostrar las entidades que tengan los componentes necesarios.
- class spawnSystem

  Sistema encargado de crear las entidades sobre el tiempo establecido.
- class StateStack

  Clase para almacenar los estados del juego.
- class Subject

  Tema al cual se suscriben los observadores Observer.
- class System

  Clase base para los sistemas.
- class SystemFactory

  Clase para crear sistemas.
- class SystemManager

  Clase contenedor para los systemas.
- class TitleState

  Muestra el titulo de la aplicacion.
- class World

  Clase para crear escenarios.
- class WorldFactory

  Clase para crear mundos.
- class WorldManager

  Clase Contenedor para los mundos.

8.2 ant::AppStates Namespace Reference

Tiene los ID de los estados de la aplicacion.

Enumerations

- enum ID {
  None, Title, Menu, Game,
  Pause, GameFailed, GameSucceed }

  ID de los estados de la aplicacion.

8.2.1 Detailed Description

Tiene los ID de los estados de la aplicacion.

8.2.2 Enumeration Type Documentation

8.2.2.1 enum ant::AppStates::ID

ID de los estados de la aplicacion.

Enumerator

None

Title

Menu

Game

Pause

GameFailed

GameSucceed

Definition at line 11 of file AppStates.hpp.

8.3 ant::GUI Namespace Reference

Classes

- class Button

  Clase de un simple boton.

8.4 ant::Utils Namespace Reference

Classes

- class Quadtree

  Clase para reducir el numero de test de collisiones.

Functions

- template<typename T >
  T toRadians (T degrees)

  Convertira el valor introducido a radianes.

- template<typename T >
  T toDegrees (T radians)
- sf::FloatRect rotateRect (const sf::FloatRect &rect, const float &angle)
- template<typename T >
  T getAngle (const sf::Vector2< T > &v)
- sf::Vector2f normalize (const sf::Vector2f &v)
- template<typename T >
  int dot (const sf::Vector2< T > &v1, const sf::Vector2< T > &v2)
- bool RectContains (const sf::FloatRect &e, const sf::FloatRect &bounds)

  Comprueba que un rectangulo esta completamente dentro de otro.

- float getDirection (const sf::Vector2f &currentPosition, const sf::Vector2f &wantedPosition)
- template<typename T >
  std::string toString (T s)

  Convierte el Tipo a string.

- template<typename T , typename... Args>
  std::unique_ptr< T > makeUniquePtr (Args &&...args)

  Utilidad para crear un puntero unico.

- template<typename T , typename... Args>
  std::shared_ptr< T > makeSharedPtr (Args &&...args)

  Utilidad para crear un puntero Compartido.

- void centerOrigin (sf::Text &text)

  Centra el texto.

- void centerOrigin (sf::Sprite &sprite)

  Centra el sprite.

Variables

- constexpr double PI = 3.14159265358979

  Definicion de PI.

8.4.1 Function Documentation

8.4.1.1 void ant::Utils::centerOrigin (sf::Text & *text*)

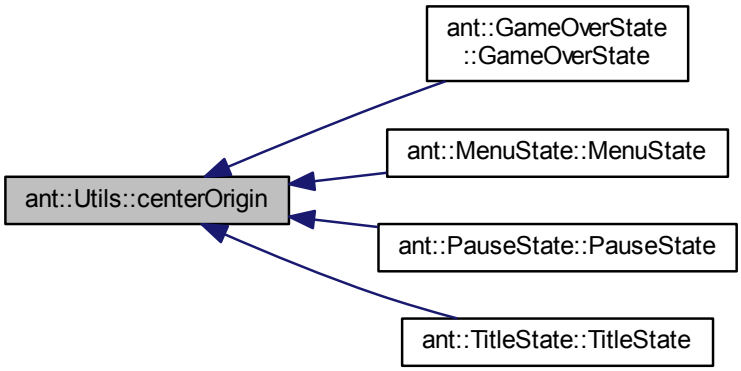
Centra el texto.

Parameters

text	sf::Text & referencia al texto.

Definition at line 10 of file utility.cpp.

Here is the caller graph for this function:



8.4.1.2 void ant::Utils::centerOrigin (sf::Sprite & *sprite*)

Centra el sprite.

Parameters

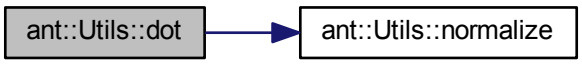
sprite	sf::Sprite & referencia al sprite.

Definition at line 6 of file utility.cpp.

8.4.1.3 template<typename T > int ant::Utils::dot (const sf::Vector2< T > & *v1*, const sf::Vector2< T > & *v2*)

Definition at line 74 of file Math.hpp.

Here is the call graph for this function:



8.4.1.4 template<typename T > T ant::Utils::getAngle (const sf::Vector2< T > & *v*)

Definition at line 51 of file Math.hpp.

8.4.1.5 float ant::Utils::getDirection (const sf::Vector2f & *currentPosition*, const sf::Vector2f & *wantedPosition*)

Definition at line 33 of file Math.cpp.

Here is the call graph for this function:

8.4.1.6 template<typename T , typename... Args> std::shared_ptr<T> ant::Utils::makeSharedPtr (Args &&... *args*)

Utilidad para crear un puntero Compartido.

Template Parameters

T	clase para crear el puntero
args	Args argumentos

Returns

std::shared_ptr<T<Args>>

Definition at line 25 of file utility.hpp.

8.4.1.7 template<typename T , typename... Args> std::unique_ptr<T> ant::Utils::makeUniquePtr (Args &&... *args*)

Utilidad para crear un puntero unico.

Template Parameters

T	clase para crear el puntero
args	Args argumentos

Returns

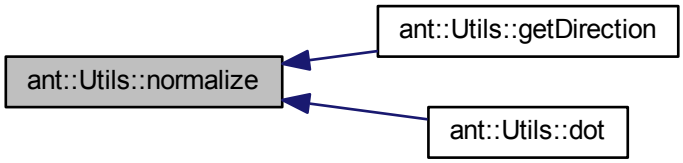
std::unique_ptr<T<Args>>

Definition at line 15 of file utility.hpp.

8.4.1.8 sf::Vector2f ant::Utils::normalize (const sf::Vector2f & *v*)

Definition at line 18 of file Math.cpp.

Here is the caller graph for this function:



8.4.1.9 bool ant::Utils::RectContains (const sf::FloatRect & *e*, const sf::FloatRect & *bounds*)

Comprueba que un rectangulo esta completamente dentro de otro.

Parameters

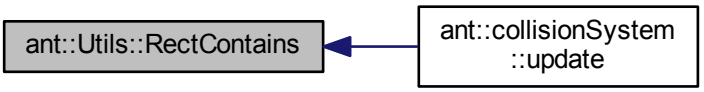
e	sf::FloatRect rectangulo que tiene que estar dentro del otro.
bounds	sf::FloatRect rectangulo en el cual tienen que estar dentro.

Returns

bool

Definition at line 27 of file Math.cpp.

Here is the caller graph for this function:



8.4.1.10 sf::FloatRect ant::Utils::rotateRect (const sf::FloatRect & *rect*, const float & *angle*)

Definition at line 5 of file Math.cpp.

Here is the call graph for this function:

8.4.1.11 template<typename T > T ant::Utils::toDegrees (T *radians*) [inline]

Definition at line 30 of file Math.hpp.

8.4.1.12 template<typename T > T ant::Utils::toRadians (T *degrees*) [inline]

Convertira el valor introducido a radianes.

Template Parameters

T	tipo a convertir a radianes

Parameters

degrees	T (float,double,char,int,long) etc

Returns

T

Definition at line 19 of file Math.hpp.

Here is the caller graph for this function:

8.4.1.13 template<typename T > std::string ant::Utils::toString (T *s*) [inline]

Convierte el Tipo a string.

Template Parameters

T	tipo a convertir a string.

Parameters

s	T tipo a convertir a string.

Returns

std::string

Definition at line 14 of file String.hpp.

Here is the caller graph for this function:

8.4.2 Variable Documentation

8.4.2.1 constexpr double ant::Utils::PI = 3.14159265358979

Definicion de PI.

Definition at line 11 of file Math.hpp.

Chapter 9

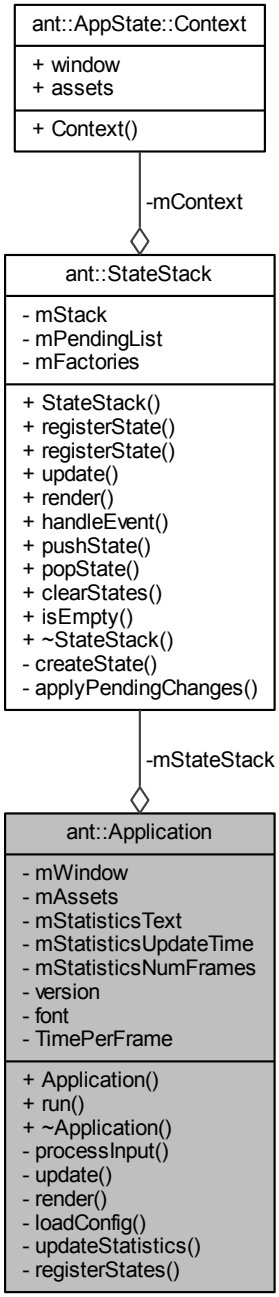
Class Documentation

9.1 ant::Application Class Reference

Clase de aplicacion.

```
#include <Application.hpp>
```

Collaboration diagram for ant::Application:



Public Member Functions

- Application ()

  Constructor Principal.
- void run ()

  Comienza la aplicacion.
- ~Application ()

Private Member Functions

- void processInput ()

  Procesa el input del teclado/raton etc.
- void update (sf::Time dt)

  Actualiza el estado de la aplicacion.
- void render ()

  Renderiza la aplicacion.
- void loadConfig (const std::string &filename)

  Carga la configuracion.
- void updateStatistics (sf::Time dt)

  Actualiza las estadisticas del juego.
- void registerStates ()

  Registra los estados que usara la aplicacion.

Private Attributes

- std::shared_ptr< sf::RenderWindow > mWindow
- std::shared_ptr< AssetManager > mAssets
- StateStack mStateStack
- sf::Text mStatisticsText
- sf::Time mStatisticsUpdateTime
- std::size_t mStatisticsNumFrames
- sf::Text version
- std::string font

Static Private Attributes

- static const sf::Time TimePerFrame = sf::seconds(1.f/60.f)

9.1.1 Detailed Description

Clase de aplicacion.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 19 of file Application.hpp.

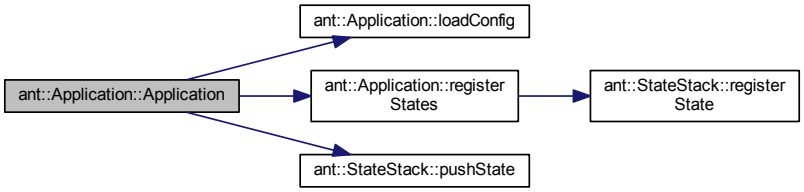
9.1.2 Constructor & Destructor Documentation

9.1.2.1 ant::Application::Application ( )

Constructor Principal.

Definition at line 8 of file Application.cpp.

Here is the call graph for this function:



9.1.2.2 ant::Application::~Application ( )

Definition at line 93 of file Application.cpp.

9.1.3 Member Function Documentation

9.1.3.1 void ant::Application::loadConfig (const std::string & *filename*) [private]

Carga la configuracion.

Parameters

filename	std::string nombre del fichero de configuracion.

Definition at line 42 of file Application.cpp.

Here is the caller graph for this function:

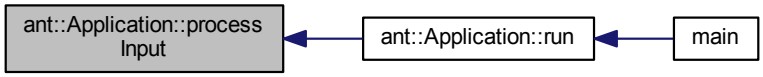
9.1.3.2 void ant::Application::processInput ( ) [private]

Procesa el input del teclado/raton etc.

Definition at line 49 of file Application.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

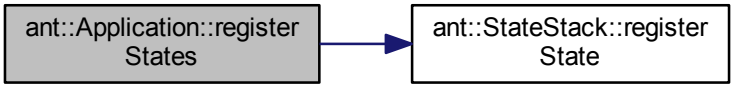


9.1.3.3 void ant::Application::registerStates ( ) [private]

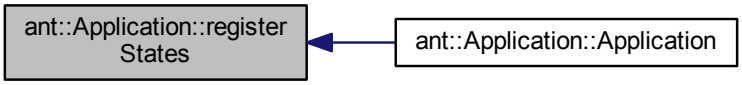
Registra los estados que usara la aplicacion.

Definition at line 85 of file Application.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



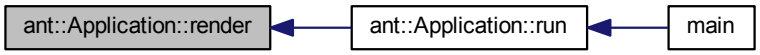
9.1.3.4 void ant::Application::render ( ) [private]

Renderiza la aplicacion.

Definition at line 61 of file Application.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

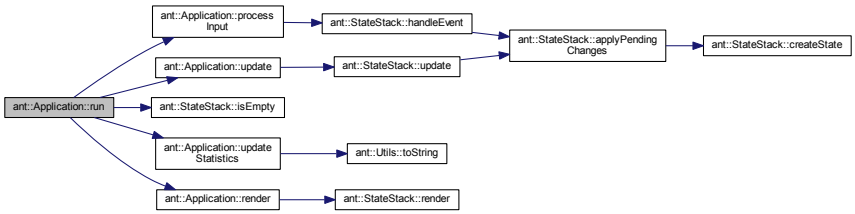


9.1.3.5 void ant::Application::run ( )

Comienza la aplicacion.

Definition at line 24 of file Application.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

9.1.3.6 void ant::Application::update (sf::Time *dt*) [private]

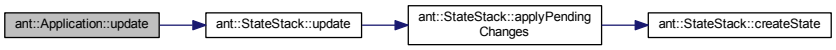
Actualiza el estado de la aplicacion.

Parameters

dt	sf::Time delta time

Definition at line 58 of file Application.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

9.1.3.7 void ant::Application::updateStatistics (sf::Time *dt*) [private]

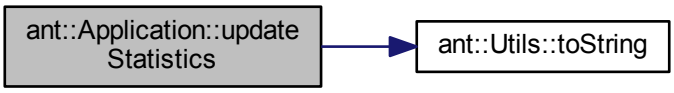
Actualiza las estadisticas del juego.

Parameters

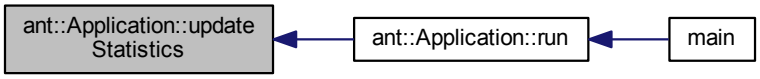
dt	sf::Time

Definition at line 69 of file Application.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.1.4 Member Data Documentation

9.1.4.1 std::string ant::Application::font [private]

Definition at line 67 of file Application.hpp.

9.1.4.2 std::shared_ptr<AssetManager> ant::Application::mAssets [private]

Definition at line 61 of file Application.hpp.

9.1.4.3 StateStack ant::Application::mStateStack [private]

Definition at line 62 of file Application.hpp.

9.1.4.4 std::size_t ant::Application::mStatisticsNumFrames [private]

Definition at line 65 of file Application.hpp.

9.1.4.5 sf::Text ant::Application::mStatisticsText [private]

Definition at line 63 of file Application.hpp.

9.1.4.6 sf::Time ant::Application::mStatisticsUpdateTime [private]

Definition at line 64 of file Application.hpp.

9.1.4.7 std::shared_ptr<sf::RenderWindow> ant::Application::mWindow [private]

Definition at line 60 of file Application.hpp.

9.1.4.8 const sf::Time ant::Application::TimePerFrame = sf::seconds(1.f/60.f) [static], [private]

Definition at line 59 of file Application.hpp.

9.1.4.9 sf::Text ant::Application::version [private]

Definition at line 66 of file Application.hpp.

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

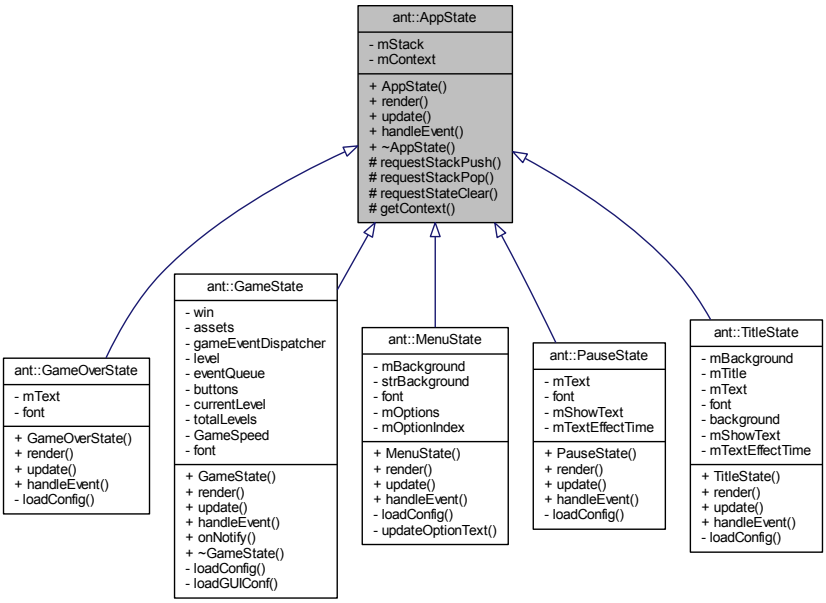
- Application.hpp
- Application.cpp

9.2 ant::AppState Class Reference

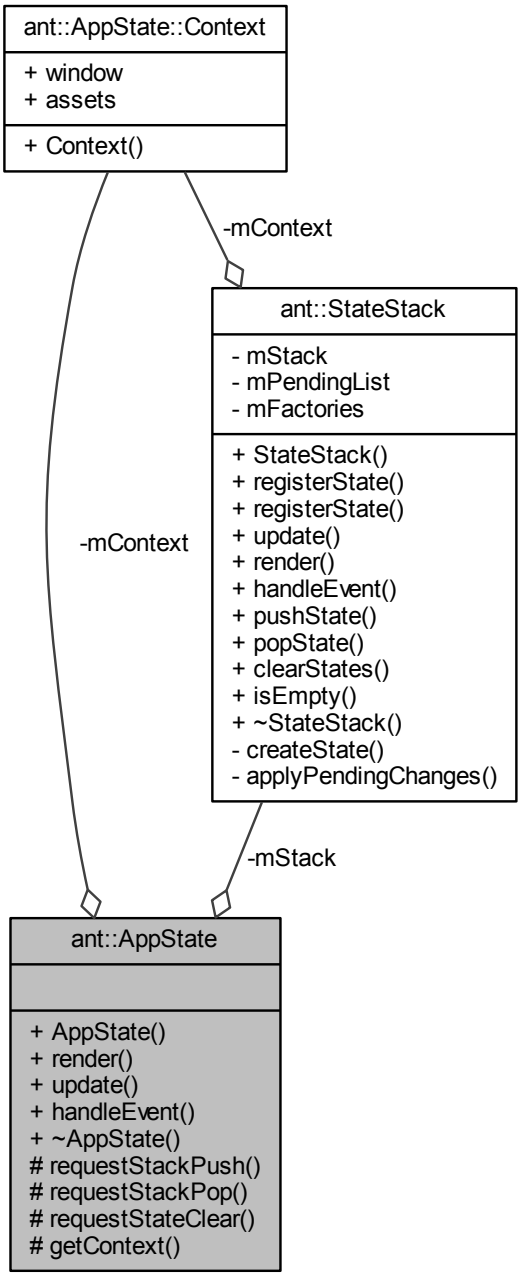
Clase abstracta de estado de aplicacion.

```
#include <AppState.hpp>
```

Inheritance diagram for ant::AppState:



Collaboration diagram for ant::AppState:



Classes

- struct Context

  Clase para compartir datos esenciales a los estados.

Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.
- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- AppState (StateStack &stack, AppState::Context context)

  Constructor principal.
- virtual void render ()=0

  Metodo abstracto para que los estados dibujen a la pantalla.
- virtual bool update (sf::Time dt)=0

  Metodo abstracto para que los estados actualizen su estado.
- virtual bool handleEvent (const sf::Event &event)=0

  Metodo abstracto para que los estados manejen el input.
- virtual ~AppState ()

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Attributes

- StateStack * mStack
- Context mContext

9.2.1 Detailed Description

Clase abstracta de estado de aplicacion.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 17 of file AppState.hpp.

9.2.2 Member Typedef Documentation

9.2.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.2.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

9.2.3 Constructor & Destructor Documentation

9.2.3.1 ant::AppState::AppState (StateStack & *stack*, AppState::Context *context*)

Constructor principal.

Parameters

stack	StateStack referencia a un stack,
context	AppState::Context contexto de ejecucion

Definition at line 7 of file AppState.cpp.

9.2.3.2 ant::AppState::~AppState ( ) [virtual]

Definition at line 24 of file AppState.cpp.

9.2.4 Member Function Documentation

9.2.4.1 AppState::Context ant::AppState::getContext ( ) const [protected]

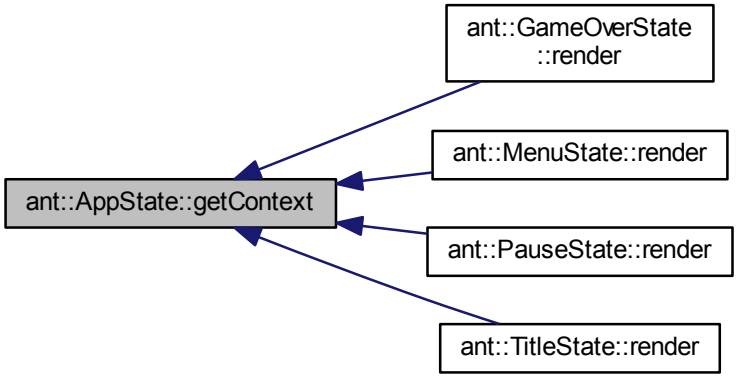
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



9.2.4.2 virtual bool ant::AppState::handleEvent (const sf::Event & *event*) [pure virtual]

Metodo abstracto para que los estados manejen el input.

Returns

bool retorna true para que los demas tambien manejen eventos.

Implemented in ant::GameState, ant::GameOverState, ant::MenuState, ant::PauseState, and ant::TitleState.

9.2.4.3 virtual void ant::AppState::render ( ) [pure virtual]

Metodo abstracto para que los estados dibujen a la pantalla.

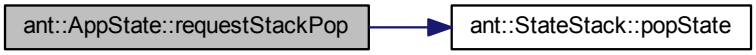
Implemented in ant::GameState, ant::GameOverState, ant::MenuState, ant::TitleState, and ant::PauseState.

9.2.4.4 void ant::AppState::requestStackPop ( ) [protected]

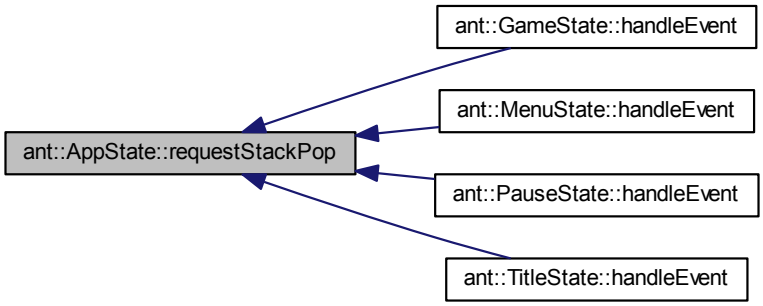
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.2.4.5 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected]

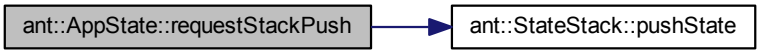
Pide al stack que a un estado nuevo.

Parameters

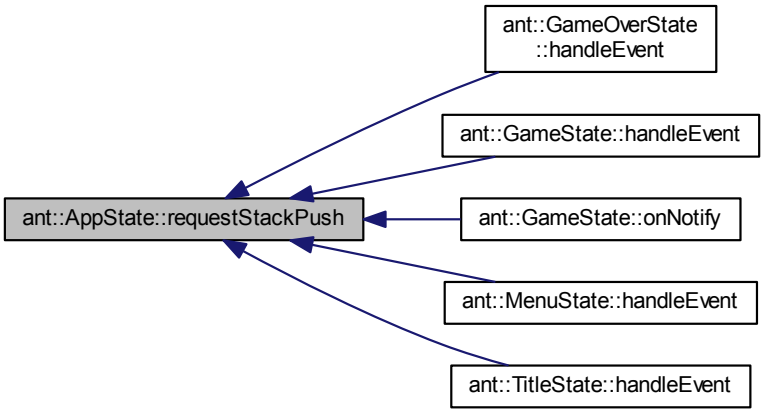
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

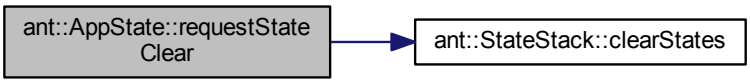


9.2.4.6 void ant::AppState::requestStateClear ( ) [protected]

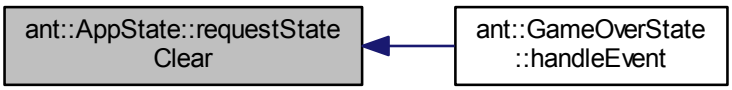
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.2.4.7 virtual bool ant::AppState::update (sf::Time *dt*) [pure virtual]

Metodo abstracto para que los estados actualizen su estado.

Returns

bool retorna true para que los demas tambien actualicen.

Implemented in ant::GameState, ant::GameOverState, ant::MenuState, ant::PauseState, and ant::TitleState.

9.2.5 Member Data Documentation

9.2.5.1 Context ant::AppState::mContext [private]

Definition at line 83 of file AppState.hpp.

9.2.5.2 StateStack* ant::AppState::mStack [private]

Definition at line 82 of file AppState.hpp.

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

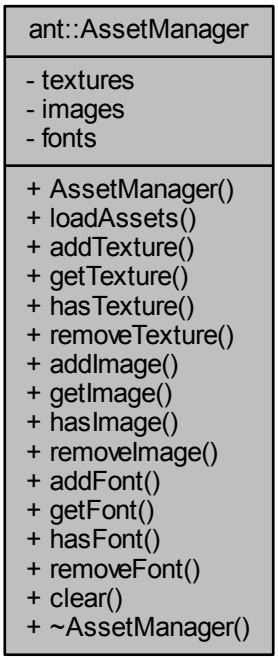
- AppState.hpp
- AppState.cpp

9.3 ant::AssetManager Class Reference

Clase para tener todos los assets(imagenes,texturas,etc) del juego.

```
#include <AssetManager.hpp>
```

Collaboration diagram for ant::AssetManager:



Public Member Functions

- AssetManager ()

  Constructor principal sin argumentos.
- bool loadAssets (const std::string &filename)

  Carga un fichero json con el siguiente formato:
- void addTexture (const std::string &id, const std::string &filename)

  A una textura con el id proporcionado.
- sf::Texture & getTexture (const std::string &id)

  Devuelve una referencia a la textura, si no existe, tira excepcion de fuera de rango.
- bool hasTexture (const std::string &id)

  Comprueba si tiene la textura.
- void removeTexture (const std::string &id)

  Elimina la textura.
- void addImage (const std::string &id, const std::string &filename)

  A una imagen con el id especificado.
- sf::Image & getImage (const std::string &id)

  Devuelve una referencia a la imagen, si no existe tira excepcion.
- bool hasImage (const std::string &id)

  Comprueba si tiene la imagen.
- void removeImage (const std::string &id)

  Elimina la imagen por el id especificado.
- void addFont (const std::string &id, const std::string &filename)

  A un tipo de letra.
- sf::Font & getFont (const std::string &id)

  Devuelve el tipo de letra.
- bool hasFont (const std::string &id)

  Comprueba si tiene la fuente especificada.
- void removeFont (const std::string &id)

  Elimina un tipo de letra.
- bool clear ()

  Descarga todos los recursos cargados.
- ~AssetManager ()

Private Attributes

- std::map< std::string,
  std::unique_ptr< sf::Texture > > textures
- std::map< std::string,
  std::unique_ptr< sf::Image > > images
- std::map< std::string,
  std::unique_ptr< sf::Font > > fonts

9.3.1 Detailed Description

Clase para tener todos los assets(imagenes,texturas,etc) del juego.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 20 of file AssetManager.hpp.

9.3.2 Constructor & Destructor Documentation

9.3.2.1 ant::AssetManager::AssetManager ( )

Constructor principal sin argumentos.

Definition at line 3 of file AssetManager.cpp.

9.3.2.2 ant::AssetManager::~AssetManager ( )

Definition at line 89 of file AssetManager.cpp.

9.3.3 Member Function Documentation

9.3.3.1 void ant::AssetManager::addFont (const std::string & *id*, const std::string & *filename*)

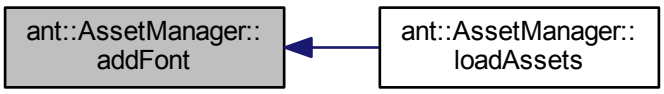
A un tipo de letra.

Parameters

id	std::string
filename	std::string

Definition at line 66 of file AssetManager.cpp.

Here is the caller graph for this function:



9.3.3.2 void ant::AssetManager::addImage (const std::string & *id*, const std::string & *filename*)

A una imagen con el id especificado.

Parameters

id	std::string id de la imagen.
filename	std::string imagen a cargar.

Definition at line 49 of file AssetManager.cpp.

Here is the caller graph for this function:

9.3.3.3 void ant::AssetManager::addTexture (const std::string & *id*, const std::string & *filename*)

A una textura con el id proporcionado.

Parameters

id	std::string Id de la textura cargada.
filename	std::string Textura a cargar.

Definition at line 32 of file AssetManager.cpp.

Here is the caller graph for this function:

9.3.3.4 bool ant::AssetManager::clear ( )

Descarga todos los recursos cargados.

Returns

true si los descargo correctamente, false si no.

Definition at line 83 of file AssetManager.cpp.

9.3.3.5 sf::Font & ant::AssetManager::getFont (const std::string & *id*)

Devuelve el tipo de letra.

Parameters

id	std::string

Definition at line 73 of file AssetManager.cpp.

9.3.3.6 sf::Image & ant::AssetManager::getImage (const std::string & *id*)

Devuelve una referencia a la imagen, si no existe tira excepcion.

Parameters

id	std::string id de la imagen.

Returns

sf::Image& referencia a imagen.

Definition at line 56 of file AssetManager.cpp.

9.3.3.7 sf::Texture & ant::AssetManager::getTexture (const std::string & *id*)

Devuelve una referencia a la textura, si no existe, tira excepcion de fuera de rango.

Parameters

id	std::string id de la textura.

Returns

sf::Texture& referencia de la textura.

Definition at line 39 of file AssetManager.cpp.

9.3.3.8 bool ant::AssetManager::hasFont (const std::string & *id*)

Comprueba si tiene la fuente especificada.

Parameters

id	const std::string & id de la fuente

Definition at line 76 of file AssetManager.cpp.

9.3.3.9 bool ant::AssetManager::hasImage (const std::string & *id*)

Comprueba si tiene la imagen.

Parameters

id	std::string & id de la imagen.

Definition at line 59 of file AssetManager.cpp.

9.3.3.10 bool ant::AssetManager::hasTexture (const std::string & *id*)

Comprueba si tiene la textura.

Parameters

id	const std::string & id de la textura.

Definition at line 42 of file AssetManager.cpp.

9.3.3.11 bool ant::AssetManager::loadAssets (const std::string & *filename*)

Carga un fichero json con el siguiente formato:

```
{
     "textures":[ {"id":"TEXTURE_ID","filepath":"data/entities/Example.png"},
                ],
     "images":[ {"id":"IMAGE_ID","filepath":"data/entities/Example.png"},
              ],
}
```

Parameters

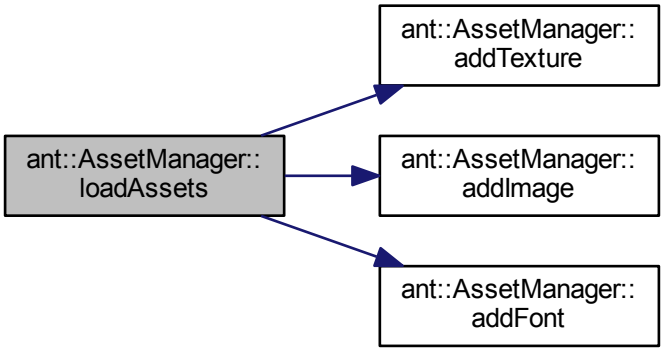
filename	std::string fichero a cargar.

Returns

true si ha cargado los datos correctamente, false si hay algun problema.

Definition at line 6 of file AssetManager.cpp.

Here is the call graph for this function:



9.3.3.12 void ant::AssetManager::removeFont (const std::string & *id*)

Elimina un tipo de letra.

Parameters

id	std::string

Definition at line 80 of file AssetManager.cpp.

9.3.3.13 void ant::AssetManager::removeImage (const std::string & *id*)

Elimina la imagen por el id especificado.

Parameters

id	std::string id de la imagen a eliminar.

Definition at line 63 of file AssetManager.cpp.

9.3.3.14 void ant::AssetManager::removeTexture (const std::string & *id*)

Elimina la textura.

Parameters

id	std::string id de la textura a eliminar.

Definition at line 46 of file AssetManager.cpp.

9.3.4 Member Data Documentation

9.3.4.1 std::map<std::string,std::unique_ptr<sf::Font> > ant::AssetManager::fonts [private]

Definition at line 119 of file AssetManager.hpp.

9.3.4.2 std::map<std::string,std::unique_ptr<sf::Image> > ant::AssetManager::images [private]

Definition at line 118 of file AssetManager.hpp.

9.3.4.3 std::map<std::string,std::unique_ptr<sf::Texture> > ant::AssetManager::textures [private]

Definition at line 117 of file AssetManager.hpp.

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

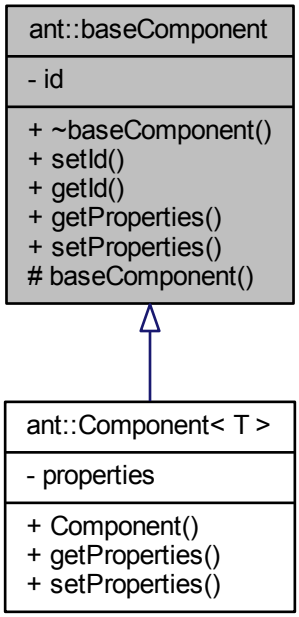
- AssetManager.hpp
- AssetManager.cpp

9.4 ant::baseComponent Class Reference

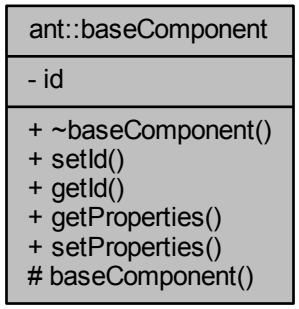
Clase base de componente.

```
#include <Component.hpp>
```

Inheritance diagram for ant::baseComponent:



Collaboration diagram for ant::baseComponent:



Public Member Functions

- virtual ~baseComponent ()=default
- void setId (long int id)

  Setter para el id.
- const long int & getId () const

  Getter para el id.

- template<typename... T>
  std::tuple< T...> & getProperties ()
  Getter de las propiedades del componente,.
- template<typename... T>
  void setProperties (std::tuple< T...> properties)
  Setter de las propiedades del componente.

Protected Member Functions

- baseComponent (long int id)
  Contructor para la clase derivada.

Private Attributes

- long int id

9.4.1 Detailed Description

Clase base de componente.

Author

Cristian Gonzalez Moreno `cristian.glez.m@gmail.com`

Version

0.1

Definition at line 14 of file Component.hpp.

9.4.2 Constructor & Destructor Documentation

9.4.2.1 ant::baseComponent::baseComponent (long int *id*) [inline],[protected]

Contructor para la clase derivada.

Todo posible cambio en el id de long int a bitset<NBits>

Parameters

id	long int id del componente.

Definition at line 23 of file Component.hpp.

9.4.2.2 virtual ant::baseComponent::~baseComponent ( ) [virtual],[default]

9.4.3 Member Function Documentation

9.4.3.1 const long int& ant::baseComponent::getId ( ) const [inline]

Getter para el id.

Returns

long int id del componente.

Definition at line 35 of file Component.hpp.

9.4.3.2 template<typename... T> std::tuple< T...> & ant::baseComponent::getProperties ( )

Getter de las propiedades del componente,.

Es necesario especificar los datos que tiene el componente en el orden adecuado.

Template Parameters

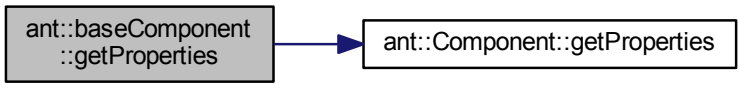
...*T*	tipo de datos que contiene el componente en su orden adecuado. `getProperties<float,int,std::string>();`

Returns

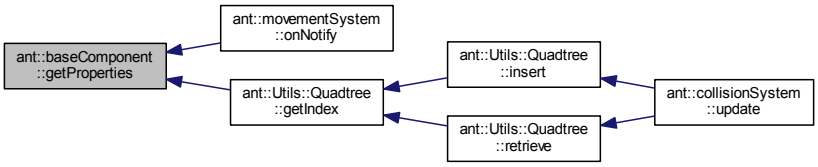
std::tuple<T...> la tupla con los datos.

Definition at line 115 of file Component.hpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.4.3.3 void ant::baseComponent::setId (long int *id*) [inline]

Setter para el id.

Parameters

id	long int id para el componente.

Definition at line 30 of file Component.hpp.

9.4.3.4 template<typename... T> void ant::baseComponent::setProperties (std::tuple< T...> *properties*)

Setter de las propiedades del componente.

Template Parameters

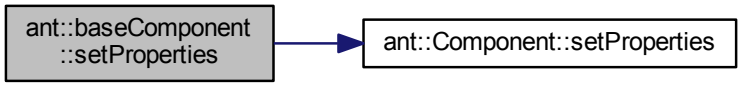
...T	tipo de datos que contiene el componente en su orden adecuado.

Parameters

properties	...T tipo de datos `setProperties<float,int,int>(std::tuple<float,int,int>(2.4,5,6));`

Definition at line 109 of file Component.hpp.

Here is the call graph for this function:



9.4.4 Member Data Documentation

9.4.4.1 long int ant::baseComponent::id [private]

Definition at line 60 of file Component.hpp.

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

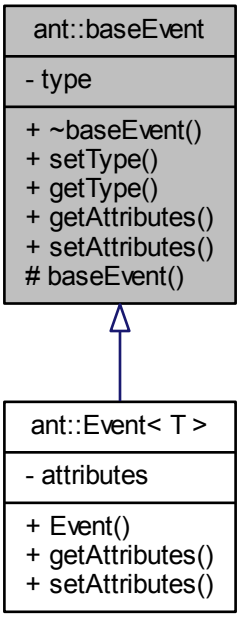
- Component.hpp

9.5 ant::baseEvent Class Reference

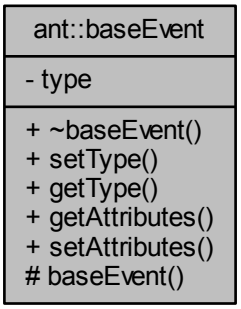
Clase Base de eventos.

`#include <Event.hpp>`

Inheritance diagram for ant::baseEvent:



Collaboration diagram for ant::baseEvent:



Public Member Functions

- virtual ~baseEvent ()=default
- void setType (int type)

  Setter para poner el tipo al evento.
- int getType () const

  Getter del tipo de evento.

- template<typename... T>
  const std::tuple< T...> & getAttributes () const

  Getter de los atributos del evento.
- template<typename... T>
  void setAttributes (std::tuple< T...> attributes)

  Setter de los atributos del evento.

Protected Member Functions

- baseEvent (int type)

  Constructor protegido del evento base.

Private Attributes

- int type

9.5.1 Detailed Description

Clase Base de eventos.

Author

Cristian Gonzalez Moreno `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 10 of file Event.hpp.

9.5.2 Constructor & Destructor Documentation

9.5.2.1 ant::baseEvent::baseEvent (int *type*) [inline],[protected]

Constructor protegido del evento base.

Parameters

type	int Tipo de evento.

Definition at line 16 of file Event.hpp.

9.5.2.2 virtual ant::baseEvent::~baseEvent ( ) [virtual],[default]

9.5.3 Member Function Documentation

9.5.3.1 template<typename... T> const std::tuple< T...> & ant::baseEvent::getAttributes ( ) const

Getter de los atributos del evento.

```
baseEvent::getAttributes<float,int>();
```

Template Parameters

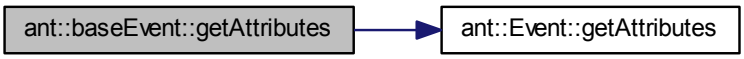
...*T*	tipo de datos que tiene el evento en el orden adecuado.

Returns

std::tuple<T...> & los attributos del evento(solo lectura)

Definition at line 90 of file Event.hpp.

Here is the call graph for this function:



9.5.3.2 int ant::baseEvent::getType ( ) const [inline]

Getter del tipo de evento.

Returns

int tipo del evento.

Definition at line 28 of file Event.hpp.

9.5.3.3 template<typename... T> void ant::baseEvent::setAttributes (std::tuple< T...> *attributes*)

Setter de los atributos del evento.

```
baseEvent::setAttributes<float,int>(5.3,5);
```

Template Parameters

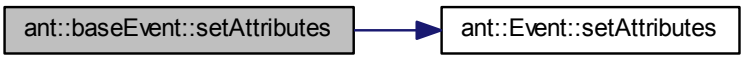
...*T*	tipo de datos que tiene el evento en el orden adecuado.

Parameters

attributes	...T los parametros de los atributos.

Definition at line 85 of file Event.hpp.

Here is the call graph for this function:



9.5.3.4 void ant::baseEvent::setType (int *type*) [inline]

Setter para poner el tipo al evento.

Parameters

type	int tipo de evento.

Definition at line 23 of file Event.hpp.

9.5.4 Member Data Documentation

9.5.4.1 int ant::baseEvent::type [private]

Definition at line 46 of file Event.hpp.

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

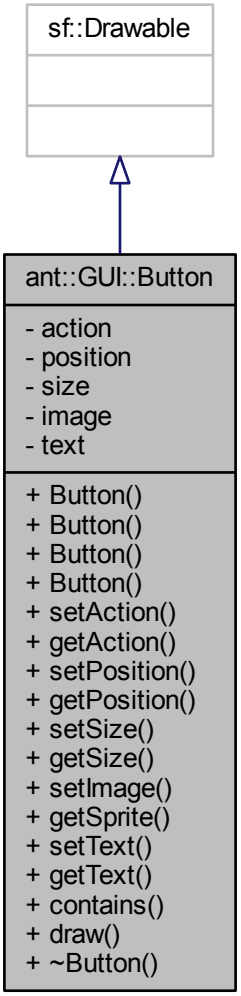
- Event.hpp

9.6 ant::GUI::Button Class Reference

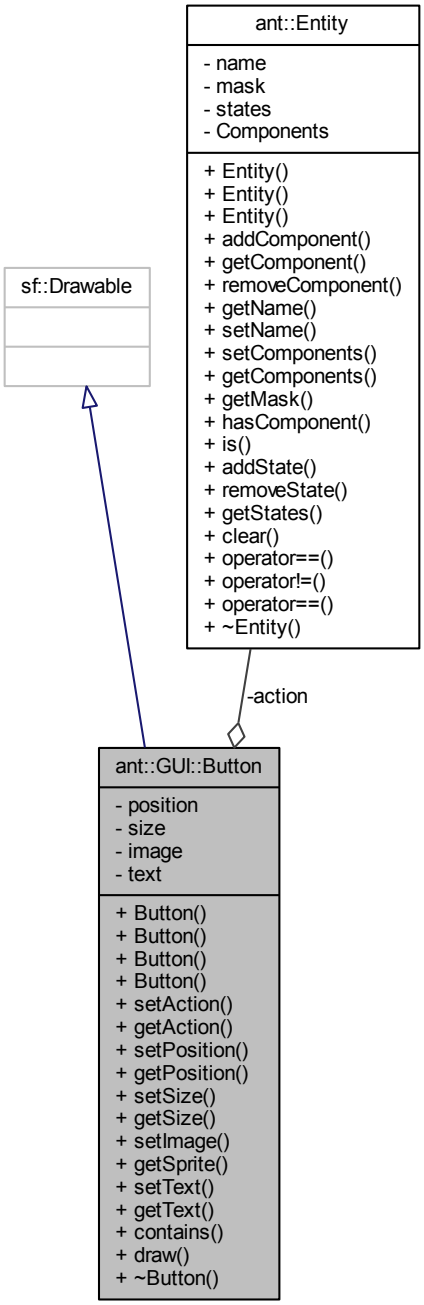
Clase de un simple boton.

```
#include <Button.hpp>
```

Inheritance diagram for ant::GUI::Button:



Collaboration diagram for ant::GUI::Button:



Public Member Functions

- Button ()
- Button (sf::Vector2f position, sf::Vector2f size, sf::Text text, constructorSystem::command action)
- Button (sf::Vector2f position, sf::Vector2f size, sf::Sprite image, constructorSystem::command action)
- Button (sf::Vector2f position, sf::Vector2f size, sf::Sprite image, sf::Text text, constructorSystem::command action)
- void setAction (constructorSystem::command action)

- constructorSystem::command getAction ()
- void setPosition (sf::Vector2f pos)
- sf::Vector2f getPosition ()
- void setSize (sf::Vector2f size)
- sf::Vector2f getSize ()
- void setImage (sf::Sprite sprite)
- sf::Sprite getSprite ()
- void setText (sf::Text text)
- sf::Text getText ()
- bool contains (sf::Vector2f point)
- virtual void draw (sf::RenderTarget &target, sf::RenderStates states) const
- ~Button ()

Private Attributes

- constructorSystem::command action
- sf::Vector2f position
- sf::Vector2f size
- sf::Sprite image
- sf::Text text

9.6.1 Detailed Description

Clase de un simple boton.

Author

Cristian Glez cristian.glez.m@gmail.com

Version

0.1

Definition at line 16 of file Button.hpp.

9.6.2 Constructor & Destructor Documentation

9.6.2.1 ant::GUI::Button::Button ( )

Definition at line 5 of file Button.cpp.

9.6.2.2 ant::GUI::Button::Button (sf::Vector2f *position*, sf::Vector2f *size*, sf::Text *text*, constructorSystem::command *action*)

Definition at line 10 of file Button.cpp.

9.6.2.3 ant::GUI::Button::Button (sf::Vector2f *position*, sf::Vector2f *size*, sf::Sprite *image*, constructorSystem::command *action*)

Definition at line 16 of file Button.cpp.

9.6.2.4 ant::GUI::Button::Button (sf::Vector2f *position,* sf::Vector2f *size,* sf::Sprite *image,* sf::Text *text,* constructorSystem::command *action*)

Definition at line 22 of file Button.cpp.

9.6.2.5 ant::GUI::Button::~Button ( )

Definition at line 52 of file Button.cpp.

9.6.3 Member Function Documentation

9.6.3.1 bool ant::GUI::Button::contains (sf::Vector2f *point*)

Definition at line 44 of file Button.cpp.

9.6.3.2 void ant::GUI::Button::draw (sf::RenderTarget & *target,* sf::RenderStates *states*) const [virtual]

Definition at line 48 of file Button.cpp.

9.6.3.3 constructorSystem::command ant::GUI::Button::getAction ( ) [inline]

Definition at line 23 of file Button.hpp.

9.6.3.4 sf::Vector2f ant::GUI::Button::getPosition ( ) [inline]

Definition at line 25 of file Button.hpp.

9.6.3.5 sf::Vector2f ant::GUI::Button::getSize ( ) [inline]

Definition at line 27 of file Button.hpp.

9.6.3.6 sf::Sprite ant::GUI::Button::getSprite ( ) [inline]

Definition at line 29 of file Button.hpp.

9.6.3.7 sf::Text ant::GUI::Button::getText ( ) [inline]

Definition at line 31 of file Button.hpp.

9.6.3.8 void ant::GUI::Button::setAction (constructorSystem::command *action*)

Definition at line 29 of file Button.cpp.

9.6.3.9 void ant::GUI::Button::setImage (sf::Sprite *sprite*)

Definition at line 38 of file Button.cpp.

9.6.3.10 void ant::GUI::Button::setPosition (sf::Vector2f *pos*)

Definition at line 32 of file Button.cpp.

9.6.3.11 void ant::GUI::Button::setSize (sf::Vector2f *size*)

Definition at line 35 of file Button.cpp.

9.6.3.12 void ant::GUI::Button::setText (sf::Text *text*)

Definition at line 41 of file Button.cpp.

9.6.4 Member Data Documentation

9.6.4.1 constructorSystem::command ant::GUI::Button::action [private]

Definition at line 36 of file Button.hpp.

9.6.4.2 sf::Sprite ant::GUI::Button::image [private]

Definition at line 39 of file Button.hpp.

9.6.4.3 sf::Vector2f ant::GUI::Button::position [private]

Definition at line 37 of file Button.hpp.

9.6.4.4 sf::Vector2f ant::GUI::Button::size [private]

Definition at line 38 of file Button.hpp.

9.6.4.5 sf::Text ant::GUI::Button::text [private]

Definition at line 40 of file Button.hpp.

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

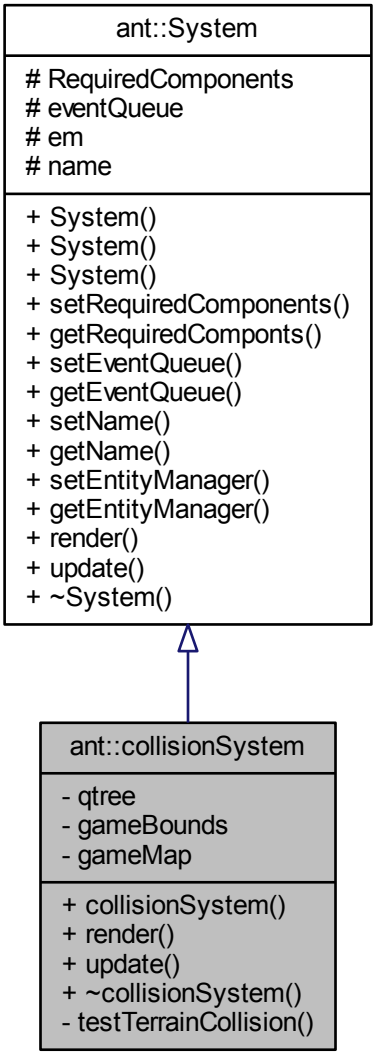
- Button.hpp
- Button.cpp

9.7 ant::collisionSystem Class Reference

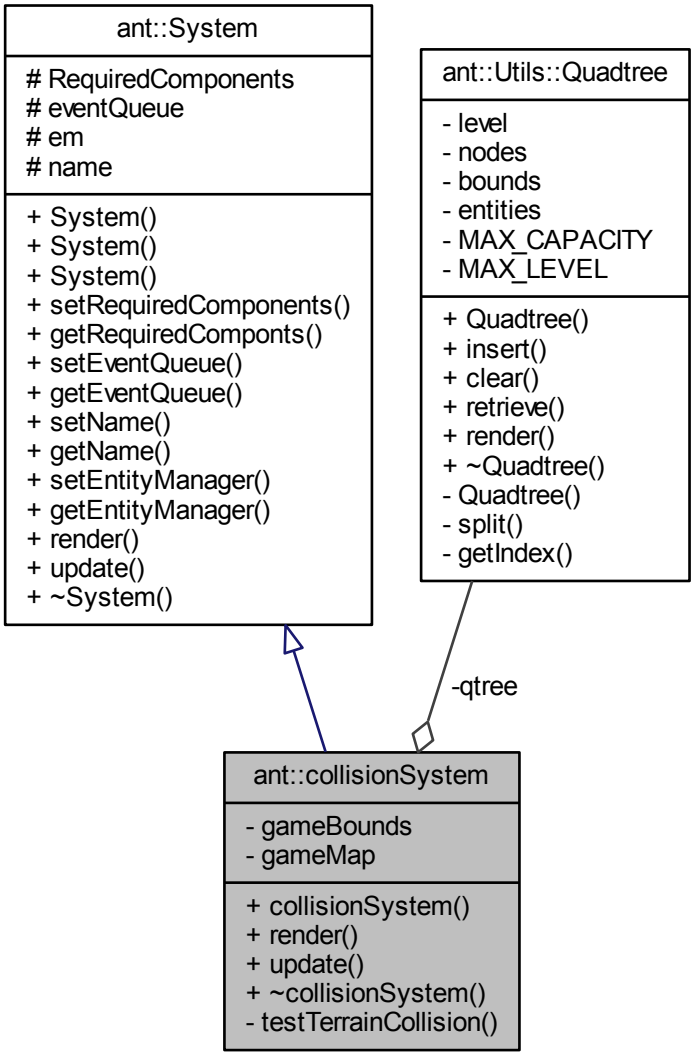
Sistema que comprueba las colisiones entre entidades.

```
#include <collisionSystem.hpp>
```

Inheritance diagram for ant::collisionSystem:



Collaboration diagram for ant::collisionSystem:



Public Member Functions

- collisionSystem (sf::FloatRect bounds, sf::VertexArray ∗GameMap)

  Constructor por defecto.
- virtual void render (sf::RenderWindow &win)

  Renderiza el Quadtree si RENDER_QTREE esta definido. si no no hace nada.
- virtual void update (sf::Time dt)

  Comprueba las colisiones de las entidades.
- virtual ∼collisionSystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.

- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

Private Member Functions

- void testTerrainCollision (Entity *entity, const sf::FloatRect &eBounds)

  Comprueba si una entidad y sus bordes estan dentro del mapa.

Private Attributes

- Utils::Quadtree qtree
- sf::FloatRect gameBounds
- sf::VertexArray * gameMap

9.7.1 Detailed Description

Sistema que comprueba las colisiones entre entidades.

Cuando detecta una colision a un evento a la EventQueue con las dos entidades que colisionaron.

1. Event<Entity*,Entity*> -> type: COLLISION_EVENT
2. Event<Entity*,int> -> type: TERRAIN_COLLISION
3. Event<Entity*> -> type: OUT_MAP

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 25 of file collisionSystem.hpp.

9.7.2 Constructor & Destructor Documentation

9.7.2.1 ant::collisionSystem::collisionSystem (sf::FloatRect *bounds,* sf::VertexArray * *GameMap*)

Constructor por defecto.

Parameters

bounds	sf::FloatRect rectangulo del QuadTree.
GameMap	sf::VertexArray * Terreno del juego.

Definition at line 4 of file collisionSystem.cpp.

9.7.2.2 ant::collisionSystem::~collisionSystem ( ) [virtual]

Definition at line 105 of file collisionSystem.cpp.

9.7.3 Member Function Documentation

9.7.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.7.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.7.3.3 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.7.3.4 long int ant::System::getRequiredComponts ( ) [inline],[inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

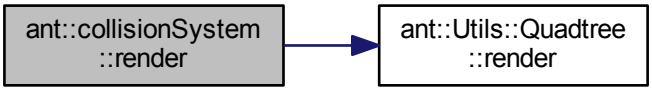
9.7.3.5 virtual void ant::collisionSystem::render (sf::RenderWindow & *win*) [inline],[virtual]

Renderiza el Quadtree si RENDER_QTREE esta definido. si no no hace nada.

Implements ant::System.

Definition at line 37 of file collisionSystem.hpp.

Here is the call graph for this function:



9.7.3.6 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.7.3.7 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.7.3.8 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.7.3.9 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.7.3.10 void ant::collisionSystem::testTerrainCollision (Entity * *entity*, const sf::FloatRect & *eBounds*) [private]

Comprueba si una entidad y sus bordes estan dentro del mapa.

Envia un evento para decir por donde a colisionado la entidad con el suelo.

1 esquina superior izquierda -> envia evento de colision con pared 0 2 esquina superior derecha -> envia evento de colision con pared 1 3 esquina inferior mediana izquierda -> envia evento de colision con escalon para subirlo 2 4 esquina inferior izquierda -> Cambia estado de FALLING a GROUND 5 esquina inferior mediana derecha -> envia evento de colision con escalon para subirlo 2 6 esquina inferior derecha -> Cambia estado de FALLING a GROUND

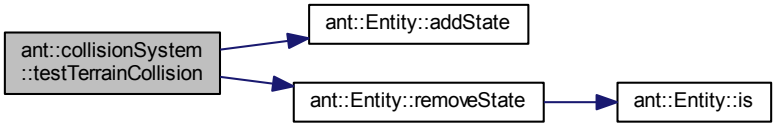
1. Event<Entity∗,int> -> type: TERRAIN_COLLISION

Parameters

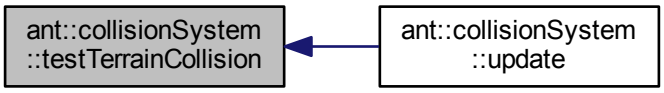
entity	Entity ∗ puntero de la entidad
eBounds	sf::FloatRect bordes de la entidad

Definition at line 64 of file collisionSystem.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.7.3.11 void ant::collisionSystem::update (sf::Time *dt*) [virtual]

Comprueba las colisiones de las entidades.

Envia eventos de colision y eventos especificos del suelo.

1. Event<Entity∗,Entity∗> -> type: COLLISION_EVENT
2. Event<Entity∗,int> -> type: TERRAIN_COLLISION
3. Event<Entity∗> -> type: OUT_MAP

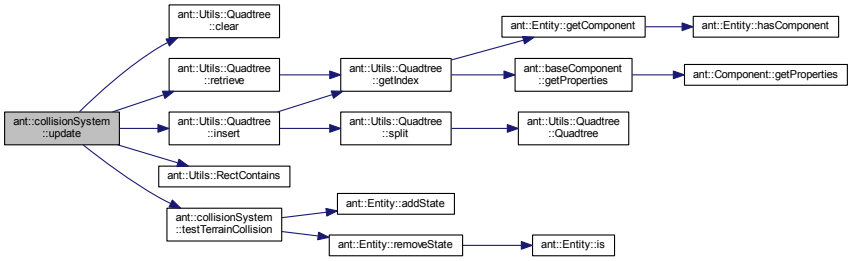
   Parameters

dt	sf::Time delta time

Implements ant::System.

Definition at line 11 of file collisionSystem.cpp.

Here is the call graph for this function:



9.7.4 Member Data Documentation

9.7.4.1 std::shared_ptr<EntityManager> ant::System::em [protected], [inherited]

Definition at line 86 of file System.hpp.

9.7.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected], [inherited]

Definition at line 85 of file System.hpp.

9.7.4.3 sf::FloatRect ant::collisionSystem::gameBounds [private]

Definition at line 75 of file collisionSystem.hpp.

9.7.4.4 sf::VertexArray* ant::collisionSystem::gameMap [private]

Definition at line 76 of file collisionSystem.hpp.

9.7.4.5 std::string ant::System::name [protected], [inherited]

Definition at line 87 of file System.hpp.

9.7.4.6 Utils::Quadtree ant::collisionSystem::qtree [private]

Definition at line 74 of file collisionSystem.hpp.

9.7.4.7 long int ant::System::RequiredComponents [protected], [inherited]

Definition at line 84 of file System.hpp.

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

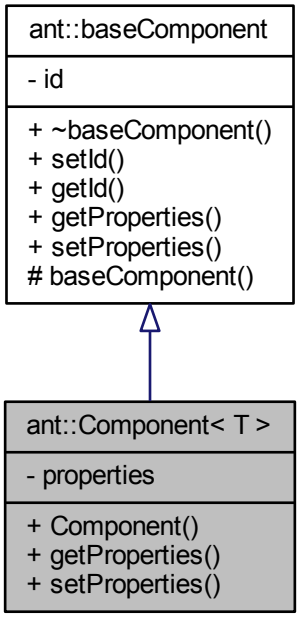
- collisionSystem.hpp
- collisionSystem.cpp

9.8 ant::Component< T > Class Template Reference

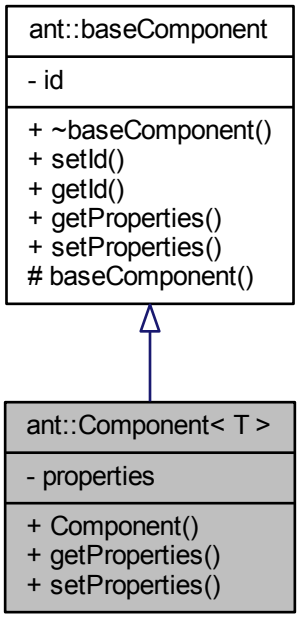
Clase especializada de baseComponent.

```
#include <Component.hpp>
```

Inheritance diagram for ant::Component< T >:



Collaboration diagram for ant::Component< T >:



Public Member Functions

- Component (long int id, T...data)

  Constructor del componente especializado.
- std::tuple< T...> & getProperties ()

  Getter de las propiedades del componente,.
- void setProperties (std::tuple< T...> properties)

  Setter de las propiedades del componente.
- void setId (long int id)

  Setter para el id.
- const long int & getId () const

  Getter para el id.

Private Attributes

- std::tuple< T...> properties

9.8.1 Detailed Description

template<typename... T>class ant::Component< T >

Clase especializada de baseComponent.

Puede tener todas las propiedades que se definan despues se tendran que acceder por el orden correcto.

Template Parameters

...*T*	tipo de datos que tendra este componente.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 73 of file Component.hpp.

9.8.2 Constructor & Destructor Documentation

9.8.2.1 template<typename... T> ant::Component< T >::Component (long int *id*, T... *data*) [inline]

Constructor del componente especializado.

Parameters

id	long int id del componente.
data	T... propiedades del componente.

Definition at line 81 of file Component.hpp.

9.8.3 Member Function Documentation

9.8.3.1 const long int& ant::baseComponent::getId ( ) const [inline],[inherited]

Getter para el id.

Returns

long int id del componente.

Definition at line 35 of file Component.hpp.

9.8.3.2 template<typename... T> std::tuple<T...>& ant::Component< T >::getProperties ( ) [inline]

Getter de las propiedades del componente,.

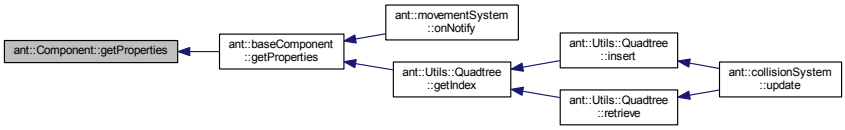
Cuando el Componente que tienes es una especializacion no hace falta especificar los componentes ya que estan definidos al crear el objecto.

Returns

std::tuple<T...> la tupla con los datos.

Definition at line 93 of file Component.hpp.

Here is the caller graph for this function:



9.8.3.3 void ant::baseComponent::setId (long int *id*) [inline],[inherited]

Setter para el id.

Parameters

id	long int id para el componente.

Definition at line 30 of file Component.hpp.

9.8.3.4 template<typename... T> void ant::Component< T >::setProperties (std::tuple< T...> *properties*) [inline]

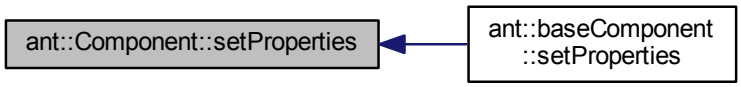
Setter de las propiedades del componente.

Parameters

properties	std::tuple<T...> tipo de datos
	
	`Component<float,int,int> c;`
	`c.setProperties(std::tuple<float,int,int>(2.4,5,6));`

Definition at line 103 of file Component.hpp.

Here is the caller graph for this function:



9.8.4 Member Data Documentation

9.8.4.1 template<typename... T> std::tuple<T...> ant::Component< T >::properties [private]

Definition at line 105 of file Component.hpp.

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

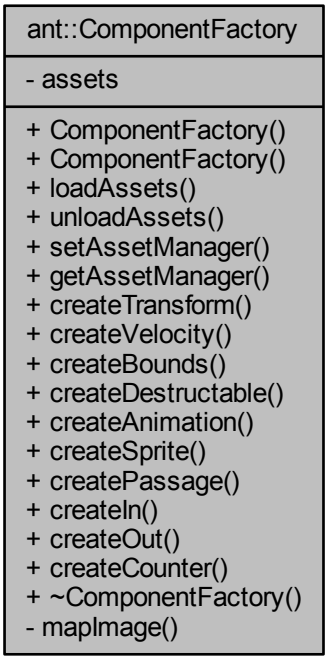
- Component.hpp

9.9 ant::ComponentFactory Class Reference

Clase para crear los componentes de las entidades.

`#include <ComponentFactory.hpp>`

Collaboration diagram for ant::ComponentFactory:



Public Member Functions

- ComponentFactory ()

  Constructor sin argumentos.
- ComponentFactory (std::shared_ptr< AssetManager > assets)

  Constructor que le pasa los assets.
- bool loadAssets (const std::string &json)

  Llama a la funcion de AssetsManager::loadAssets() para cargar los assets desde un fichero, tiene que tener el formato especificado en la funcion.
- bool unloadAssets ()

  Descarga los assets cargados.
- void setAssetManager (std::shared_ptr< AssetManager > assets)

  Pone el AssetManager especificado para usar, en la creacion de los componentes que requieran de assets.
- std::shared_ptr< AssetManager > getAssetManager ()

Devuelve el asset manager que utiliza.

- std::unique_ptr< baseComponent > createTransform (sf::Vector2f position, sf::Vector2f scale, float rotation)

  Crea un componente tipo Transform.
- std::unique_ptr< baseComponent > createVelocity (float speed, float minSpeed, float maxSpeed)

  Crea un componente Tipo Velocity.
- std::unique_ptr< baseComponent > createBounds (sf::FloatRect bounds)

  Crea un componente Tipo Bounds.
- std::unique_ptr< baseComponent > createDestructable (sf::Vector2f position, const std::string &imageID)

  Crea un componente tipo Destructable.
- std::unique_ptr< baseComponent > createAnimation (std::vector< std::string > ids)

  Crea un componente tipo Animation.
- std::unique_ptr< baseComponent > createSprite (std::string id)

  Crea un componente tipo Sprite.
- std::unique_ptr< baseComponent > createPassage (sf::Vector2f dest)

  Crea un componente tipo Passage.
- std::unique_ptr< baseComponent > createIn ()

  Crea un componente que indica la direccion.
- std::unique_ptr< baseComponent > createOut ()

  Crea un componente que indica la direccion.
- std::unique_ptr< baseComponent > createCounter (int count)

  Crea un componente que tiene un numero de veces.
- ~ComponentFactory ()

Private Member Functions

- std::unique_ptr< sf::VertexArray > mapImage (const sf::Vector2f &position, sf::Image &img, sf::FloatRect *bounds)

  Sirve para mapear una imagen a un sf::VertexArray.

Private Attributes

- std::shared_ptr< AssetManager > assets

9.9.1 Detailed Description

Clase para crear los componentes de las entidades.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file ComponentFactory.hpp.

9.9.2 Constructor & Destructor Documentation

9.9.2.1 ant::ComponentFactory::ComponentFactory ( )

Constructor sin argumentos.

Definition at line 29 of file ComponentFactory.cpp.

9.9.2.2 ant::ComponentFactory::ComponentFactory (std::shared_ptr< AssetManager > *assets*)

Constructor que le pasa los assets.

Parameters

assets	std::shared_ptr<AssetManager>

Definition at line 32 of file ComponentFactory.cpp.

9.9.2.3 ant::ComponentFactory::~ComponentFactory ( )

Definition at line 97 of file ComponentFactory.cpp.

9.9.3 Member Function Documentation

9.9.3.1 std::unique_ptr< baseComponent > ant::ComponentFactory::createAnimation (std::vector< std::string > *ids*)

Crea un componente tipo Animation.

Este componente basico es el cual permite hacer animaciones. necesita un sistema especifico por cada animacion.

```
Component<std::vector<std::string>>>
```

Parameters

ids	std::vector<std::string> id de las texturas para la animacion.

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 70 of file ComponentFactory.cpp.

9.9.3.2 std::unique_ptr< baseComponent > ant::ComponentFactory::createBounds (sf::FloatRect *bounds*)

Crea un componente Tipo Bounds.

Este componente basico es el cual permite a la entidad tener colisiones.

```
Component<sf::FloatReact>
```

Parameters

bounds	sf::FloatRect rectangulo de colision.

Returns

std::unique_ptr<baseComponent> componente Creado.

Definition at line 59 of file ComponentFactory.cpp.

9.9.3.3 std::unique_ptr< baseComponent > ant::ComponentFactory::createCounter (int *count*)

Crea un componente que tiene un numero de veces.

```
Component<int>
```

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 93 of file ComponentFactory.cpp.

9.9.3.4 std::unique_ptr< baseComponent > ant::ComponentFactory::createDestructable (sf::Vector2f *position*, const std::string & *imageID*)

Crea un componente tipo Destructable.

Este componente basico es el cual permite a la entidad ser destruida. Son pixeles individuales las operaciones de mover no se pueden aplicar sin tener un coste elevado, tarda mas segun lo grande que sea la imagen.

```
Component<std::string,sf::VertexArray>
```

Parameters

position	sf::Vector2f posicion en la que dibuja el primer pixel.
imageID	std::string& id de la imagen a mapear.

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 63 of file ComponentFactory.cpp.

Here is the call graph for this function:

9.9.3.5 std::unique_ptr< baseComponent > ant::ComponentFactory::createIn ( )

Crea un componente que indica la direccion.

Este componente permite hacer que una entidad haga aparecer otras entidades.

```
Component<>
```

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 85 of file ComponentFactory.cpp.

9.9.3.6 std::unique_ptr< baseComponent > ant::ComponentFactory::createOut ( )

Crea un componente que indica la direccion.

Este componente permite hacer que una entidad haga desparecer las entidades que colisiones contra ella.

```
Component<>
```

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 89 of file ComponentFactory.cpp.

9.9.3.7 std::unique_ptr< baseComponent > ant::ComponentFactory::createPassage (sf::Vector2f *dest*)

Crea un componente tipo Passage.

Este componente permite ir de un sitio a otro.

```
Component<sf::Vector2f>
```

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 81 of file ComponentFactory.cpp.

9.9.3.8 std::unique_ptr< baseComponent > ant::ComponentFactory::createSprite (std::string *id*)

Crea un componente tipo Sprite.

Este componente basico es el cual permite a una entidad ser renderizada en 2d.

```
Component<std::string,sf::Sprite>
```

Parameters

id	std::string id de la textura.

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 75 of file ComponentFactory.cpp.

9.9.3.9 std::unique_ptr< baseComponent > ant::ComponentFactory::createTransform (sf::Vector2f *position,* sf::Vector2f *scale,* float *rotation*)

Crea un componente tipo Transform.

Este componente basico es el cual permite a la entidad tener posicion. necesario para poder renderizar, moverse, etc.

```
Component<sf::Vector2f,sf::Vector2f,float>
```

Parameters

position	sf::Vector2f posicion de la entidad.
scale	sf::Vector2f escala de la entidad.
rotation	float rotacion de la entidad.

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 44 of file ComponentFactory.cpp.

9.9.3.10 std::unique_ptr< baseComponent > ant::ComponentFactory::createVelocity (float *speed,* float *minSpeed,* float *maxSpeed*)

Crea un componente Tipo Velocity.

Este componente basico es el cual permite a la entidad moverse. necesario para poder moverse.

```
Component<sf::Vector2f,float,float,float>
```

Parameters

speed	float Velocidad actual.
minSpeed	float Velocidad minima a la cual puede ir.
maxSpeed	float Velocidad Maxima a la cual puede ir.

Returns

std::unique_ptr<baseComponent> componente creado.

Definition at line 49 of file ComponentFactory.cpp.

9.9.3.11 std::shared_ptr<AssetManager> ant::ComponentFactory::getAssetManager ( ) [inline]

Devuelve el asset manager que utiliza.

Returns

std::shared_ptr<AssetManager> AssetManager que utiliza.

Definition at line 53 of file ComponentFactory.hpp.

9.9.3.12 bool ant::ComponentFactory::loadAssets (const std::string & *json*)

Llama a la funcion de AssetsManager::loadAssets() para cargar los assets desde un fichero, tiene que tener el formato especificado en la funcion.

Parameters

json	std::string fichero json a cargar.

Returns

bool

Definition at line 35 of file ComponentFactory.cpp.

9.9.3.13 std::unique_ptr< sf::VertexArray > ant::ComponentFactory::mapImage (const sf::Vector2f & *position*, sf::Image & *img*, sf::FloatRect * *bounds*) [private]

Sirve para mapear una imagen a un sf::VertexArray.

Se utiliza para mapear una imagen a un sf::VertexArray, si la posicion es diferente a 0 la array se recorrera otra vez para poner las posiciones de los pixels segun la posicion puesta.

Parameters

position	sf::Vector2f Donde empezara a poner la posicion de cada pixel.
img	sf::Image& referencia de la imagen a mapear.
bounds	* sf::FloatRect la posicion y tamal destuctible

Definition at line 4 of file ComponentFactory.cpp.

Here is the caller graph for this function:

9.9.3.14 void ant::ComponentFactory::setAssetManager (std::shared_ptr< AssetManager > *assets*)

Pone el AssetManager especificado para usar, en la creacion de los componentes que requieran de assets.

Parameters

assets	std::shared_ptr<AssetManager> Assets

Definition at line 41 of file ComponentFactory.cpp.

9.9.3.15 bool ant::ComponentFactory::unloadAssets ( )

Descarga los assets cargados.

Returns

bool

Definition at line 38 of file ComponentFactory.cpp.

9.9.4 Member Data Documentation

9.9.4.1 std::shared_ptr<AssetManager> ant::ComponentFactory::assets [private]

Definition at line 174 of file ComponentFactory.hpp.

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

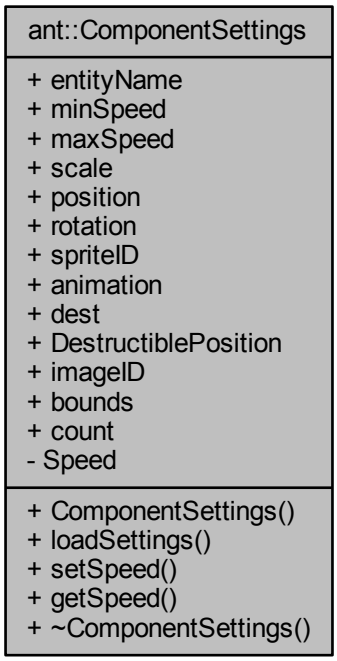
- ComponentFactory.hpp
- ComponentFactory.cpp

9.10 ant::ComponentSettings Class Reference

Clase para la configuracion de los componentes en EntityFactory.

```
#include <ComponentSettings.hpp>
```

Collaboration diagram for ant::ComponentSettings:



Public Member Functions

- ComponentSettings ()
- void loadSettings (const std::string &filename)

  Carga la configuracion de la entidad.
- void setSpeed (float speed)

  Establece la Velocidad.
- float getSpeed ()
- ~ComponentSettings ()

Public Attributes

- std::string entityName
- float minSpeed
- float maxSpeed
- sf::Vector2f scale
- sf::Vector2f position
- float rotation
- std::string spriteID
- std::vector< std::string > animation
- sf::Vector2f dest
- sf::Vector2f DestructiblePosition
- std::string imageID
- sf::FloatRect bounds
- int count

Private Attributes

- float Speed

9.10.1 Detailed Description

Clase para la configuracion de los componentes en EntityFactory.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 14 of file ComponentSettings.hpp.

9.10.2 Constructor & Destructor Documentation

9.10.2.1 ant::ComponentSettings::ComponentSettings ( )

Definition at line 4 of file ComponentSettings.cpp.

9.10.2.2 ant::ComponentSettings::~ComponentSettings ( )

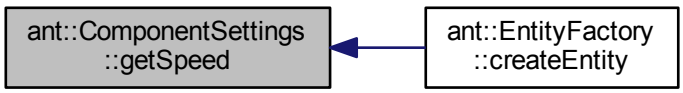
Definition at line 69 of file ComponentSettings.cpp.

9.10.3 Member Function Documentation

9.10.3.1 float ant::ComponentSettings::getSpeed ( ) [inline]

Definition at line 47 of file ComponentSettings.hpp.

Here is the caller graph for this function:



9.10.3.2 void ant::ComponentSettings::loadSettings (const std::string & *filename*)

Carga la configuracion de la entidad.

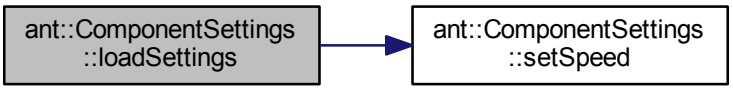
Carga un fichero json con las propiedades de los componentes las propiedades que no encuentre las pondra por defecto.

Parameters

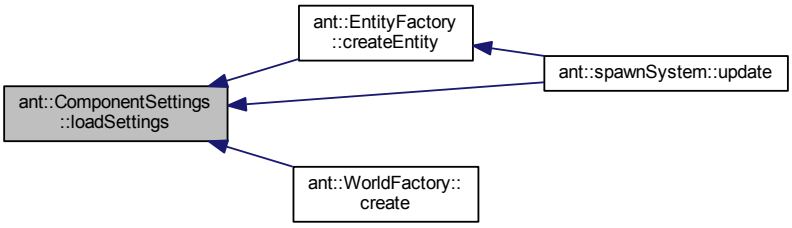
filename	std::string nombre del fichero(json) a cargar.

Definition at line 19 of file ComponentSettings.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.3.3 void ant::ComponentSettings::setSpeed (float *speed*)

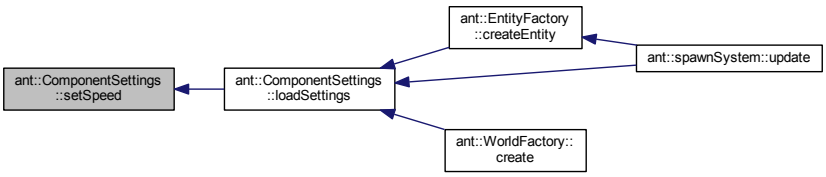
Establece la Velocidad.

Parameters

speed	float velocidad

Definition at line 61 of file ComponentSettings.cpp.

Here is the caller graph for this function:



9.10.4 Member Data Documentation

9.10.4.1 std::vector<std::string> ant::ComponentSettings::animation

Definition at line 23 of file ComponentSettings.hpp.

9.10.4.2 sf::FloatRect ant::ComponentSettings::bounds

Definition at line 27 of file ComponentSettings.hpp.

9.10.4.3 int ant::ComponentSettings::count

Definition at line 28 of file ComponentSettings.hpp.

9.10.4.4 sf::Vector2f ant::ComponentSettings::dest

Definition at line 24 of file ComponentSettings.hpp.

9.10.4.5 sf::Vector2f ant::ComponentSettings::DestructiblePosition

Definition at line 25 of file ComponentSettings.hpp.

9.10.4.6 std::string ant::ComponentSettings::entityName

Definition at line 16 of file ComponentSettings.hpp.

9.10.4.7 std::string ant::ComponentSettings::imageID

Definition at line 26 of file ComponentSettings.hpp.

9.10.4.8 float ant::ComponentSettings::maxSpeed

Definition at line 18 of file ComponentSettings.hpp.

9.10.4.9 float ant::ComponentSettings::minSpeed

Definition at line 17 of file ComponentSettings.hpp.

9.10.4.10 sf::Vector2f ant::ComponentSettings::position

Definition at line 20 of file ComponentSettings.hpp.

9.10.4.11 float ant::ComponentSettings::rotation

Definition at line 21 of file ComponentSettings.hpp.

9.10.4.12 sf::Vector2f ant::ComponentSettings::scale

Definition at line 19 of file ComponentSettings.hpp.

9.10.4.13 float ant::ComponentSettings::Speed [private]

Definition at line 50 of file ComponentSettings.hpp.

9.10.4.14 std::string ant::ComponentSettings::spriteID

Definition at line 22 of file ComponentSettings.hpp.

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

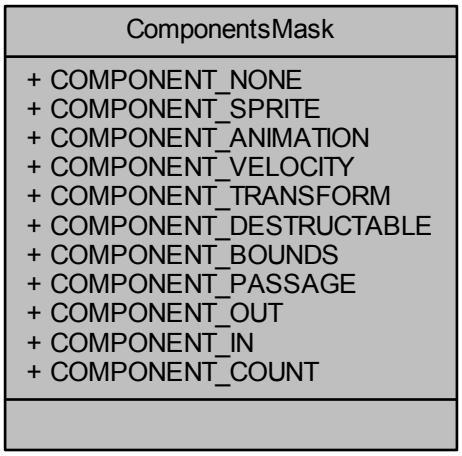
- ComponentSettings.hpp
- ComponentSettings.cpp

9.11 ComponentsMask Class Reference

tipos de componentes.

`#include <ComponentMask.hpp>`

Collaboration diagram for ComponentsMask:



Static Public Attributes

- static const long int COMPONENT_NONE = 0
- static const long int COMPONENT_SPRITE = 1 << 0
- static const long int COMPONENT_ANIMATION = 1 << 1
- static const long int COMPONENT_VELOCITY = 1 << 2
- static const long int COMPONENT_TRANSFORM = 1 << 3
- static const long int COMPONENT_DESTRUCTABLE = 1 << 4
- static const long int COMPONENT_BOUNDS = 1 << 5
- static const long int COMPONENT_PASSAGE = 1 << 6
- static const long int COMPONENT_OUT = 1 << 7
- static const long int COMPONENT_IN = 1 << 8
- static const long int COMPONENT_COUNT = 1 << 9

9.11.1 Detailed Description

tipos de componentes.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 9 of file ComponentMask.hpp.

9.11.2 Member Data Documentation

9.11.2.1 const long int ComponentsMask::COMPONENT_ANIMATION = 1 << 1 [static]

Definition at line 13 of file ComponentMask.hpp.

9.11.2.2 const long int ComponentsMask::COMPONENT_BOUNDS = 1 << 5 [static]

Definition at line 17 of file ComponentMask.hpp.

9.11.2.3 const long int ComponentsMask::COMPONENT_COUNT = 1 << 9 [static]

Definition at line 21 of file ComponentMask.hpp.

9.11.2.4 const long int ComponentsMask::COMPONENT_DESTRUCTABLE = 1 << 4 [static]

Definition at line 16 of file ComponentMask.hpp.

9.11.2.5 const long int ComponentsMask::COMPONENT_IN = 1 << 8 [static]

Definition at line 20 of file ComponentMask.hpp.

9.11.2.6 const long int ComponentsMask::COMPONENT_NONE = 0 [static]

Definition at line 11 of file ComponentMask.hpp.

9.11.2.7 const long int ComponentsMask::COMPONENT_OUT = 1 << 7 [static]

Definition at line 19 of file ComponentMask.hpp.

9.11.2.8 const long int ComponentsMask::COMPONENT_PASSAGE = 1 << 6 [static]

Definition at line 18 of file ComponentMask.hpp.

9.11.2.9 const long int ComponentsMask::COMPONENT_SPRITE = 1 << 0 [static]

Definition at line 12 of file ComponentMask.hpp.

9.11.2.10 const long int ComponentsMask::COMPONENT_TRANSFORM = 1 << 3 [static]

Definition at line 15 of file ComponentMask.hpp.

9.11.2.11 const long int ComponentsMask::COMPONENT_VELOCITY = 1 << 2 [static]

Definition at line 14 of file ComponentMask.hpp.

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

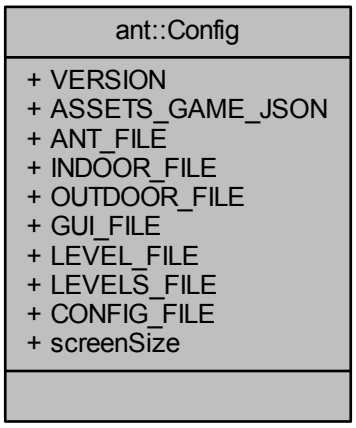
- ComponentMask.hpp

9.12 ant::Config Class Reference

Clase de configuracion con constantes de ficheros.

`#include <Config.hpp>`

Collaboration diagram for ant::Config:



Static Public Attributes

- static const std::string VERSION = "v0.1a"
- static const std::string ASSETS_GAME_JSON = "data/config/assets.json"
- static const std::string ANT_FILE = "data/config/entities/Ant.json"
- static const std::string INDOOR_FILE = "data/config/entities/InDoor.json"
- static const std::string OUTDOOR_FILE = "data/config/entities/OutDoor.json"
- static const std::string GUI_FILE = "data/config/GUI.json"
- static const std::string LEVEL_FILE = "data/config/entities/Level.json"
- static const std::string LEVELS_FILE = "data/config/levels.json"
- static const std::string CONFIG_FILE = "data/config/config.json"
- static const sf::FloatRect screenSize

9.12.1 Detailed Description

Clase de configuracion con constantes de ficheros.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 11 of file Config.hpp.

9.12.2 Member Data Documentation

9.12.2.1 const std::string ant::Config::ANT_FILE = "data/config/entities/Ant.json" [static]

Definition at line 16 of file Config.hpp.

9.12.2.2 const std::string ant::Config::ASSETS_GAME_JSON = "data/config/assets.json" [static]

Definition at line 15 of file Config.hpp.

9.12.2.3 const std::string ant::Config::CONFIG_FILE = "data/config/config.json" [static]

Definition at line 22 of file Config.hpp.

9.12.2.4 const std::string ant::Config::GUI_FILE = "data/config/GUI.json" [static]

Definition at line 19 of file Config.hpp.

9.12.2.5 const std::string ant::Config::INDOOR_FILE = "data/config/entities/InDoor.json" [static]

Definition at line 17 of file Config.hpp.

9.12.2.6 const std::string ant::Config::LEVEL_FILE = "data/config/entities/Level.json" [static]

Definition at line 20 of file Config.hpp.

9.12.2.7 const std::string ant::Config::LEVELS_FILE = "data/config/levels.json" [static]

Definition at line 21 of file Config.hpp.

9.12.2.8 const std::string ant::Config::OUTDOOR_FILE = "data/config/entities/OutDoor.json" [static]

Definition at line 18 of file Config.hpp.

9.12.2.9 const sf::FloatRect ant::Config::screenSize [static]

Definition at line 23 of file Config.hpp.

9.12.2.10 const std::string ant::Config::VERSION = "v0.1a" [static]

Definition at line 14 of file Config.hpp.

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

- Config.hpp
- Config.cpp

9.13 ant::Constructions Class Reference

Clase estatica con los comandos para las hormigas.

`#include <Constructions.hpp>`

Collaboration diagram for ant::Constructions:

Static Public Attributes

- static constructorSystem::command tunnel

  Construye un tunnel.
- static constructorSystem::command bridge

  Construye un puente.
- static constructorSystem::command wall

  Construye un muro para escalar o bloquar el paso.
- static constructorSystem::command stairs

  Construye unas escaleras.
- static constructorSystem::command hole

  Crea un agujero en el suelo.
- static constructorSystem::command climb

  la Hormiga empieza a escalar.

- static constructorSystem::command stop

  para la construccion / accion que este haciendo.
- static constructorSystem::command downhill

  Crea un tunel para abajo.
- static constructorSystem::command uphill

  Crea un tunel para arriba.
- static sf::Color Grey
- static sf::Color lightGrey
- static sf::Color Brown
- static sf::Color lightBrown

9.13.1 Detailed Description

Clase estatica con los comandos para las hormigas.

Author

Cristian Glez cristian.glez.m@gmail.com

Version

0.1

Definition at line 12 of file Constructions.hpp.

9.13.2 Member Data Documentation

9.13.2.1 constructorSystem::command ant::Constructions::bridge [static]

Construye un puente.

Definition at line 22 of file Constructions.hpp.

9.13.2.2 sf::Color ant::Constructions::Brown [static]

Definition at line 54 of file Constructions.hpp.

9.13.2.3 constructorSystem::command ant::Constructions::climb [static]

Initial value:

```
= [](Entity* e,sf::VertexArray* map,sf::FloatRect bounds){
        if(e->hasComponent(ComponentsMask::COMPONENT_COUNT)){
            auto& count = std::get<0>(e->getComponent(
      ComponentsMask::COMPONENT_COUNT)->getProperties<int>());
            if(count == 9){
                count = 20;
            }
            if(count <= 0){
                e->removeState(States::CLIMBING);
            }else{
                e->addState(States::CLIMBING);
            }
        }
    }
```

la Hormiga empieza a escalar.

Definition at line 38 of file Constructions.hpp.

9.13.2.4 constructorSystem::command ant::Constructions::downhill [static]

Crea un tunel para abajo.

Definition at line 46 of file Constructions.hpp.

9.13.2.5 sf::Color ant::Constructions::Grey [static]

Definition at line 52 of file Constructions.hpp.

9.13.2.6 constructorSystem::command ant::Constructions::hole [static]

Initial value:

```
= [](Entity* e,sf::VertexArray* map,sf::FloatRect bounds){
        if(e->hasComponent(ComponentsMask::COMPONENT_BOUNDS) && e->
      hasComponent(ComponentsMask::COMPONENT_TRANSFORM)){
            auto& cTrans = e->getComponent(ComponentsMask::COMPONENT_TRANSFORM
      )
                                ->getProperties<sf::Vector2f,sf::Vector2f,float>();
            auto& position = std::get<0>(cTrans);
            auto& eBounds = std::get<0>(e->getComponent(
      ComponentsMask::COMPONENT_BOUNDS)
                                                ->getProperties<sf::FloatRect>());
            sf::Vector2f currentPos = sf::Vector2f(position.x,
                                                   position.y + eBounds.height);
            if(Utils::RectContains(eBounds,Config::screenSize)){
                for(int x=currentPos.x-10;x<(currentPos.x+eBounds.width+10);++x){
                    for(int y=currentPos.y-1;y<(currentPos.y+10);++y){
                        if(Config::screenSize.contains(x,y)){
                            if((*map)[bounds.height * x + y].color.a == 255){
                                (*map)[bounds.height * x + y].color.a = 0;
                            }
                        }
                    }
                }
            }
        }
    }
```

Crea un agujero en el suelo.

Definition at line 34 of file Constructions.hpp.

9.13.2.7 sf::Color ant::Constructions::lightBrown [static]

Definition at line 55 of file Constructions.hpp.

9.13.2.8 sf::Color ant::Constructions::lightGrey [static]

Definition at line 53 of file Constructions.hpp.

9.13.2.9 constructorSystem::command ant::Constructions::stairs [static]

Construye unas escaleras.

Definition at line 30 of file Constructions.hpp.

9.13.2.10 constructorSystem::command ant::Constructions::stop [static]

Initial value:

```
= [](Entity* e,sf::VertexArray* map,sf::FloatRect bounds){
        if(e->hasComponent(ComponentsMask::COMPONENT_COUNT)){
            auto& count = std::get<0>(e->getComponent(
      ComponentsMask::COMPONENT_COUNT)->getProperties<int>());
            if(count <= 0){
                e->removeState(States::BUILDING);
                e->removeState(States::CLIMBING);
            }
        }
    }
```

para la construccion / accion que este haciendo.

Definition at line 42 of file Constructions.hpp.

9.13.2.11 constructorSystem::command ant::Constructions::tunnel [static]

Construye un tunnel.

Definition at line 18 of file Constructions.hpp.

9.13.2.12 constructorSystem::command ant::Constructions::uphill [static]

Crea un tunel para arriba.

Definition at line 50 of file Constructions.hpp.

9.13.2.13 constructorSystem::command ant::Constructions::wall [static]

Construye un muro para escalar o bloquar el paso.

Definition at line 26 of file Constructions.hpp.

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

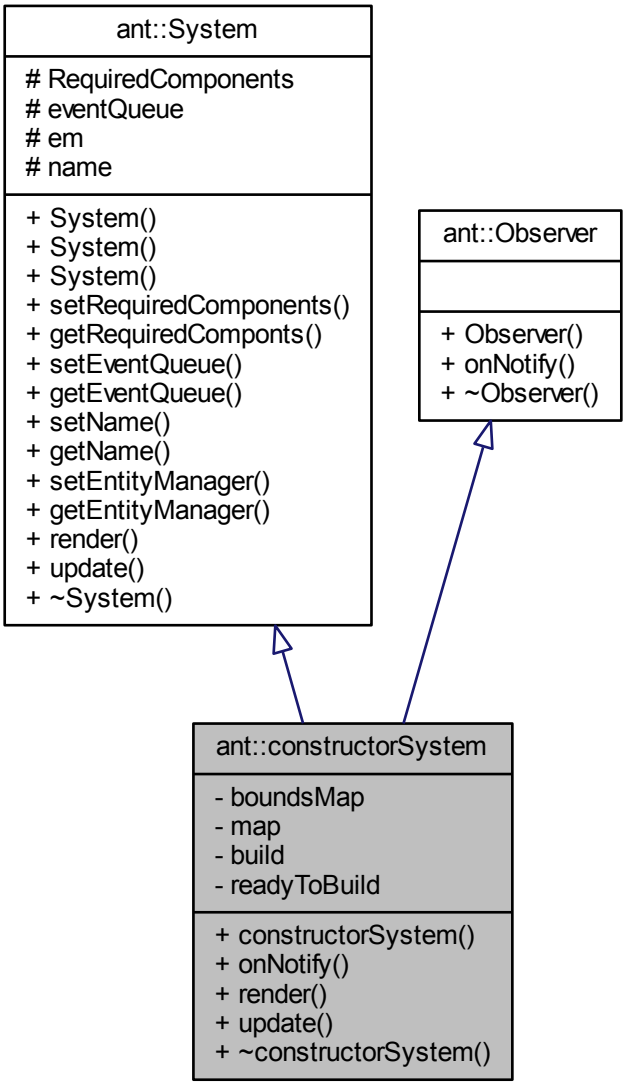
- Constructions.hpp
- Constructions.cpp

9.14 ant::constructorSystem Class Reference

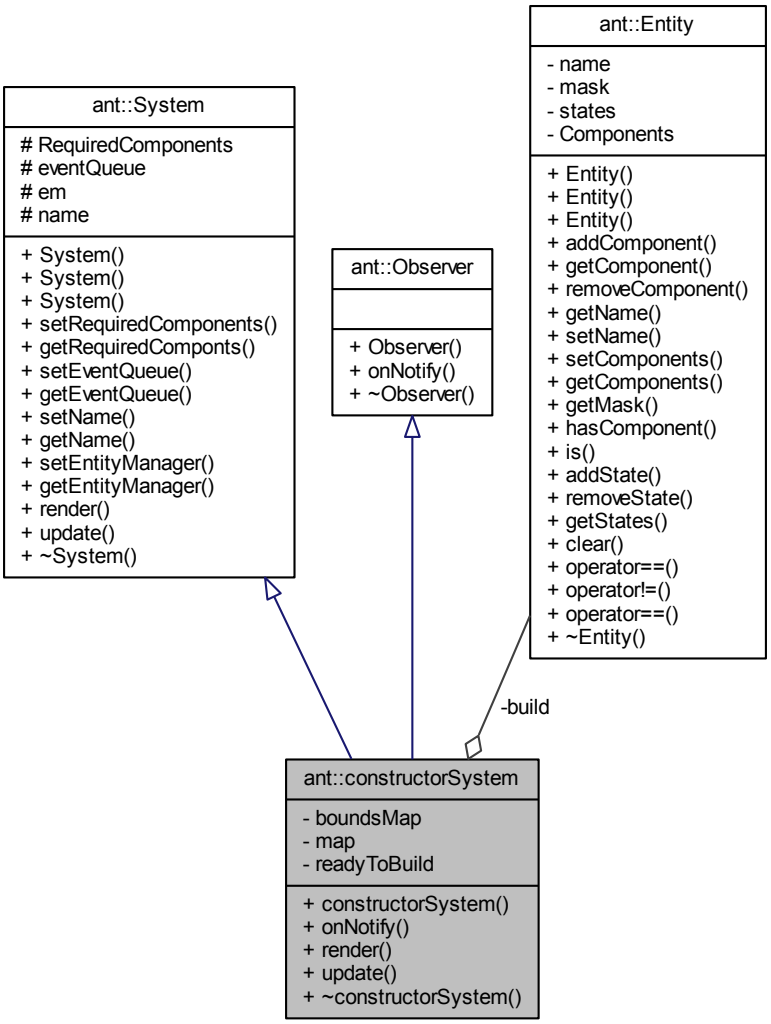
Sistema encargado de que las hormigas construyan etc.

`#include <constructorSystem.hpp>`

Inheritance diagram for ant::constructorSystem:



Collaboration diagram for ant::constructorSystem:



Public Types

- typedef void(∗ command)(Entity ∗, sf::VertexArray ∗, sf::FloatRect)

  Comandos que recibe por eventos.

Public Member Functions

- constructorSystem (sf::VertexArray ∗map, sf::FloatRect boundsMap)

  Constructor por defecto.
- virtual void onNotify (std::shared_ptr< baseEvent > e)

  recibe dos eventos
- virtual void render (sf::RenderWindow &win)

  No hace nada.
- virtual void update (sf::Time dt)

Gestiona la construccion de los elementos.

- ~constructorSystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

Private Attributes

- sf::FloatRect boundsMap
- sf::VertexArray * map
- command build
- bool readyToBuild

9.14.1 Detailed Description

Sistema encargado de que las hormigas construyan etc.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 17 of file constructorSystem.hpp.

9.14.2 Member Typedef Documentation

9.14.2.1 typedef void(∗ ant::constructorSystem::command)(Entity ∗, sf::VertexArray ∗, sf::FloatRect)

Comandos que recibe por eventos.

```
Event<command>(EventType::CHANGE_COMMAND,[](Entity* e,sf::VertexArray*
      map, sf::FloatRect mapBounds){
                // do something
                });
```

Definition at line 28 of file constructorSystem.hpp.

9.14.3 Constructor & Destructor Documentation

9.14.3.1 ant::constructorSystem::constructorSystem (sf::VertexArray ∗ *map,* sf::FloatRect *boundsMap*)

Constructor por defecto.

Parameters

map	sf::VertexArray ∗ Destructable para construir, etc.
boundsMap	sf::FloatRect bordes del destructable.

Definition at line 4 of file constructorSystem.cpp.

9.14.3.2 ant::constructorSystem::∼constructorSystem ( )

Definition at line 62 of file constructorSystem.cpp.

9.14.4 Member Function Documentation

9.14.4.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.14.4.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.14.4.3 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.14.4.4 long int ant::System::getRequiredComponts ( ) [inline], [inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.14.4.5 void ant::constructorSystem::onNotify (std::shared_ptr< baseEvent > *e*) [virtual]

recibe dos eventos

Eventos:

1. EventType::CHANGE_COMMAND Comando que cambiara la construccion que haran las entidades.
2. EventType::SELECT_ENTITY Seleccionara una entidad para aplicarle el comando actual.

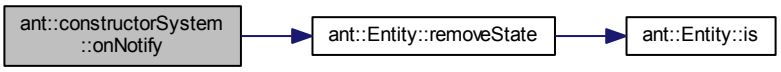
Parameters

e	std::shared_ptr<baseEvent> evento

Implements ant::Observer.

Definition at line 10 of file constructorSystem.cpp.

Here is the call graph for this function:



9.14.4.6 void ant::constructorSystem::render (sf::RenderWindow & *win*) [virtual]

No hace nada.

Implements ant::System.

Definition at line 59 of file constructorSystem.cpp.

9.14.4.7 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.14.4.8 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.14.4.9 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.14.4.10 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.14.4.11 void ant::constructorSystem::update (sf::Time *dt*) [virtual]

Gestiona la construccion de los elementos.

Parameters

dt	sf::Time delta time

Implements ant::System.

Definition at line 41 of file constructorSystem.cpp.

9.14.5 Member Data Documentation

9.14.5.1 sf::FloatRect ant::constructorSystem::boundsMap [private]

Definition at line 55 of file constructorSystem.hpp.

9.14.5.2 command ant::constructorSystem::build [private]

Definition at line 57 of file constructorSystem.hpp.

9.14.5.3 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.14.5.4 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.14.5.5 sf::VertexArray* ant::constructorSystem::map [private]

Definition at line 56 of file constructorSystem.hpp.

9.14.5.6 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.14.5.7 bool ant::constructorSystem::readyToBuild [private]

Definition at line 58 of file constructorSystem.hpp.

9.14.5.8 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

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

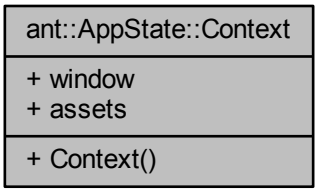
- constructorSystem.hpp
- constructorSystem.cpp

9.15 ant::AppState::Context Struct Reference

Clase para compartir datos esenciales a los estados.

`#include <AppState.hpp>`

Collaboration diagram for ant::AppState::Context:



Public Member Functions

- Context (std::shared_ptr< sf::RenderWindow > window, std::shared_ptr< AssetManager > assets)

  Constructor Principal.

Public Attributes

- std::shared_ptr< sf::RenderWindow > window
- std::shared_ptr< AssetManager > assets

9.15.1 Detailed Description

Clase para compartir datos esenciales a los estados.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Definition at line 31 of file AppState.hpp.

9.15.2 Constructor & Destructor Documentation

9.15.2.1 ant::AppState::Context::Context (std::shared_ptr< sf::RenderWindow > *window,* std::shared_ptr< AssetManager > *assets*)

Constructor Principal.

Definition at line 4 of file AppState.cpp.

9.15.3 Member Data Documentation

9.15.3.1 std::shared_ptr<AssetManager> ant::AppState::Context::assets

Definition at line 37 of file AppState.hpp.

9.15.3.2 std::shared_ptr<sf::RenderWindow> ant::AppState::Context::window

Definition at line 36 of file AppState.hpp.

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

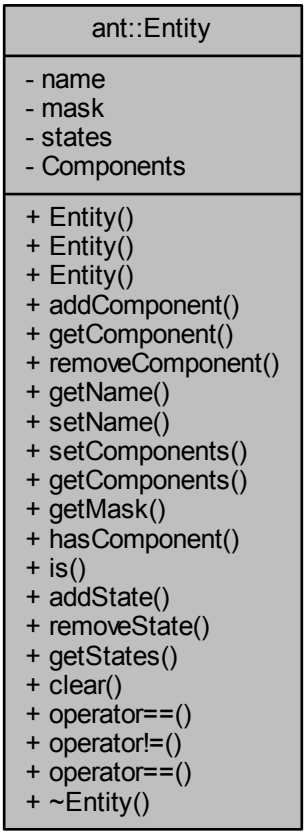
- AppState.hpp
- AppState.cpp

9.16 ant::Entity Class Reference

Clase para tener los componentes, solo puede tener un componente de cada tipo.

```
#include <Entity.hpp>
```

Collaboration diagram for ant::Entity:



Public Types

- typedef std::map< long int, std::unique_ptr< baseComponent > > map

Public Member Functions

- Entity ()

  Constructor por defecto.
- Entity (std::string name)

  Constructor con nombre.
- Entity (std::string name, map Components)

  Constructor con nombre y componentes.
- void addComponent (std::unique_ptr< baseComponent > component)

  A un componente a la entidad.
- baseComponent * getComponent (long int id)

  Obtiene una referencia al componente.
- void removeComponent (long int id)

Elimina el componente de la entidad.

- std::string getName () const

  Getter del nombre de la entidad.
- void setName (std::string name)

  Setter del nombre de la entidad.
- void setComponents (map Components)

  Setter de los componentes de la entidad.
- map & getComponents ()

  Getter de los componentes.
- long int getMask () const

  Getter de la mascara de la entidad.
- bool hasComponent (long int id)

  Devuelve si esta entidad tiene el componente.
- bool is (long int statesID)

  Devuelve los estados que tiene la entidad.
- void addState (long int state)

  A un estado a la entidad.
- void removeState (long int state)

  Elimina un estado de la entidad.
- long int getStates ()

  devuelve los estados de la entidad.
- void clear ()

  Limpia los componentes de la entidad.
- bool operator== (const Entity &e) const

  Operator overload para comparar la entidad contra otra entidad.
- bool operator!= (const Entity &e) const

  Operator overload para comparar la entidad contra otra entidad.
- bool operator== (const std::string &name) const

  Operator overload para comparar la entidad por nombre.
- ~Entity ()

Private Attributes

- std::string name
- long int mask
- long int states
- map Components

9.16.1 Detailed Description

Clase para tener los componentes, solo puede tener un componente de cada tipo.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 16 of file Entity.hpp.

9.16.2 Member Typedef Documentation

9.16.2.1 typedef std::map<long int,std::unique_ptr<baseComponent> > ant::Entity::map

Contenedor que utiliza la entidad para los componentes.

Definition at line 21 of file Entity.hpp.

9.16.3 Constructor & Destructor Documentation

9.16.3.1 ant::Entity::Entity ( )

Constructor por defecto.

Definition at line 3 of file Entity.cpp.

9.16.3.2 ant::Entity::Entity (std::string *name*)

Constructor con nombre.

Parameters

name	std::string nombre que tendra la entidad.

Definition at line 8 of file Entity.cpp.

9.16.3.3 ant::Entity::Entity (std::string *name,* map *Components*)

Constructor con nombre y componentes.

Parameters

name	std::string nombre que tendra la entidad.
Components	Entity::map mapa de los componentes.

Definition at line 13 of file Entity.cpp.

9.16.3.4 ant::Entity::~Entity ( )

Definition at line 63 of file Entity.cpp.

9.16.4 Member Function Documentation

9.16.4.1 void ant::Entity::addComponent (std::unique_ptr< baseComponent > *component*)

A un componente a la entidad.

Parameters

component	std::unique_ptr<baseComponent> componente para ar.

Definition at line 22 of file Entity.cpp.

9.16.4.2 void ant::Entity::addState (long int *state*)

A un estado a la entidad.

Parameters

state	long int estado a ar.

Definition at line 33 of file Entity.cpp.

Here is the caller graph for this function:

9.16.4.3 void ant::Entity::clear ( )

Limpia los componentes de la entidad.

Definition at line 50 of file Entity.cpp.

9.16.4.4 baseComponent* ant::Entity::getComponent (long int *id*) [inline]

Obtiene una referencia al componente.

Si el componente no se encuentra devuelve un nullptr.

Parameters

id	long int id del componente.

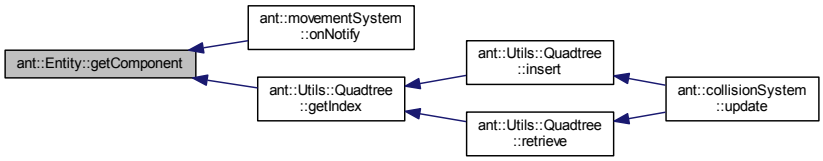
Returns

baseComponent * puntero al componente.

Definition at line 51 of file Entity.hpp.

Here is the call graph for this function:

Here is the caller graph for this function:



9.16.4.5 map& ant::Entity::getComponents ( ) [inline]

Getter de los componentes.

Returns

Entity::map referencia a los componentes.

Definition at line 82 of file Entity.hpp.

9.16.4.6 long int ant::Entity::getMask ( ) const [inline]

Getter de la mascara de la entidad.

Con este metodo comprobaras si una entidad tiene los componentes que necesitas antes de utilizar el get↩ Component(Component::ID);

```
if((e.getMask() & Component:ID) == Component::ID)
```

Returns

long int

Definition at line 92 of file Entity.hpp.

9.16.4.7 std::string ant::Entity::getName ( ) const [inline]

Getter del nombre de la entidad.

Returns

std::string nombre de la entidad.

Definition at line 67 of file Entity.hpp.

9.16.4.8 long int ant::Entity::getStates ( ) [inline]

devuelve los estados de la entidad.

Returns

long int Estados de la entidad.

Definition at line 122 of file Entity.hpp.

9.16.4.9 bool ant::Entity::hasComponent (long int *id*) [inline]

Devuelve si esta entidad tiene el componente.

Parameters

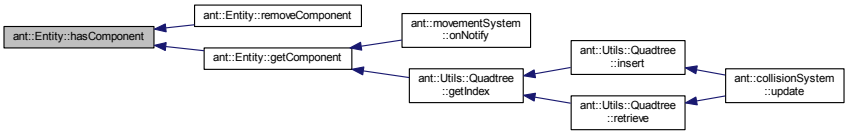
id	long int Id del componente.

Returns

bool

Definition at line 98 of file Entity.hpp.

Here is the caller graph for this function:



9.16.4.10 bool ant::Entity::is (long int *statesID*) [inline]

Devuelve los estados que tiene la entidad.

Parameters

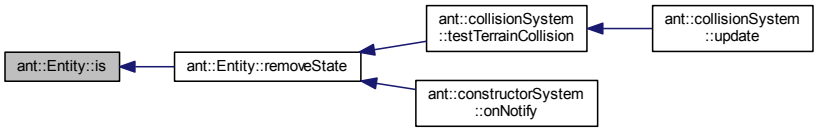
statesID	long int id de los estados. `e->is(States::FALLING | States::OnFIRE);`

Returns

bool

Definition at line 107 of file Entity.hpp.

Here is the caller graph for this function:



9.16.4.11 bool ant::Entity::operator!= (const Entity & *e*) const

Operator overload para comparar la entidad contra otra entidad.

Parameters

e	Entity entidad a la cual se compara.

Returns

bool

Definition at line 57 of file Entity.cpp.

9.16.4.12 bool ant::Entity::operator== (const Entity & *e*) const

Operator overload para comparar la entidad contra otra entidad.

Parameters

e	Entity entidad a la cual compararse.

Returns

bool

Definition at line 54 of file Entity.cpp.

9.16.4.13 bool ant::Entity::operator== (const std::string & *name*) const

Operator overload para comparar la entidad por nombre.

Parameters

name	std::string nombre con el cual comparar.

Returns

bool

Definition at line 60 of file Entity.cpp.

9.16.4.14 void ant::Entity::removeComponent (long int *id*)

Elimina el componente de la entidad.

Parameters

id	long int

Definition at line 27 of file Entity.cpp.

Here is the call graph for this function:

9.16.4.15 void ant::Entity::removeState (long int *state*)

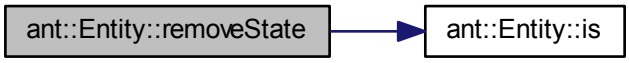
Elimina un estado de la entidad.

Parameters

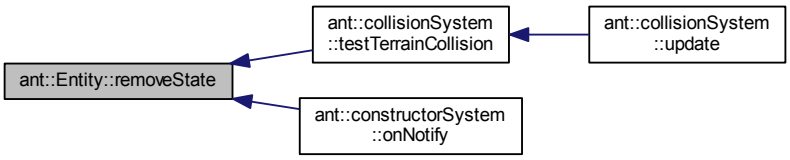
state	long int estado a eliminar.

Definition at line 36 of file Entity.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.16.4.16 void ant::Entity::setComponents (map *Components*)

Setter de los componentes de la entidad.

Parameters

Components	Entity::map

Definition at line 44 of file Entity.cpp.

9.16.4.17 void ant::Entity::setName (std::string *name*)

Setter del nombre de la entidad.

Parameters

name	std::string nombre nuevo de la entidad.

Definition at line 41 of file Entity.cpp.

9.16.5 Member Data Documentation

9.16.5.1 map ant::Entity::Components [private]

Definition at line 150 of file Entity.hpp.

9.16.5.2 long int ant::Entity::mask [private]

Definition at line 148 of file Entity.hpp.

9.16.5.3 std::string ant::Entity::name [private]

Definition at line 147 of file Entity.hpp.

9.16.5.4 long int ant::Entity::states [private]

Definition at line 149 of file Entity.hpp.

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

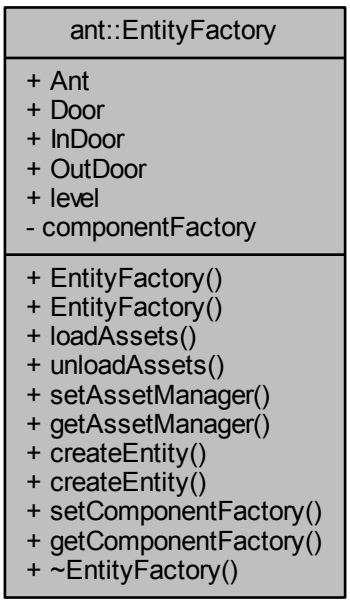
- Entity.hpp
- Entity.cpp

9.17 ant::EntityFactory Class Reference

Clase para crear las entidades del juego.

```
#include <EntityFactory.hpp>
```

Collaboration diagram for ant::EntityFactory:



Public Member Functions

- EntityFactory ()

Constructor por defecto.

- EntityFactory (const std::string &json)

  Constructor que carga los assets del json especificado.
- bool loadAssets (const std::string &json)

  Carga los assets especificados en el fichero json.
- bool unloadAssets ()

  Descarga los assets cargados.
- void setAssetManager (std::shared_ptr< AssetManager > assets)

  Setter del assetManager.
- std::shared_ptr< AssetManager > getAssetManager ()

  Getter del AssetManager.
- std::unique_ptr< Entity > createEntity (long int mask)

  Crea la entidad especificada.
- std::unique_ptr< Entity > createEntity (long int mask, ComponentSettings &cs)

  Crea la entidad especificada con los datos especificados.
- void setComponentFactory (std::shared_ptr< ComponentFactory > cf)

  Setter de la fabrica de componentes a utilizar.
- std::shared_ptr< ComponentFactory > getComponentFactory ()

  Getter de la fabrica de componentes utilizada.
- ~EntityFactory ()

Static Public Attributes

- static const long int Ant
- static const long int Door
- static const long int InDoor
- static const long int OutDoor
- static const long int level

Private Attributes

- std::shared_ptr< ComponentFactory > componentFactory

9.17.1 Detailed Description

Clase para crear las entidades del juego.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 14 of file EntityFactory.hpp.

9.17.2 Constructor & Destructor Documentation

9.17.2.1 ant::EntityFactory::EntityFactory ( )

Constructor por defecto.

Definition at line 4 of file EntityFactory.cpp.

9.17.2.2 ant::EntityFactory::EntityFactory (const std::string & *json*)

Constructor que carga los assets del json especificado.

Parameters

json	std::string fichero json a cargar.

Definition at line 7 of file EntityFactory.cpp.

9.17.2.3 ant::EntityFactory::~EntityFactory ( )

Definition at line 68 of file EntityFactory.cpp.

9.17.3 Member Function Documentation

9.17.3.1 std::unique_ptr< Entity > ant::EntityFactory::createEntity (long int *mask*)

Crea la entidad especificada.

Se puede especificar otros componentes pero tendran los datos por defecto

Parameters

mask	long int Mascara de la entidad o componentes. `EntityFactory::createEntity( EntityFactory::ANT | Component::Wings | Component::Horns);` Creas una entidad Hormiga con los componentes adicionales especificados los cuales tendran valores por defecto.

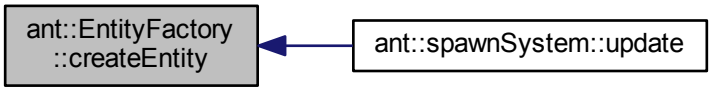
Returns

std::unique_ptr<Entity> entidad creada.

Definition at line 20 of file EntityFactory.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:



9.17.3.2 std::unique_ptr< Entity > ant::EntityFactory::createEntity (long int *mask,* ComponentSettings & *cs*)

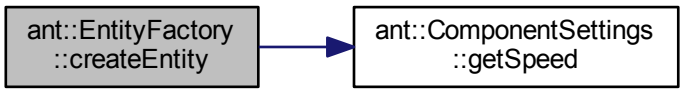
Crea la entidad especificada con los datos especificados.

Parameters

mask	long int mascara de la entidad o componentes.
cs	ComponentSettings

Definition at line 36 of file EntityFactory.cpp.

Here is the call graph for this function:



9.17.3.3 std::shared_ptr<AssetManager> ant::EntityFactory::getAssetManager ( ) [inline]

Getter del AssetManager.

Returns

std::shared_ptr<AssetManager>

Definition at line 68 of file EntityFactory.hpp.

9.17.3.4 std::shared_ptr<ComponentFactory> ant::EntityFactory::getComponentFactory ( ) [inline]

Getter de la fabrica de componentes utilizada.

Returns

std::shared_ptr<ComponentFactory>

Definition at line 97 of file EntityFactory.hpp.

9.17.3.5 bool ant::EntityFactory::loadAssets (const std::string & *json*)

Carga los assets especificados en el fichero json.

Formato del fichero json AssetManager::loadAssets()

Parameters

json	std::string fichero json a cargar.

Returns

bool

Definition at line 11 of file EntityFactory.cpp.

9.17.3.6 void ant::EntityFactory::setAssetManager (std::shared_ptr< AssetManager > *assets*)

Setter del assetManager.

Parameters

assets	std::shared_ptr<AssetManager> assets que utilizara para cargar componentes que los utilicen.

Definition at line 17 of file EntityFactory.cpp.

9.17.3.7 void ant::EntityFactory::setComponentFactory (std::shared_ptr< ComponentFactory > *cf*)

Setter de la fabrica de componentes a utilizar.

Parameters

cf	std::shared_ptr<ComponentFactory>

Definition at line 65 of file EntityFactory.cpp.

9.17.3.8 bool ant::EntityFactory::unloadAssets ( )

Descarga los assets cargados.

Returns

bool

Definition at line 14 of file EntityFactory.cpp.

9.17.4 Member Data Documentation

9.17.4.1 const long int ant::EntityFactory::Ant [static]

Initial value:

```
= ComponentsMask::COMPONENT_VELOCITY
                                | ComponentsMask::COMPONENT_SPRITE
                                | ComponentsMask::COMPONENT_TRANSFORM
                                | ComponentsMask::COMPONENT_COUNT
                                | ComponentsMask::COMPONENT_BOUNDS
```

Definition at line 16 of file EntityFactory.hpp.

9.17.4.2 std::shared_ptr<ComponentFactory> ant::EntityFactory::componentFactory [private]

Definition at line 100 of file EntityFactory.hpp.

9.17.4.3 const long int ant::EntityFactory::Door [static]

Initial value:

```
= ComponentsMask::COMPONENT_PASSAGE
                                | ComponentsMask::COMPONENT_TRANSFORM
                                | ComponentsMask::COMPONENT_SPRITE
                                | ComponentsMask::COMPONENT_BOUNDS
```

Definition at line 21 of file EntityFactory.hpp.

9.17.4.4 const long int ant::EntityFactory::InDoor [static]

Initial value:

```
= ComponentsMask::COMPONENT_TRANSFORM
                                | ComponentsMask::COMPONENT_SPRITE
                                | ComponentsMask::COMPONENT_IN
```

Definition at line 25 of file EntityFactory.hpp.

9.17.4.5 const long int ant::EntityFactory::level [static]

Initial value:

```
= ComponentsMask::COMPONENT_DESTRUCTABLE
                                | ComponentsMask::COMPONENT_TRANSFORM
                                | ComponentsMask::COMPONENT_SPRITE
```

Definition at line 32 of file EntityFactory.hpp.

9.17.4.6 const long int ant::EntityFactory::OutDoor [static]

Initial value:

```
= ComponentsMask::COMPONENT_TRANSFORM
                                | ComponentsMask::COMPONENT_BOUNDS
                                | ComponentsMask::COMPONENT_SPRITE
                                | ComponentsMask::COMPONENT_OUT
```

Definition at line 28 of file EntityFactory.hpp.

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

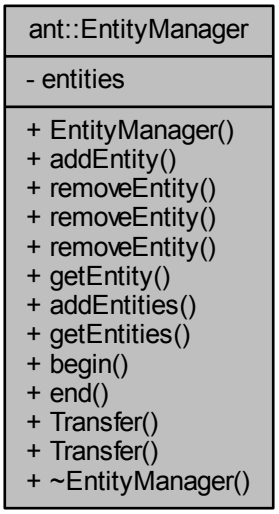
- EntityFactory.hpp
- EntityFactory.cpp

9.18 ant::EntityManager Class Reference

Clase Para manejar las entidades.

`#include <EntityManager.hpp>`

Collaboration diagram for ant::EntityManager:



Public Types

- typedef std::list < std::unique_ptr< Entity > >::iterator iterator

  Iterador del EntityManager.
- typedef std::list < std::unique_ptr< Entity > > list

  Contenedor utilizado por el EntityManager.

Public Member Functions

- EntityManager ()

  Constructor por defecto.
- void addEntity (std::unique_ptr< Entity > e)

  A una entidad al manager.
- void removeEntity (std::unique_ptr< Entity > &e)

  Elimina una entidad del manager.
- iterator removeEntity (const std::string &name)

  Elimina la primera entidad que coincida el nombre.
- iterator removeEntity (iterator it)

  Elimina la primera entidad que coincida.
- Entity * getEntity (const std::string &name)

  Obtiene la primera entidad que tenga el nombre proporcionado.
- void addEntities (list e)

  A todas las entidades a la vez.
- list & getEntities ()

devuelve la lista de las entidades.

- iterator begin ()

  Devuelve el principio de la lista.
- iterator end ()

  Devuelve el final de la lista.
- std::unique_ptr< Entity > Transfer (const std::string &name)

  Transfiere la primera entidad que coincida con el nombre.
- std::unique_ptr< Entity > Transfer (const Entity *e)

  Transfiere la primera entidad que coincida con el puntero de entidad.
- ~EntityManager ()

Private Attributes

- list entities

9.18.1 Detailed Description

Clase Para manejar las entidades.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 13 of file EntityManager.hpp.

9.18.2 Member Typedef Documentation

9.18.2.1 typedef std::list<std::unique_ptr<Entity> >::iterator ant::EntityManager::iterator

Iterador del EntityManager.

Definition at line 18 of file EntityManager.hpp.

9.18.2.2 typedef std::list<std::unique_ptr<Entity> > ant::EntityManager::list

Contenedor utilizado por el EntityManager.

Definition at line 22 of file EntityManager.hpp.

9.18.3 Constructor & Destructor Documentation

9.18.3.1 ant::EntityManager::EntityManager ( )

Constructor por defecto.

Definition at line 4 of file EntityManager.cpp.

9.18.3.2 ant::EntityManager::~EntityManager ( )

Definition at line 65 of file EntityManager.cpp.

9.18.4 Member Function Documentation

9.18.4.1 void ant::EntityManager::addEntities (list *e*)

A todas las entidades a la vez.

Parameters

e	list lista de entidades.

Definition at line 36 of file EntityManager.cpp.

9.18.4.2 void ant::EntityManager::addEntity (std::unique_ptr< Entity > *e*)

A una entidad al manager.

Parameters

e	std::unique_ptr<Entity>

Definition at line 7 of file EntityManager.cpp.

9.18.4.3 iterator ant::EntityManager::begin ( ) [inline]

Devuelve el principio de la lista.

Definition at line 80 of file EntityManager.hpp.

9.18.4.4 iterator ant::EntityManager::end ( ) [inline]

Devuelve el final de la lista.

Definition at line 84 of file EntityManager.hpp.

9.18.4.5 list& ant::EntityManager::getEntities ( ) [inline]

devuelve la lista de las entidades.

Returns

list &

Definition at line 76 of file EntityManager.hpp.

9.18.4.6 Entity * ant::EntityManager::getEntity (const std::string & *name*)

Obtiene la primera entidad que tenga el nombre proporcionado.

Si no encuentra la entidad devuelve un puntero nulo(nullptr).

Parameters

name	std::string nombre de la entidad.

Returns

Entity * naked pointer hacia la entidad o nullptr si no la encuentra.

Definition at line 28 of file EntityManager.cpp.

9.18.4.7 void ant::EntityManager::removeEntity (std::unique_ptr< Entity > & *e*)

Elimina una entidad del manager.

Elimina la primera entidad que sea igual a la proporcionada.

Parameters

e	std::unique_ptr<Entity>

Definition at line 10 of file EntityManager.cpp.

9.18.4.8 EntityManager::iterator ant::EntityManager::removeEntity (const std::string & *name*)

Elimina la primera entidad que coincida el nombre.

Devuelve un iterador por si estas iterando y necesitas borrar alguna entidad.

Parameters

name	std::string nombre de la entidad.

Returns

iterator la posicion siguiente de la entidad eliminada.

Definition at line 13 of file EntityManager.cpp.

9.18.4.9 EntityManager::iterator ant::EntityManager::removeEntity (EntityManager::iterator *it*)

Elimina la primera entidad que coincida.

Devuelve un iterador por si estas iterando y necesitas borrar alguna entidad.

Returns

iterator la posicion siguiente de la entidad eliminada.

Definition at line 25 of file EntityManager.cpp.

9.18.4.10 std::unique_ptr< Entity > ant::EntityManager::Transfer (const std::string & *name*)

Transfiere la primera entidad que coincida con el nombre.

Transfiere la posesion(ownership) de la entidad y la elimina del contenedor.

Parameters

name	std::string nombre de la entidad a transferir.

Returns

std::unique_ptr<Entity> entidad transferida o nullptr si no es encontrada.

Definition at line 39 of file EntityManager.cpp.

9.18.4.11 std::unique_ptr< Entity > ant::EntityManager::Transfer (const Entity * *e*)

Transfiere la primera entidad que coincida con el puntero de entidad.

Transfiere la posesion(ownership) de la entidad y la elimina del contenedor.

Parameters

e	Entity * Puntero a la entidad

Returns

std::unique_ptr<Entity> entidad transferida o nullptr si no es encontrada.

Definition at line 52 of file EntityManager.cpp.

9.18.5 Member Data Documentation

9.18.5.1 list ant::EntityManager::entities [private]

Definition at line 105 of file EntityManager.hpp.

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

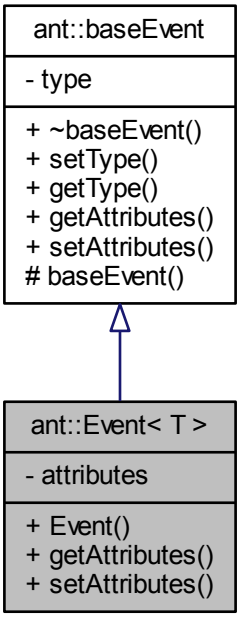
- EntityManager.hpp
- EntityManager.cpp

9.19 ant::Event< T > Class Template Reference

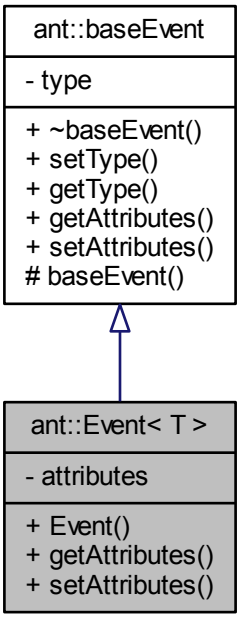
Clase Especializada de evento.

`#include <Event.hpp>`

Inheritance diagram for ant::Event< T >:



Collaboration diagram for ant::Event< T >:



Public Member Functions

- Event (int type, T...data)

  Constructor del Evento Especializado.
- const std::tuple< T...> & getAttributes () const

  Getter de los attributos del evento.
- void setAttributes (std::tuple< T...> attributes)

  Setter de los attributos del evento.
- void setType (int type)

  Setter para poner el tipo al evento.
- int getType () const

  Getter del tipo de evento.

Private Attributes

- std::tuple< T...> attributes

9.19.1 Detailed Description

template<typename... T>class ant::Event< T >

Clase Especializada de evento.

Template Parameters

...*T*	tipo de datos que tendra el evento.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 55 of file Event.hpp.

9.19.2 Constructor & Destructor Documentation

9.19.2.1 template<typename... T> ant::Event< T >::Event (int *type*, T... *data*) [inline]

Constructor del Evento Especializado.

Parameters

type	int Tipo del evento.
data	T... attributos del evento.

Definition at line 63 of file Event.hpp.

9.19.3 Member Function Documentation

9.19.3.1 template<typename... T> const std::tuple<T...>& ant::Event< T >::getAttributes ( ) const [inline]

Getter de los attributos del evento.

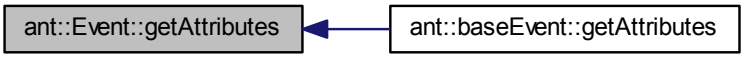
Los datos del evento no se pueden modificar, son solo lectura.

Returns

std::tuple<T...>& const reference a la tupla.

Definition at line 73 of file Event.hpp.

Here is the caller graph for this function:



9.19.3.2 int ant::baseEvent::getType ( ) const [inline], [inherited]

Getter del tipo de evento.

Returns

int tipo del evento.

Definition at line 28 of file Event.hpp.

9.19.3.3 template<typename... T> void ant::Event< T >::setAttributes (std::tuple< T...> *attributes*) [inline]

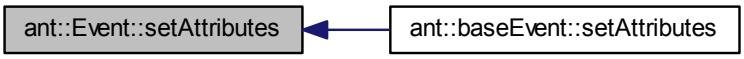
Setter de los attributos del evento.

Parameters

attributes	std::tuple<T...> attributos del evento.

Definition at line 79 of file Event.hpp.

Here is the caller graph for this function:



9.19.3.4 void ant::baseEvent::setType (int *type*) [inline],[inherited]

Setter para poner el tipo al evento.

Parameters

type	int tipo de evento.

Definition at line 23 of file Event.hpp.

9.19.4 Member Data Documentation

9.19.4.1 template<typename... T> std::tuple<T...> ant::Event< T >::attributes [private]

Definition at line 81 of file Event.hpp.

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

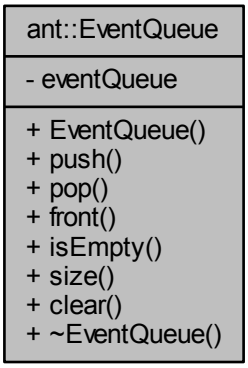
- Event.hpp

9.20 ant::EventQueue Class Reference

Clase donde se almacenan los eventos para ser procesados despues.

```
#include <EventQueue.hpp>
```

Collaboration diagram for ant::EventQueue:



Public Member Functions

- EventQueue ()
- void push (std::shared_ptr< baseEvent > e)

  A un evento al final de la cola.
- std::shared_ptr< baseEvent > pop ()

  Elimina el evento de la cola y lo devuelve.
- std::shared_ptr< baseEvent > & front ()

  Consulta el evento que esta al frente de la cola.
- bool isEmpty ()

  Si la cola esta vacia o no.
- int size ()

  Devuelve el numero de eventos en la cola.
- void clear ()

  limpia la cola.
- ~EventQueue ()

Private Attributes

- std::queue< std::shared_ptr < baseEvent > > eventQueue

9.20.1 Detailed Description

Clase donde se almacenan los eventos para ser procesados despues.

Author

Cristian Gonzalez Moreno `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 12 of file EventQueue.hpp.

9.20.2 Constructor & Destructor Documentation

9.20.2.1 ant::EventQueue::EventQueue ( )

Constructor por defecto.

Definition at line 3 of file EventQueue.cpp.

9.20.2.2 ant::EventQueue::~EventQueue ( )

Definition at line 22 of file EventQueue.cpp.

9.20.3 Member Function Documentation

9.20.3.1 void ant::EventQueue::clear ( )

limpia la cola.

Definition at line 17 of file EventQueue.cpp.

9.20.3.2 std::shared_ptr< baseEvent > & ant::EventQueue::front ( )

Consulta el evento que esta al frente de la cola.

Returns

std::shared_ptr<baseEvent> & referencia al evento.

Definition at line 6 of file EventQueue.cpp.

9.20.3.3 bool ant::EventQueue::isEmpty ( ) [inline]

Si la cola esta vacia o no.

Returns

bool

Definition at line 37 of file EventQueue.hpp.

9.20.3.4 std::shared_ptr< baseEvent > ant::EventQueue::pop ( )

Elimina el evento de la cola y lo devuelve.

Returns

std::shared_ptr<baseEvent>

Definition at line 9 of file EventQueue.cpp.

9.20.3.5 void ant::EventQueue::push (std::shared_ptr< baseEvent > *e*)

A un evento al final de la cola.

Parameters

e	std::shared_ptr<baseEvent> Evento a ar.

Definition at line 14 of file EventQueue.cpp.

9.20.3.6 int ant::EventQueue::size ( ) [inline]

Devuelve el numero de eventos en la cola.

Returns

int

Definition at line 42 of file EventQueue.hpp.

9.20.4 Member Data Documentation

9.20.4.1 std::queue<std::shared_ptr<baseEvent> > ant::EventQueue::eventQueue [private]

Definition at line 49 of file EventQueue.hpp.

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

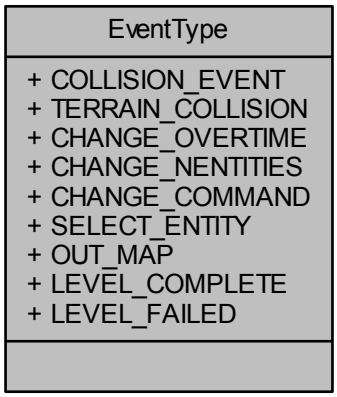
- EventQueue.hpp
- EventQueue.cpp

9.21 EventType Class Reference

Tipos de eventos.

`#include <EventType.hpp>`

Collaboration diagram for EventType:



Static Public Attributes

- static const long int COLLISION_EVENT = 0
- static const long int TERRAIN_COLLISION = 1
- static const long int CHANGE_OVERTIME = 2
- static const long int CHANGE_NENTITIES = 3
- static const long int CHANGE_COMMAND = 4
- static const long int SELECT_ENTITY = 5
- static const long int OUT_MAP = 6
- static const long int LEVEL_COMPLETE = 7
- static const long int LEVEL_FAILED = 8

9.21.1 Detailed Description

Tipos de eventos.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 8 of file EventType.hpp.

9.21.2 Member Data Documentation

9.21.2.1 const long int EventType::CHANGE_COMMAND = 4 [static]

Definition at line 14 of file EventType.hpp.

9.21.2.2 const long int EventType::CHANGE_NENTITIES = 3 [static]

Definition at line 13 of file EventType.hpp.

9.21.2.3 const long int EventType::CHANGE_OVERTIME = 2 [static]

Definition at line 12 of file EventType.hpp.

9.21.2.4 const long int EventType::COLLISION_EVENT = 0 [static]

Definition at line 10 of file EventType.hpp.

9.21.2.5 const long int EventType::LEVEL_COMPLETE = 7 [static]

Definition at line 17 of file EventType.hpp.

9.21.2.6 const long int EventType::LEVEL_FAILED = 8 [static]

Definition at line 18 of file EventType.hpp.

9.21.2.7 const long int EventType::OUT_MAP = 6 [static]

Definition at line 16 of file EventType.hpp.

9.21.2.8 const long int EventType::SELECT_ENTITY = 5 [static]

Definition at line 15 of file EventType.hpp.

9.21.2.9 const long int EventType::TERRAIN_COLLISION = 1 [static]

Definition at line 11 of file EventType.hpp.

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

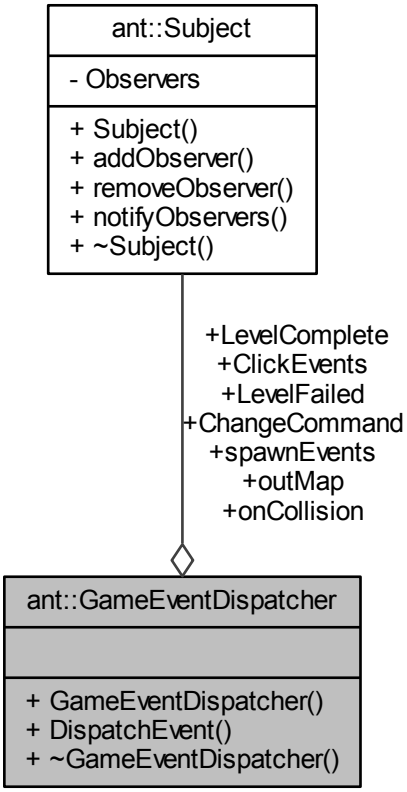
- EventType.hpp

9.22 ant::GameEventDispatcher Class Reference

Clase encargada de processar eventos.

`#include <GameEventDispatcher.hpp>`

Collaboration diagram for ant::GameEventDispatcher:



Public Member Functions

- GameEventDispatcher ()

  Constructor por defecto.
- bool DispatchEvent (std::shared_ptr< baseEvent > e)

  Procesa el evento y lo envia a los observadores que esten subscritos.
- ~GameEventDispatcher ()

Public Attributes

- Subject onCollision

  lista de observadores del evento onCollision.
- Subject spawnEvents

  Eventos para el sistema de creacion de entidades.
- Subject ClickEvents

  Eventos de click dan evento con un punto.
- Subject ChangeCommand

  Envia un Comando al sistema de construccion.
- Subject outMap

  Envia una entidad que ha salido del mapa.
- Subject LevelComplete

  El sistema de out envia este evento.
- Subject LevelFailed

  El sistema de out envia este evento.

9.22.1 Detailed Description

Clase encargada de processar eventos.

Esta clase contiene a los eventos a los cuales se pueden suscribir los observadores.

Author

Cristian Gonzalez Moreno `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 16 of file GameEventDispatcher.hpp.

9.22.2 Constructor & Destructor Documentation

9.22.2.1 ant::GameEventDispatcher::GameEventDispatcher ( )

Constructor por defecto.

Definition at line 3 of file GameEventDispatcher.cpp.

9.22.2.2 ant::GameEventDispatcher::~GameEventDispatcher ( )

Definition at line 34 of file GameEventDispatcher.cpp.

9.22.3 Member Function Documentation

9.22.3.1 bool ant::GameEventDispatcher::DispatchEvent (std::shared_ptr< baseEvent > *e*)

Procesa el evento y lo envia a los observadores que esten subscritos.

Parameters

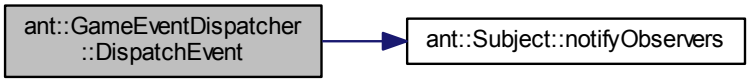
e	std::shared_ptr<baseEvent>

Returns

bool

Definition at line 6 of file GameEventDispatcher.cpp.

Here is the call graph for this function:



9.22.4 Member Data Documentation

9.22.4.1 Subject ant::GameEventDispatcher::ChangeCommand

Envia un Comando al sistema de construccion.

Definition at line 37 of file GameEventDispatcher.hpp.

9.22.4.2 Subject ant::GameEventDispatcher::ClickEvents

Eventos de click dan evento con un punto.

Definition at line 33 of file GameEventDispatcher.hpp.

9.22.4.3 Subject ant::GameEventDispatcher::LevelComplete

El sistema de out envia este evento.

Definition at line 45 of file GameEventDispatcher.hpp.

9.22.4.4 Subject ant::GameEventDispatcher::LevelFailed

El sistema de out envia este evento.

Definition at line 49 of file GameEventDispatcher.hpp.

9.22.4.5 Subject ant::GameEventDispatcher::onCollision

lista de observadores del evento onCollision.

Definition at line 25 of file GameEventDispatcher.hpp.

9.22.4.6 Subject ant::GameEventDispatcher::outMap

Envia una entidad que ha salido del mapa.

Definition at line 41 of file GameEventDispatcher.hpp.

9.22.4.7 Subject ant::GameEventDispatcher::spawnEvents

Eventos para el sistema de creacion de entidades.

Definition at line 29 of file GameEventDispatcher.hpp.

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

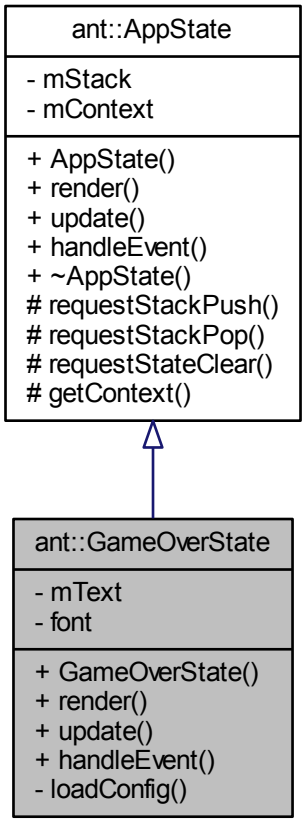
- GameEventDispatcher.hpp
- GameEventDispatcher.cpp

9.23 ant::GameOverState Class Reference

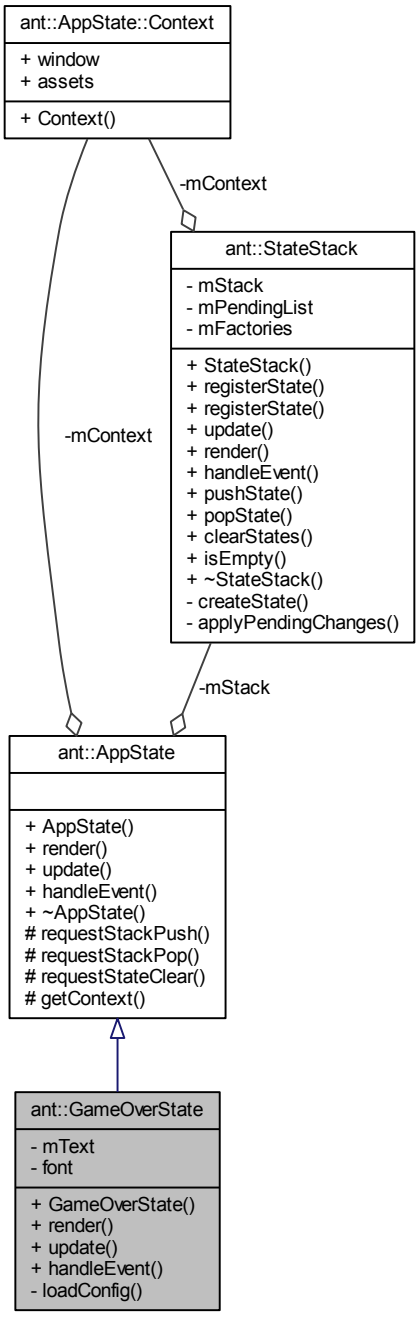
Muestra el mensaje de final de juego.

```
#include <GameOverState.hpp>
```

Inheritance diagram for ant::GameOverState:



Collaboration diagram for ant::GameOverState:



Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.

- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- GameOverState (StateStack &stack, Context context, std::string msg)

  Constructor principal.
- virtual void render ()

  Renderiza el estado.
- virtual bool update (sf::Time dt)

  Actualiza el estado.
- virtual bool handleEvent (const sf::Event &event)

  Maneja la input.

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Member Functions

- void loadConfig (const std::string &filename)

  Carga la configuracion de este estado.

Private Attributes

- sf::Text mText
- std::string font

9.23.1 Detailed Description

Muestra el mensaje de final de juego.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1a

Definition at line 13 of file GameOverState.hpp.

9.23.2 Member Typedef Documentation

9.23.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr [inherited]

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.23.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr [inherited]

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

9.23.3 Constructor & Destructor Documentation

9.23.3.1 ant::GameOverState::GameOverState (StateStack & *stack*, Context *context*, std::string *msg*)

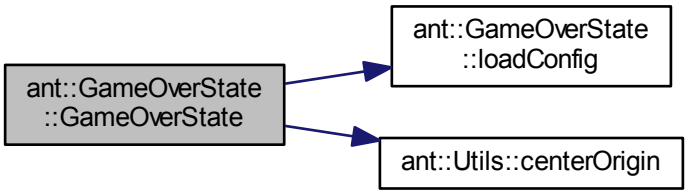
Constructor principal.

Parameters

stack	StateStack stack para cambiar estados internamente.
context	Context contexto de la aplicacion.
msg	std::string mensaje que mostrara.

Definition at line 4 of file GameOverState.cpp.

Here is the call graph for this function:



9.23.4 Member Function Documentation

9.23.4.1 AppState::Context ant::AppState::getContext ( ) const [protected],[inherited]

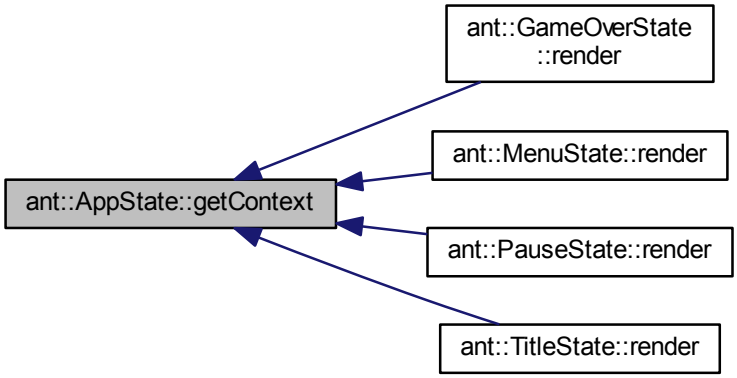
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



9.23.4.2 bool ant::GameOverState::handleEvent (const sf::Event & *event*) [virtual]

Maneja la input.

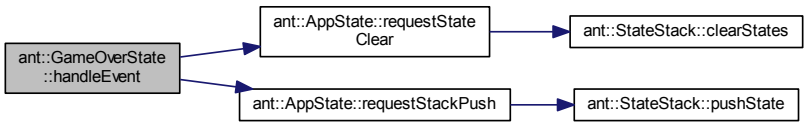
Parameters

event	sf::Event

Implements ant::AppState.

Definition at line 27 of file GameOverState.cpp.

Here is the call graph for this function:



9.23.4.3 void ant::GameOverState::loadConfig (const std::string & *filename*) [private]

Carga la configuracion de este estado.

Parameters

filename	std::string fichero a cargar.

Definition at line 17 of file GameOverState.cpp.

Here is the caller graph for this function:

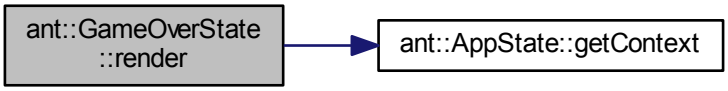
9.23.4.4 void ant::GameOverState::render ( ) [virtual]

Renderiza el estado.

Implements ant::AppState.

Definition at line 13 of file GameOverState.cpp.

Here is the call graph for this function:

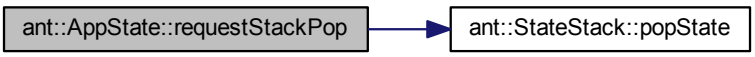


9.23.4.5 void ant::AppState::requestStackPop ( ) [protected],[inherited]

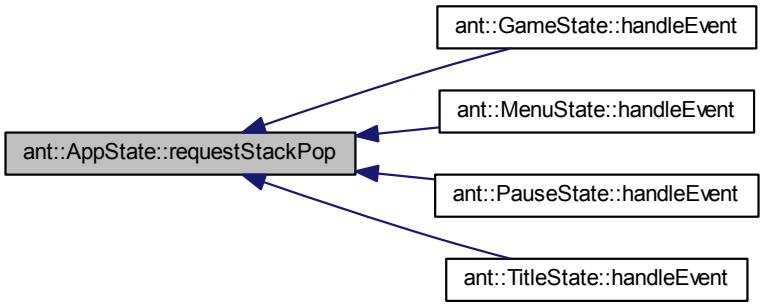
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.23.4.6 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected], [inherited]

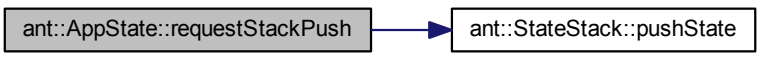
Pide al stack que a un estado nuevo.

Parameters

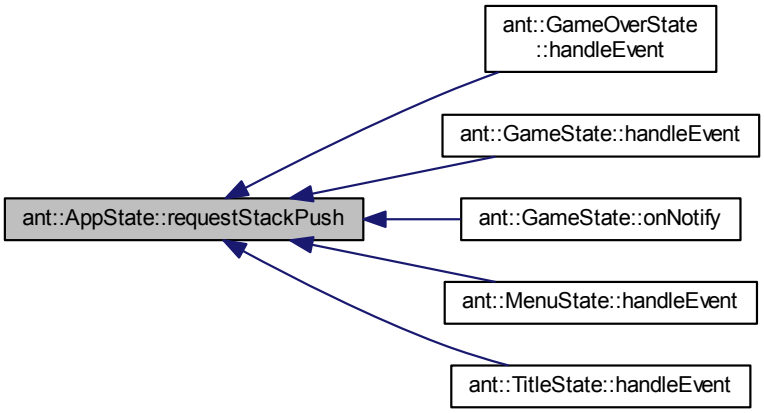
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

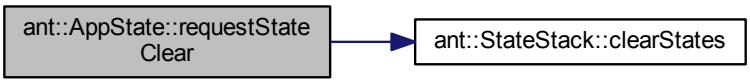


9.23.4.7 void ant::AppState::requestStateClear ( ) [protected],[inherited]

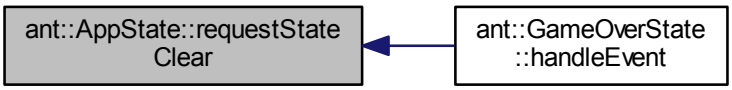
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.23.4.8 bool ant::GameOverState::update (sf::Time *dt*) [virtual]

Actualiza el estado.

Parameters

dt	sf::Time

Implements ant::AppState.

Definition at line 24 of file GameOverState.cpp.

9.23.5 Member Data Documentation

9.23.5.1 std::string ant::GameOverState::font [private]

Definition at line 44 of file GameOverState.hpp.

9.23.5.2 sf::Text ant::GameOverState::mText [private]

Definition at line 43 of file GameOverState.hpp.

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

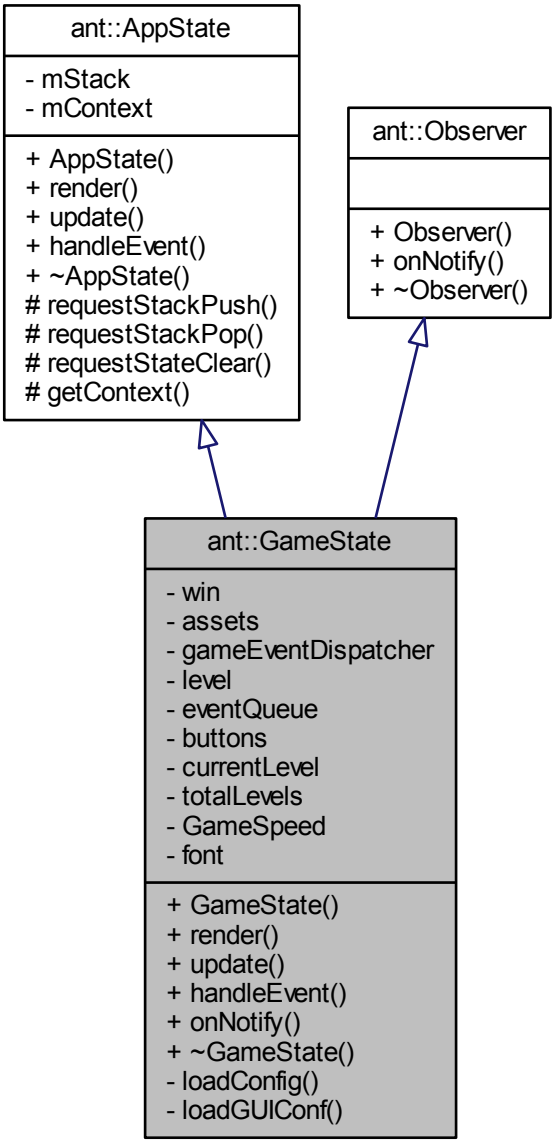
- GameOverState.hpp
- GameOverState.cpp

9.24 ant::GameState Class Reference

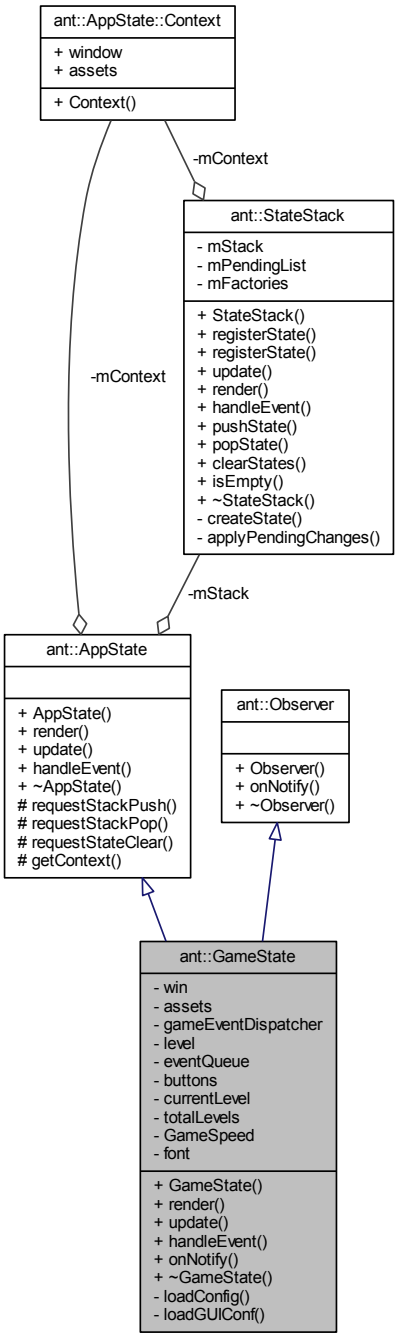
Estado de juego, Inicia el juego.

```
#include <GameState.hpp>
```

Inheritance diagram for ant::GameState:



Collaboration diagram for ant::GameState:



Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.

- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- GameState (StateStack &stack, Context context)

  Constructor principal.
- virtual void render ()

  Renderiza el juego.
- virtual bool update (sf::Time dt)

  Actualiza el juego.
- virtual bool handleEvent (const sf::Event &event)

  Maneja el input del juego.
- void onNotify (std::shared_ptr< baseEvent > e)

  Recibe los eventos.
  1. *LEVEL_COMPLETE cambia de nivel.*
  2. *LEVEL_FAILED vuelve al menu.*
- virtual ~GameState ()

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Member Functions

- void loadConfig (const std::string &filename)

  Carga la configuracion de este estado.
- void loadGUIConf (const std::string &filename)

  Carga la interfaz grafica.

Private Attributes

- sf::RenderWindow & win
- std::shared_ptr< AssetManager > assets
- std::shared_ptr
  < GameEventDispatcher > gameEventDispatcher
- std::shared_ptr< Level > level
- std::shared_ptr< EventQueue > eventQueue
- std::vector< std::unique_ptr
  < GUI::Button > > buttons
- int currentLevel
- int totalLevels
- float GameSpeed
- std::string font

9.24.1 Detailed Description

Estado de juego, Inicia el juego.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 18 of file GameState.hpp.

9.24.2 Member Typedef Documentation

9.24.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr [inherited]

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.24.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr [inherited]

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

9.24.3 Constructor & Destructor Documentation

9.24.3.1 ant::GameState::GameState (StateStack & *stack*, Context *context*)

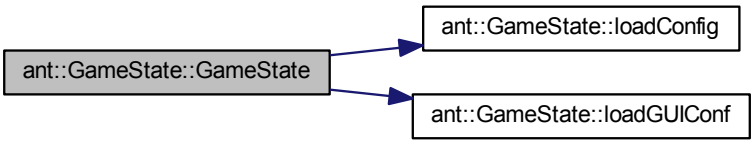
Constructor principal.

Parameters

stack	StateStack stack para pedir otros estados etc.
context	Context contexto de la applicacion.

Definition at line 4 of file GameState.cpp.

Here is the call graph for this function:



9.24.3.2 ant::GameState::~GameState ( ) [virtual]

Definition at line 219 of file GameState.cpp.

9.24.4 Member Function Documentation

9.24.4.1 AppState::Context ant::AppState::getContext ( ) const [protected], [inherited]

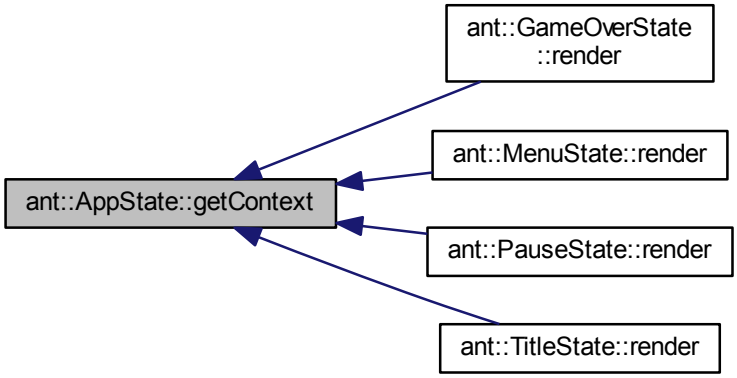
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



9.24.4.2 bool ant::GameState::handleEvent (const sf::Event & *event*) [virtual]

Maneja el input del juego.

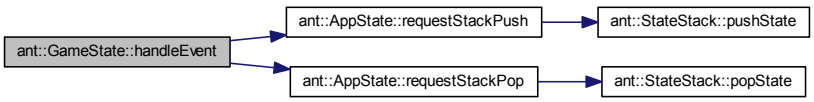
Parameters

event	sf::Event & eventos de teclado/Raton etc

Implements ant::AppState.

Definition at line 32 of file GameState.cpp.

Here is the call graph for this function:



9.24.4.3 void ant::GameState::loadConfig (const std::string & *filename*) [private]

Carga la configuracion de este estado.

Definition at line 212 of file GameState.cpp.

Here is the caller graph for this function:

9.24.4.4 void ant::GameState::loadGUIConf (const std::string & *filename*) [private]

Carga la interfaz grafica.

Parameters

filename	std::string

Definition at line 165 of file GameState.cpp.

Here is the caller graph for this function:

9.24.4.5 void ant::GameState::onNotify (std::shared_ptr< **baseEvent** > *e*) [virtual]

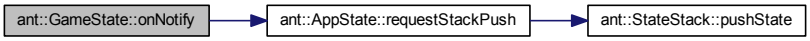
Recibe los eventos.

1. LEVEL_COMPLETE cambia de nivel.
2. LEVEL_FAILED vuelve al menu.

Implements ant::Observer.

Definition at line 149 of file GameState.cpp.

Here is the call graph for this function:



9.24.4.6 void ant::GameState::render ( ) [virtual]

Renderiza el juego.

Implements ant::AppState.

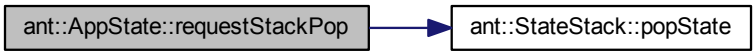
Definition at line 19 of file GameState.cpp.

9.24.4.7 void ant::AppState::requestStackPop ( ) [protected],[inherited]

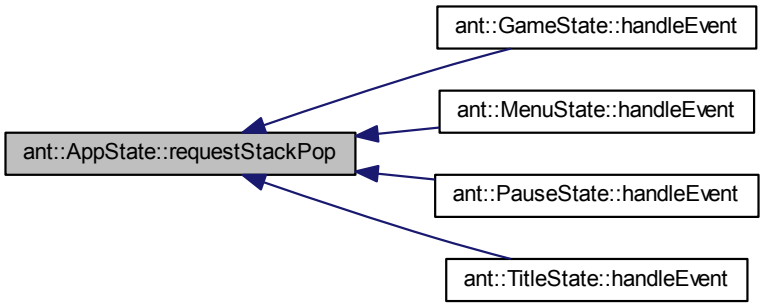
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.8 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected],[inherited]

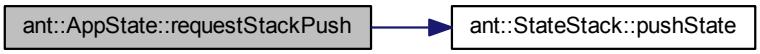
Pide al stack que a un estado nuevo.

Parameters

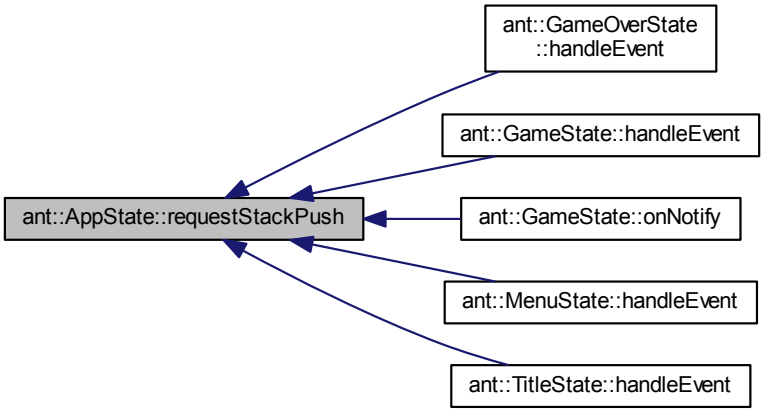
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

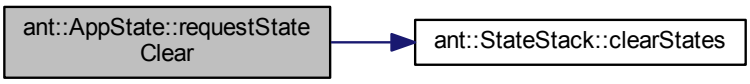


9.24.4.9 void ant::AppState::requestStateClear ( ) [protected],[inherited]

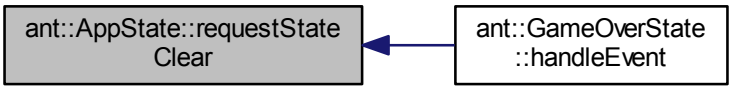
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.10 bool ant::GameState::update (sf::Time *dt*) [virtual]

Actualiza el juego.

Parameters

dt	sf::Time delta time

Implements ant::AppState.

Definition at line 25 of file GameState.cpp.

9.24.5 Member Data Documentation

9.24.5.1 std::shared_ptr<AssetManager> ant::GameState::assets [private]

Definition at line 59 of file GameState.hpp.

9.24.5.2 std::vector<std::unique_ptr<GUI::Button> > ant::GameState::buttons [private]

Definition at line 63 of file GameState.hpp.

9.24.5.3 int ant::GameState::currentLevel [private]

Definition at line 64 of file GameState.hpp.

9.24.5.4 std::shared_ptr<EventQueue> ant::GameState::eventQueue [private]

Definition at line 62 of file GameState.hpp.

9.24.5.5 std::string ant::GameState::font [private]

Definition at line 67 of file GameState.hpp.

9.24.5.6 std::shared_ptr<GameEventDispatcher> ant::GameState::gameEventDispatcher [private]

Definition at line 60 of file GameState.hpp.

9.24.5.7 float ant::GameState::GameSpeed [private]

Definition at line 66 of file GameState.hpp.

9.24.5.8 std::shared_ptr<Level> ant::GameState::level [private]

Definition at line 61 of file GameState.hpp.

9.24.5.9 int ant::GameState::totalLevels [private]

Definition at line 65 of file GameState.hpp.

9.24.5.10 sf::RenderWindow& ant::GameState::win [private]

Definition at line 58 of file GameState.hpp.

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

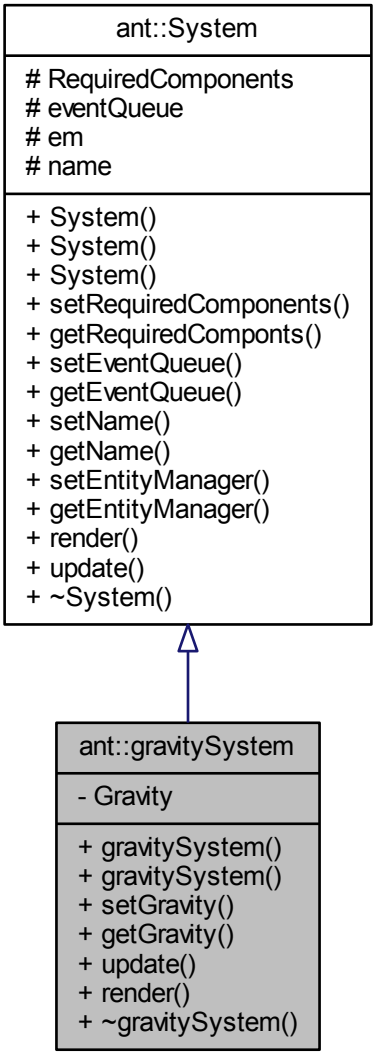
- GameState.hpp
- GameState.cpp

9.25 ant::gravitySystem Class Reference

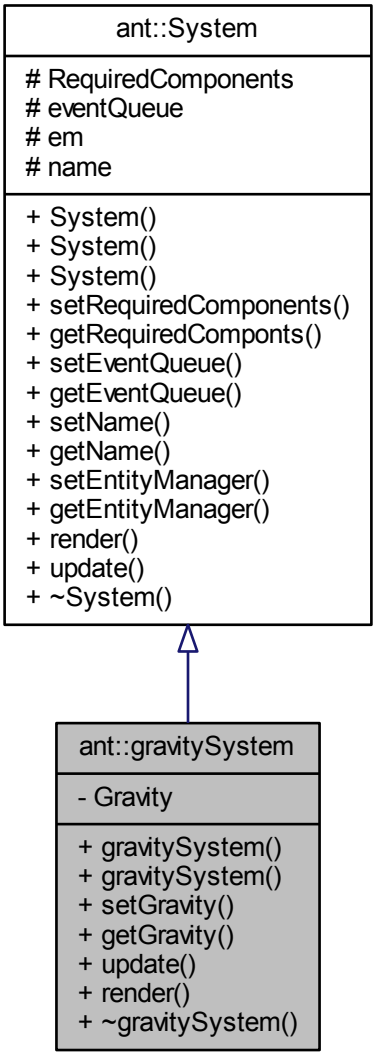
Sistema para aplicar la gravedad a las entidades.

```
#include <gravitySystem.hpp>
```

Inheritance diagram for ant::gravitySystem:



Collaboration diagram for ant::gravitySystem:



Public Member Functions

- gravitySystem ()

  Constructor por defecto.
- gravitySystem (float g)

  Overload de constructor para configurar la gravedad directamente.
- void setGravity (float g)

  Establece la gravedad que aplicara el sistema.
- float getGravity ()

  Obtiene la gravedad que aplica a las entidades.
- virtual void update (sf::Time dt)

  Metodo abstracto para actualizar entidades.

- virtual void render (sf::RenderWindow &win)

  No hace nada.
- ~gravitySystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

Private Attributes

- float Gravity

9.25.1 Detailed Description

Sistema para aplicar la gravedad a las entidades.

Aplica por defecto 0.8

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file gravitySystem.hpp.

9.25.2 Constructor & Destructor Documentation

9.25.2.1 ant::gravitySystem::gravitySystem ( )

Constructor por defecto.

Definition at line 4 of file gravitySystem.cpp.

9.25.2.2 ant::gravitySystem::gravitySystem (float *g*)

Overload de constructor para configurar la gravedad directamente.

Parameters

g	float gravedad a aplicar.

Definition at line 9 of file gravitySystem.cpp.

9.25.2.3 ant::gravitySystem::~gravitySystem ( )

Definition at line 29 of file gravitySystem.cpp.

9.25.3 Member Function Documentation

9.25.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.25.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.25.3.3 float ant::gravitySystem::getGravity ( ) [inline]

Obtiene la gravedad que aplica a las entidades.

Returns

float Gravedad que aplica.

Definition at line 35 of file gravitySystem.hpp.

9.25.3.4 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.25.3.5 long int ant::System::getRequiredComponts ( ) [inline],[inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.25.3.6 void ant::gravitySystem::render (sf::RenderWindow & *win*) [virtual]

No hace nada.

Implements ant::System.

Definition at line 26 of file gravitySystem.cpp.

9.25.3.7 void ant::System::setEntityManager (std::shared_ptr< **EntityManager** > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.25.3.8 void ant::System::setEventQueue (std::shared_ptr< **EventQueue** > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.25.3.9 void ant::gravitySystem::setGravity (float *g*)

Establece la gravedad que aplicara el sistema.

Parameters

g	float gravedad a aplicar.

Definition at line 14 of file gravitySystem.cpp.

9.25.3.10 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.25.3.11 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.25.3.12 void ant::gravitySystem::update (sf::Time *dt*) [virtual]

Metodo abstracto para actualizar entidades.

Parameters

dt	sf::Time delta time para los calculos.

Implements ant::System.

Definition at line 17 of file gravitySystem.cpp.

9.25.4 Member Data Documentation

9.25.4.1 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.25.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.25.4.3 float ant::gravitySystem::Gravity [private]

Definition at line 47 of file gravitySystem.hpp.

9.25.4.4 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.25.4.5 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

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

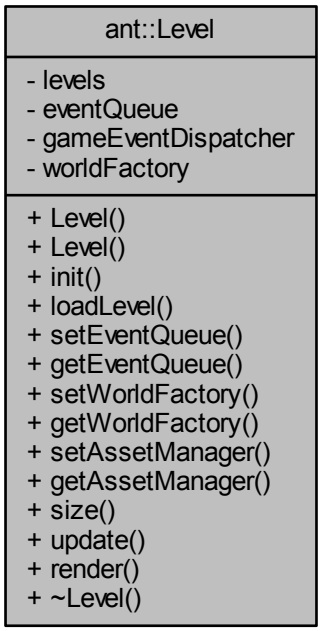
- gravitySystem.hpp
- gravitySystem.cpp

9.26 ant::Level Class Reference

Clase que tiene el nivel del juego.

```
#include <Level.hpp>
```

Collaboration diagram for ant::Level:



Public Member Functions

- Level (const std::string &assetsFilename, sf::FloatRect bounds, std::shared_ptr< GameEventDispatcher > ged)

  Constructor principal.
- Level (std::shared_ptr< AssetManager > am, sf::FloatRect bounds, std::shared_ptr< GameEventDispatcher > ged)

  Constructor secundario.
- void init (const sf::FloatRect &bounds)

  Inicializa el nivel.
- bool loadLevel (const sf::FloatRect &bounds, const std::string &filename)

  Carga el nivel de un fichero json.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la cola de eventos.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la cola de eventos.
- void setWorldFactory (std::shared_ptr< WorldFactory > wf)

  Setter para la fabrica de mundos.
- std::shared_ptr< WorldFactory > getWorldFactory ()

  Getter de la fabrica de mundos.
- void setAssetManager (std::shared_ptr< AssetManager > assets)

  Setter para el AssetManager.
- std::shared_ptr< AssetManager > getAssetManager ()

  Getter para el AssetManager.

- int size ()

  Devuelve el numero total de niveles que tiene.
- void update (long int id, sf::Time dt)

  Actualiza el mundo especificado.
- void render (long int id, sf::RenderWindow &win)

  Renderiza el mundo especificado.
- ~Level ()

Private Attributes

- std::shared_ptr< WorldManager > levels
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr
  < GameEventDispatcher > gameEventDispatcher
- std::shared_ptr< WorldFactory > worldFactory

9.26.1 Detailed Description

Clase que tiene el nivel del juego.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 19 of file Level.hpp.

9.26.2 Constructor & Destructor Documentation

9.26.2.1 ant::Level::Level (const std::string & *assetsFilename*, sf::FloatRect *bounds*, std::shared_ptr< GameEventDispatcher > *ged*)

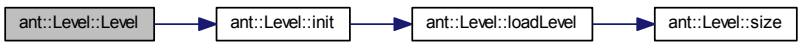
Constructor principal.

Parameters

assetsFilename	std::string fichero json a cargar. formato: AssetManager::loadAssets()
bounds	sf::FloatRect rectangulo con los limites del nivel.
ged	std::shared_ptr<GameEventDispatcher> para que se suscriban a los eventos.

Definition at line 4 of file Level.cpp.

Here is the call graph for this function:



9.26.2.2 ant::Level::Level (std::shared_ptr< AssetManager > *am*, sf::FloatRect *bounds*, std::shared_ptr< GameEventDispatcher > *ged*)

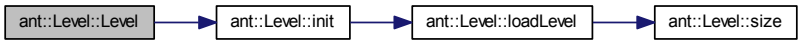
Constructor secundario.

Parameters

am	std::shared_ptr<AssetManager>
bounds	sf::FloatRect rectangulo con los limites del nivel.
ged	std::shared_ptr<GameEventDispatcher> para que se suscriban a los eventos.

Definition at line 12 of file Level.cpp.

Here is the call graph for this function:



9.26.2.3 ant::Level::~Level ( )

Definition at line 61 of file Level.cpp.

9.26.3 Member Function Documentation

9.26.3.1 std::shared_ptr<AssetManager> ant::Level::getAssetManager ( ) [inline]

Getter para el AssetManager.

Returns

std::shared_ptr<AssetManager>

Definition at line 78 of file Level.hpp.

9.26.3.2 std::shared_ptr<EventQueue> ant::Level::getEventQueue ( ) [inline]

Getter para la cola de eventos.

Returns

std::shared_ptr<EventQueue> Devuelve la cola de eventos.

Definition at line 58 of file Level.hpp.

9.26.3.3 std::shared_ptr<WorldFactory> ant::Level::getWorldFactory ( ) [inline]

Getter de la fabrica de mundos.

Returns

std::shared_ptr<WorldFactory>

Definition at line 68 of file Level.hpp.

9.26.3.4 void ant::Level::init (const sf::FloatRect & *bounds*)

Inicializa el nivel.

A los mundos iniciales al nivel.

Parameters

bounds	sf::FloatRect rectangulo con los limites del nivel.

Definition at line 20 of file Level.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

9.26.3.5 bool ant::Level::loadLevel (const sf::FloatRect & *bounds*, const std::string & *filename*)

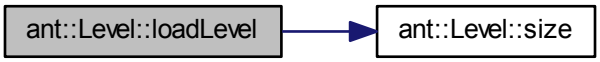
Carga el nivel de un fichero json.

Returns

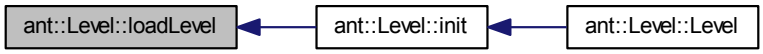
bool

Definition at line 24 of file Level.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.3.6 void ant::Level::render (long int *id*, sf::RenderWindow & *win*)

Renderiza el mundo especificado.

Parameters

id	long int Mundo a actualizar
win	sf::RenderWindow & ventana a la cual dibujar.

Definition at line 58 of file Level.cpp.

9.26.3.7 void ant::Level::setAssetManager (std::shared_ptr< **AssetManager** > *assets*)

Setter para el AssetManager.

Parameters

assets	std::shared_ptr<AssetManager>

Definition at line 52 of file Level.cpp.

9.26.3.8 void ant::Level::setEventQueue (std::shared_ptr< **EventQueue** > *eq*)

Setter para la cola de eventos.

Parameters

eq	std::shared_ptr<EventQueue> Cola para que los sistemas an los eventos que sucenden en el juego.

Definition at line 44 of file Level.cpp.

9.26.3.9 void ant::Level::setWorldFactory (std::shared_ptr< **WorldFactory** > *wf*)

Setter para la fabrica de mundos.

Parameters

wf	std::shared_ptr<WorldFactory>

Definition at line 49 of file Level.cpp.

9.26.3.10 int ant::Level::size ( ) [inline]

Devuelve el numero total de niveles que tiene.

Definition at line 82 of file Level.hpp.

Here is the caller graph for this function:

9.26.3.11 void ant::Level::update (long int *id*, sf::Time *dt*)

Actualiza el mundo especificado.

Actualiza el mundo especificado, si se pone un -1 se actualizaran todos.

Parameters

id	long int Mundo a actualizar
dt	sf::Time delta time.

Definition at line 55 of file Level.cpp.

9.26.4 Member Data Documentation

9.26.4.1 std::shared_ptr<EventQueue> ant::Level::eventQueue [private]

Definition at line 101 of file Level.hpp.

9.26.4.2 std::shared_ptr<GameEventDispatcher> ant::Level::gameEventDispatcher [private]

Definition at line 102 of file Level.hpp.

9.26.4.3 std::shared_ptr<WorldManager> ant::Level::levels [private]

Definition at line 100 of file Level.hpp.

9.26.4.4 std::shared_ptr<WorldFactory> ant::Level::worldFactory [private]

Definition at line 103 of file Level.hpp.

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

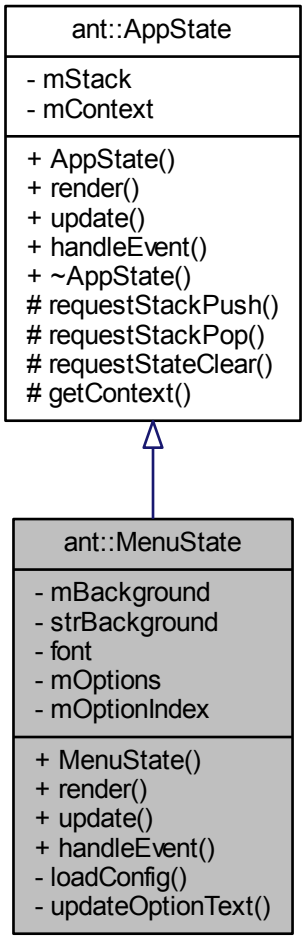
- Level.hpp
- Level.cpp

9.27 ant::MenuState Class Reference

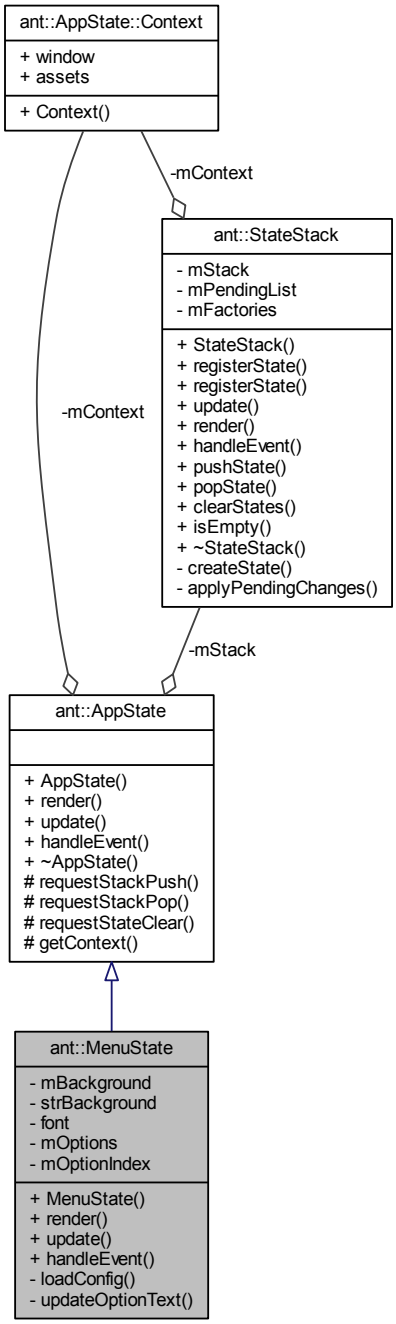
Clase de estado del menu.

```
#include <MenuState.hpp>
```

Inheritance diagram for ant::MenuState:



Collaboration diagram for ant::MenuState:



Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.

- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- MenuState (StateStack &stack, Context context)

  Constructor principal.
- virtual void render ()

  Renderiza el estado.
- virtual bool update (sf::Time dt)

  Actualiza el estado.
- virtual bool handleEvent (const sf::Event &event)

  Maneja la input.

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Types

- enum OptionNames { Play, Exit }

Private Member Functions

- void loadConfig (const std::string &filename)

  Carga la configuracion de este estado.
- void updateOptionText ()

Private Attributes

- sf::Sprite mBackground
- std::string strBackground
- std::string font
- std::vector< sf::Text > mOptions
- std::size_t mOptionIndex

9.27.1 Detailed Description

Clase de estado del menu.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 14 of file MenuState.hpp.

9.27.2 Member Typedef Documentation

9.27.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr [inherited]

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.27.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr [inherited]

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

9.27.3 Member Enumeration Documentation

9.27.3.1 enum ant::MenuState::OptionNames [private]

Enumerator

Play

Exit

Definition at line 44 of file MenuState.hpp.

9.27.4 Constructor & Destructor Documentation

9.27.4.1 ant::MenuState::MenuState (StateStack & *stack*, Context *context*)

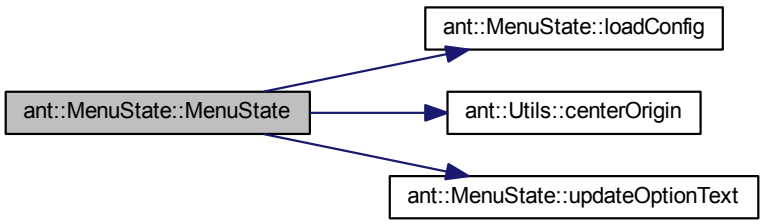
Constructor principal.

Parameters

stack	StateStack stack para cambiar el estado internamente.
context	Context contexto del estado.

Definition at line 4 of file MenuState.cpp.

Here is the call graph for this function:



9.27.5 Member Function Documentation

9.27.5.1 AppState::Context ant::AppState::getContext ( ) const [protected],[inherited]

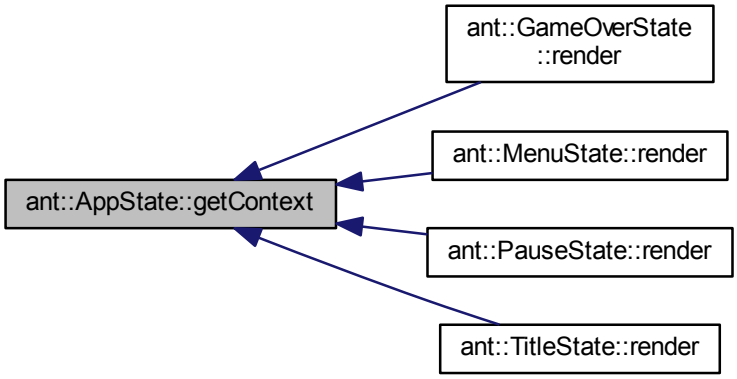
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



9.27.5.2 bool ant::MenuState::handleEvent (const sf::Event & *event*) [virtual]

Maneja la input.

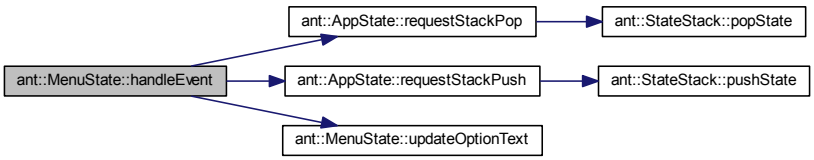
Parameters

event	sf::Event

Implements ant::AppState.

Definition at line 42 of file MenuState.cpp.

Here is the call graph for this function:



9.27.5.3 void ant::MenuState::loadConfig (const std::string & *filename*) [private]

Carga la configuracion de este estado.

Parameters

filename	std::string & fichero a cargar.

Definition at line 72 of file MenuState.cpp.

Here is the caller graph for this function:

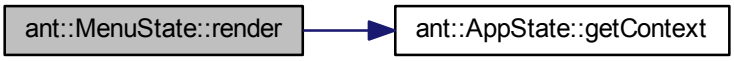
9.27.5.4 void ant::MenuState::render ( ) [virtual]

Renderiza el estado.

Implements ant::AppState.

Definition at line 31 of file MenuState.cpp.

Here is the call graph for this function:

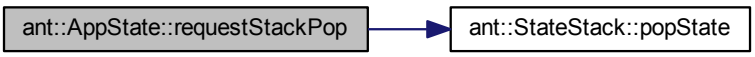


9.27.5.5 void ant::AppState::requestStackPop ( ) [protected],[inherited]

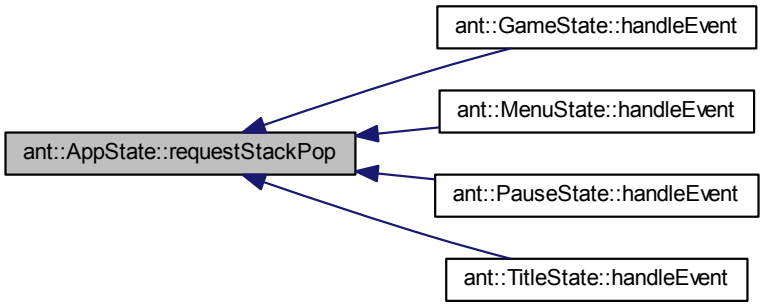
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.27.5.6 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected], [inherited]

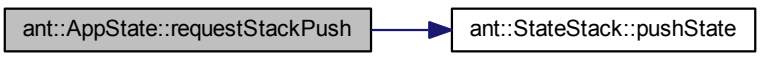
Pide al stack que a un estado nuevo.

Parameters

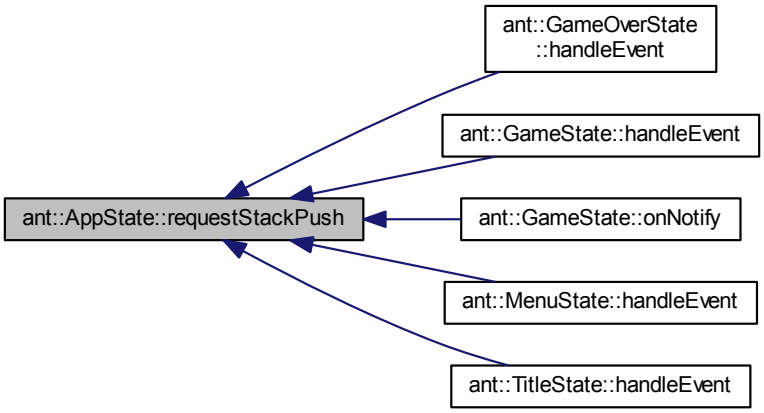
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

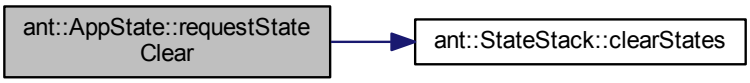


9.27.5.7 void ant::AppState::requestStateClear ( ) [protected],[inherited]

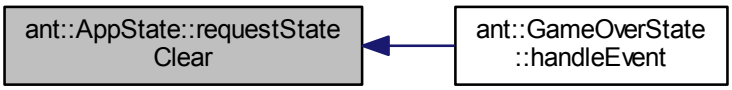
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.27.5.8 bool ant::MenuState::update (sf::Time *dt*) [virtual]

Actualiza el estado.

Parameters

dt	sf::Time

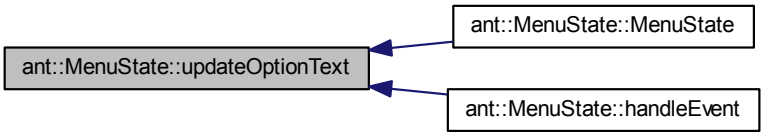
Implements ant::AppState.

Definition at line 39 of file MenuState.cpp.

9.27.5.9 void ant::MenuState::updateOptionText ( ) [private]

Definition at line 82 of file MenuState.cpp.

Here is the caller graph for this function:



9.27.6 Member Data Documentation

9.27.6.1 std::string ant::MenuState::font [private]

Definition at line 51 of file MenuState.hpp.

9.27.6.2 sf::Sprite ant::MenuState::mBackground [private]

Definition at line 49 of file MenuState.hpp.

9.27.6.3 std::size_t ant::MenuState::mOptionIndex [private]

Definition at line 53 of file MenuState.hpp.

9.27.6.4 std::vector<sf::Text> ant::MenuState::mOptions [private]

Definition at line 52 of file MenuState.hpp.

9.27.6.5 std::string ant::MenuState::strBackground [private]

Definition at line 50 of file MenuState.hpp.

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

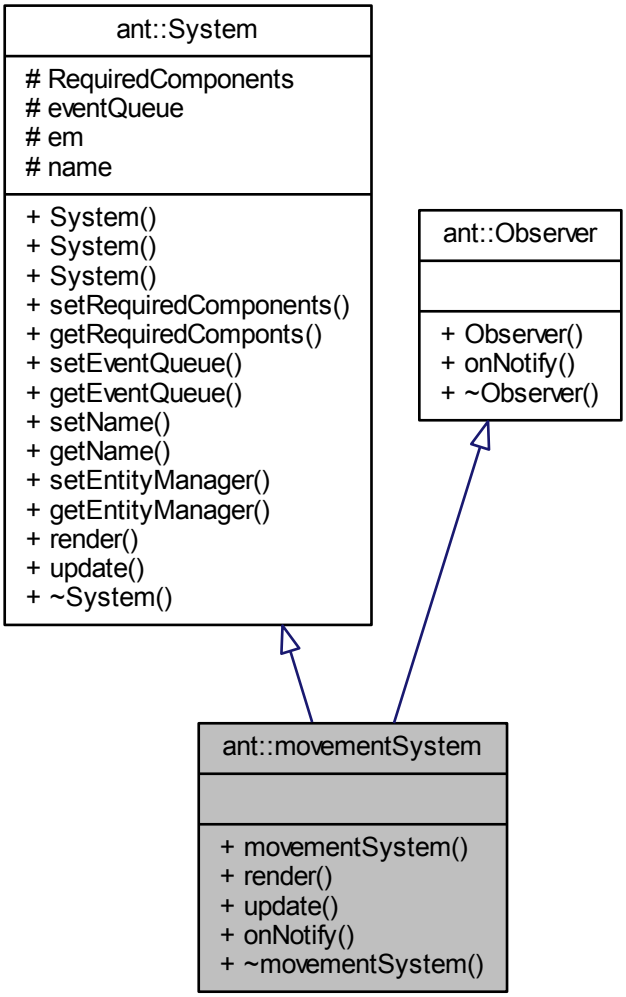
- MenuState.hpp
- MenuState.cpp

9.28 ant::movementSystem Class Reference

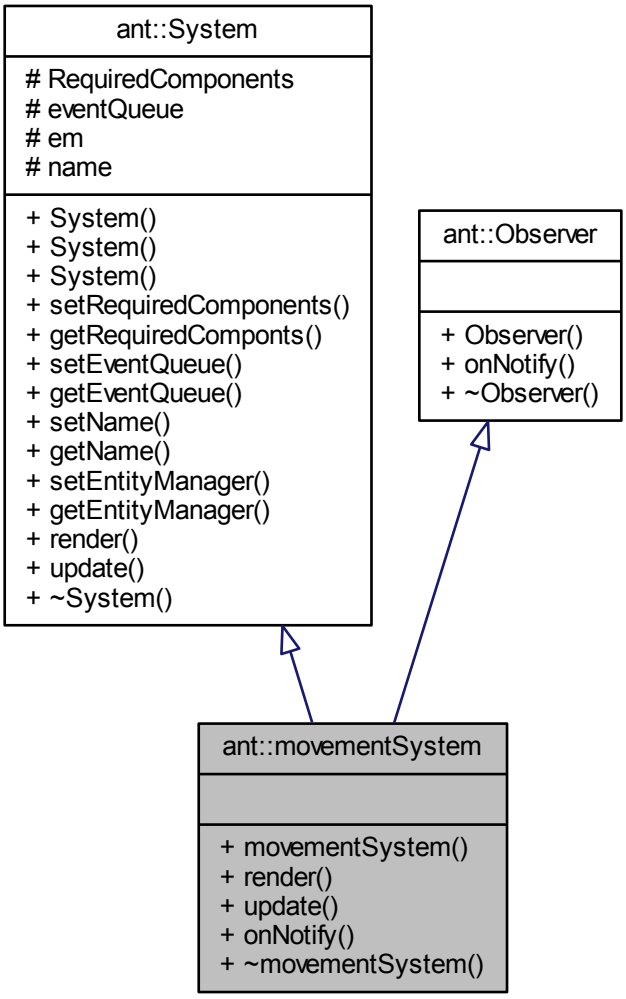
Sistema encargado de hacer que las entidades se muevan.

```
#include <movementSystem.hpp>
```

Inheritance diagram for ant::movementSystem:



Collaboration diagram for ant::movementSystem:



Public Member Functions

- movementSystem ()

  Constructor por defecto.
- virtual void render (sf::RenderWindow &win)

  No hace nada.
- virtual void update (sf::Time dt)

  Aplica la velocidad a todas las entidades con los componentes necesarios.
- void onNotify (std::shared_ptr< baseEvent > e)

  Cuando recibe Eventos COLLISION_EVENT parara a las entidades si se mueven.
- ~movementSystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

Getter de los requisitos para el sistema.

- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

9.28.1 Detailed Description

Sistema encargado de hacer que las entidades se muevan.

Las entidades se moveran si tienen el componente Velocity y Transform. Se moveran segun la rotacion que tengan y la velocidad actual. Maneja a las entidades segun los eventos que reciba del sistema de colision.

Eventos que maneja:

1. TERRAIN_COLLISION: cambiara la rotacion de las entidades
2. COLLISION_EVENT: ~

   Author

   Cristian Glez `Cristian.glez.m@gmail.com`

   Version

   0.1

Definition at line 22 of file movementSystem.hpp.

9.28.2 Constructor & Destructor Documentation

9.28.2.1 ant::movementSystem::movementSystem ( )

Constructor por defecto.

Definition at line 4 of file movementSystem.cpp.

9.28.2.2 ant::movementSystem::~movementSystem ( )

Definition at line 92 of file movementSystem.cpp.

9.28.3 Member Function Documentation

9.28.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.28.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.28.3.3 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.28.3.4 long int ant::System::getRequiredComponts ( ) [inline],[inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.28.3.5 void ant::movementSystem::onNotify (std::shared_ptr< baseEvent > *e*) [virtual]

Cuando recibe Eventos COLLISION_EVENT parara a las entidades si se mueven.

Parameters

e	std::shared_ptr<baseEvent> evento.

Implements ant::Observer.

Definition at line 8 of file movementSystem.cpp.

Here is the call graph for this function:

9.28.3.6 void ant::movementSystem::render (sf::RenderWindow & *win*) [virtual]

No hace nada.

Implements ant::System.

Definition at line 73 of file movementSystem.cpp.

9.28.3.7 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.28.3.8 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.28.3.9 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.28.3.10 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.28.3.11 void ant::movementSystem::update (sf::Time *dt*) [virtual]

Aplica la velocidad a todas las entidades con los componentes necesarios.

Parameters

dt	sf::Time delta time necesario para que se muevan siempre igual sin tener en cuenta los fps.

Todo agregar al calculo delta time

Implements ant::System.

Definition at line 76 of file movementSystem.cpp.

9.28.4 Member Data Documentation

9.28.4.1 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.28.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.28.4.3 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.28.4.4 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

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

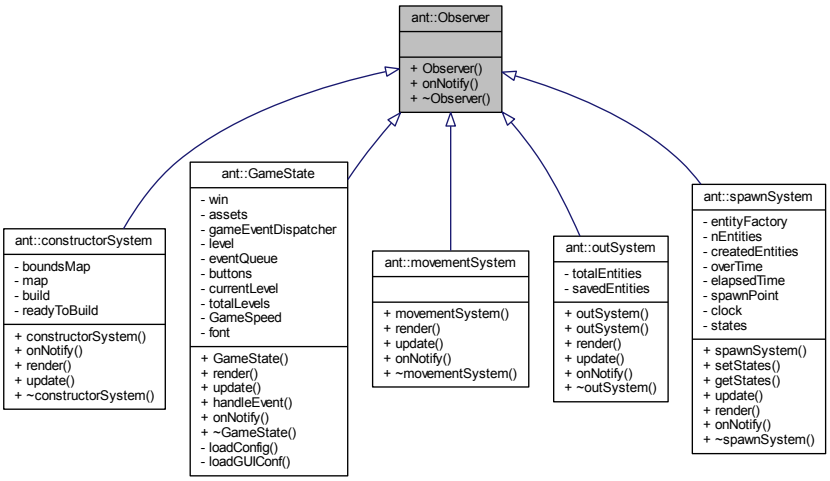
- movementSystem.hpp
- movementSystem.cpp

9.29 ant::Observer Class Reference

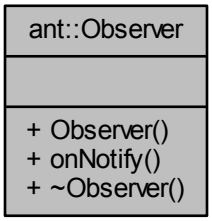
Interfaz para los observadores.

`#include <Observer.hpp>`

Inheritance diagram for ant::Observer:



Collaboration diagram for ant::Observer:



Public Member Functions

- Observer ()

  Constructor principal.
- virtual void onNotify (std::shared_ptr< baseEvent > e)=0

  Metodo para notificar al observador del evento.
- virtual ~Observer ()

9.29.1 Detailed Description

Interfaz para los observadores.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 11 of file Observer.hpp.

9.29.2 Constructor & Destructor Documentation

9.29.2.1 ant::Observer::Observer ( ) [inline]

Constructor principal.

Definition at line 16 of file Observer.hpp.

9.29.2.2 virtual ant::Observer::~Observer ( ) [inline], [virtual]

Definition at line 22 of file Observer.hpp.

9.29.3 Member Function Documentation

9.29.3.1 virtual void ant::Observer::onNotify (std::shared_ptr< baseEvent > *e*) [pure virtual]

Metodo para notificar al observador del evento.

Parameters

e	std::shared_ptr<baseEvent> evento.

Implemented in ant::spawnSystem, ant::GameState, ant::constructorSystem, ant::outSystem, and ant::movement↩System.

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

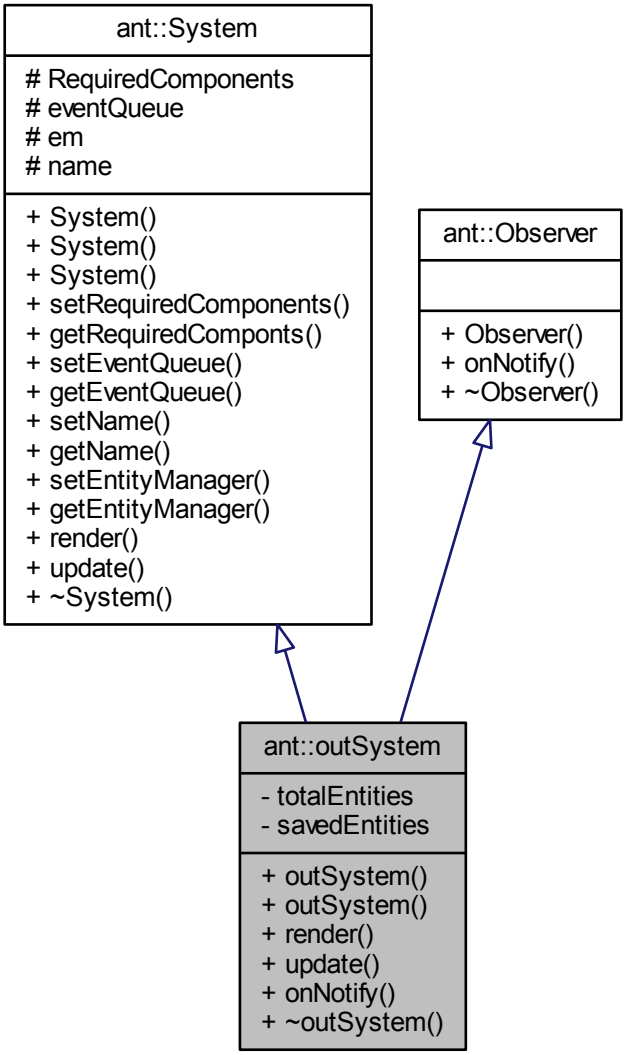
- Observer.hpp

9.30 ant::outSystem Class Reference

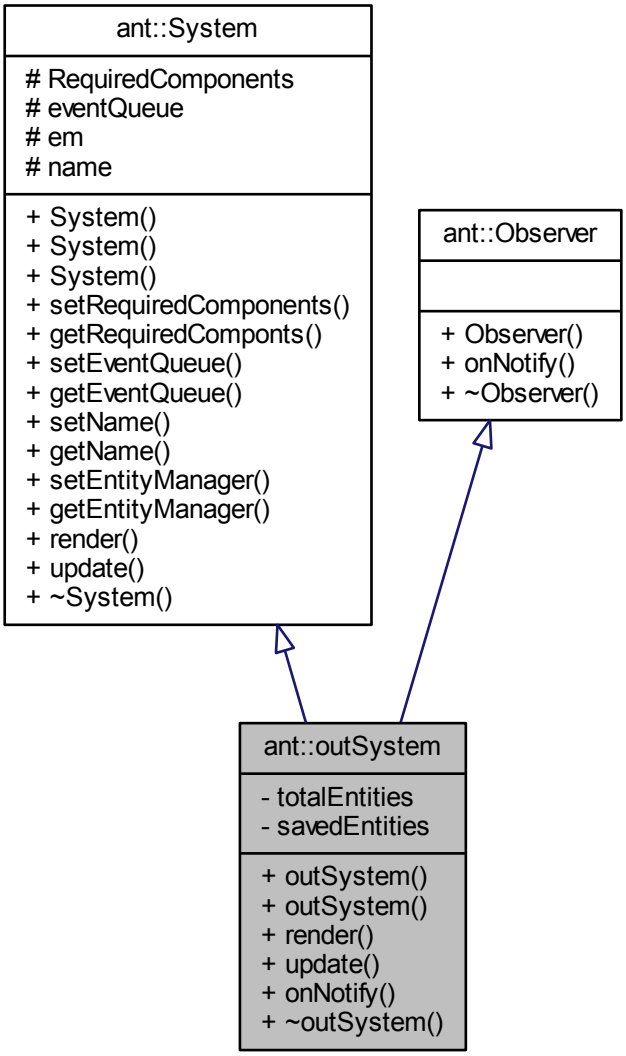
Sistema encargado de quitar a las entidades que chocan con las entidades con componente OUT.

```
#include <outSystem.hpp>
```

Inheritance diagram for ant::outSystem:



Collaboration diagram for ant::outSystem:



Public Member Functions

- outSystem ()

  Constructor por defecto.
- outSystem (int TotalEntities)

  Constructor con el numero totales de entidades de este nivel.
- void render (sf::RenderWindow &win)

  No hace nada.
- void update (sf::Time dt)

  Actualiza el sistema.
- void onNotify (std::shared_ptr< baseEvent > e)

  recibe los Out_MAP para eliminar estas entidades.

- ~outSystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

Private Attributes

- int totalEntities
- int savedEntities

9.30.1 Detailed Description

Sistema encargado de quitar a las entidades que chocan con las entidades con componente OUT.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file outSystem.hpp.

9.30.2 Constructor & Destructor Documentation

9.30.2.1 ant::outSystem::outSystem ( )

Constructor por defecto.

Definition at line 4 of file outSystem.cpp.

9.30.2.2 ant::outSystem::outSystem (int *TotalEntities*)

Constructor con el numero totales de entidades de este nivel.

Este Sistema envia un evento cuando el numero de entidades del level

Parameters

TotalEntities	int

Definition at line 8 of file outSystem.cpp.

9.30.2.3 ant::outSystem::~outSystem ( )

Definition at line 57 of file outSystem.cpp.

9.30.3 Member Function Documentation

9.30.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.30.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.30.3.3 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.30.3.4 long int ant::System::getRequiredComponts ( ) [inline],[inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.30.3.5 void ant::outSystem::onNotify (std::shared_ptr< baseEvent > *e*) [virtual]

recibe los Out_MAP para eliminar estas entidades.

Implements ant::Observer.

Definition at line 12 of file outSystem.cpp.

9.30.3.6 void ant::outSystem::render (sf::RenderWindow & *win*) [virtual]

No hace nada.

Implements ant::System.

Definition at line 34 of file outSystem.cpp.

9.30.3.7 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.30.3.8 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.30.3.9 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.30.3.10 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.30.3.11 void ant::outSystem::update (sf::Time *dt*) [virtual]

Actualiza el sistema.

Comprueba las entidades que han no tienen componentes para eliminarlas.

Implements ant::System.

Definition at line 37 of file outSystem.cpp.

9.30.4 Member Data Documentation

9.30.4.1 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.30.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.30.4.3 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.30.4.4 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

9.30.4.5 int ant::outSystem::savedEntities [private]

Definition at line 47 of file outSystem.hpp.

9.30.4.6 int ant::outSystem::totalEntities [private]

Definition at line 46 of file outSystem.hpp.

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

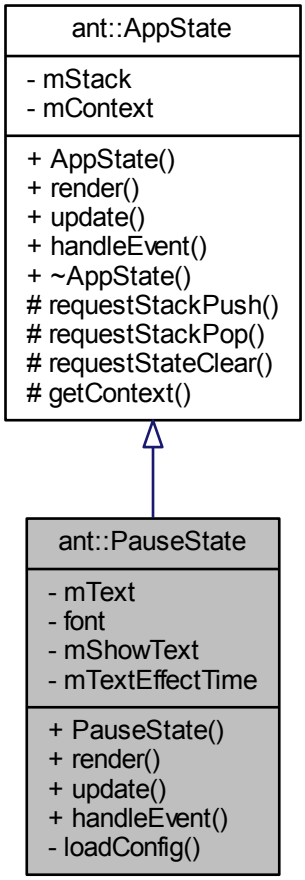
- outSystem.hpp
- outSystem.cpp

9.31 ant::PauseState Class Reference

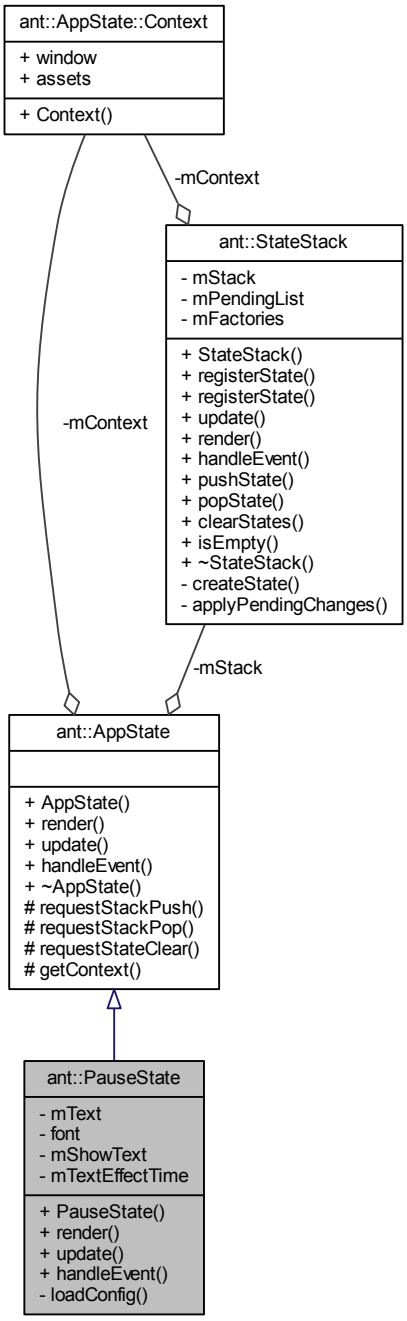
Estado de pausa.

```
#include <PauseState.hpp>
```

Inheritance diagram for ant::PauseState:



Collaboration diagram for ant::PauseState:



Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.

- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- PauseState (StateStack &stack, Context context)

  Constructor por defecto.
- virtual void render ()

  Renderiza el estado.
- virtual bool update (sf::Time dt)

  Actualiza.
- virtual bool handleEvent (const sf::Event &event)

  Maneja el input del teclado/raton.

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Member Functions

- void loadConfig (const std::string &filename)

  Carga la configuracion de este estado.

Private Attributes

- sf::Text mText
- std::string font
- bool mShowText
- sf::Time mTextEffectTime

9.31.1 Detailed Description

Estado de pausa.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 13 of file PauseState.hpp.

9.31.2 Member Typedef Documentation

9.31.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr [inherited]

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.31.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr [inherited]

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

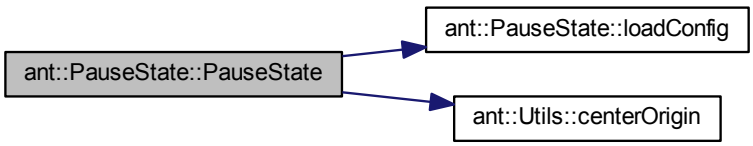
9.31.3 Constructor & Destructor Documentation

9.31.3.1 ant::PauseState::PauseState (StateStack & *stack*, Context *context*)

Constructor por defecto.

Definition at line 3 of file PauseState.cpp.

Here is the call graph for this function:



9.31.4 Member Function Documentation

9.31.4.1 AppState::Context ant::AppState::getContext ( ) const [protected],[inherited]

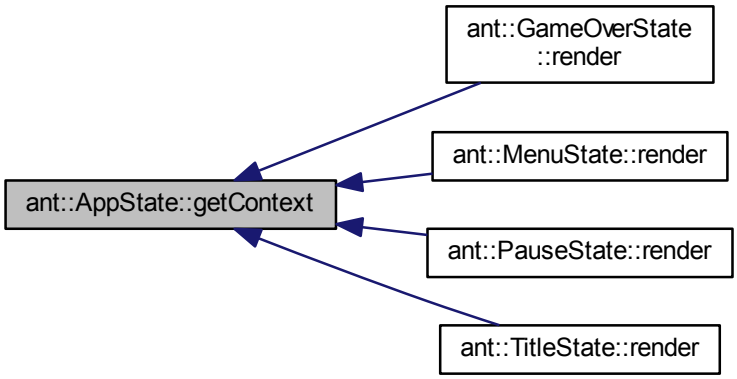
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



9.31.4.2 bool ant::PauseState::handleEvent (const sf::Event & *event*) [virtual]

Maneja el input del teclado/raton.

Parameters

event	sf::Event evento.

Implements ant::AppState.

Definition at line 35 of file PauseState.cpp.

Here is the call graph for this function:

9.31.4.3 void ant::PauseState::loadConfig (const std::string & *filename*) [private]

Carga la configuracion de este estado.

Definition at line 20 of file PauseState.cpp.

Here is the caller graph for this function:

9.31.4.4 void ant::PauseState::render ( ) [virtual]

Renderiza el estado.

Implements ant::AppState.

Definition at line 14 of file PauseState.cpp.

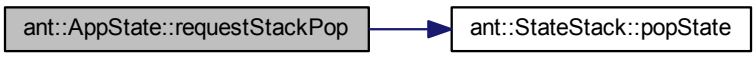
Here is the call graph for this function:

9.31.4.5 void ant::AppState::requestStackPop ( ) [protected],[inherited]

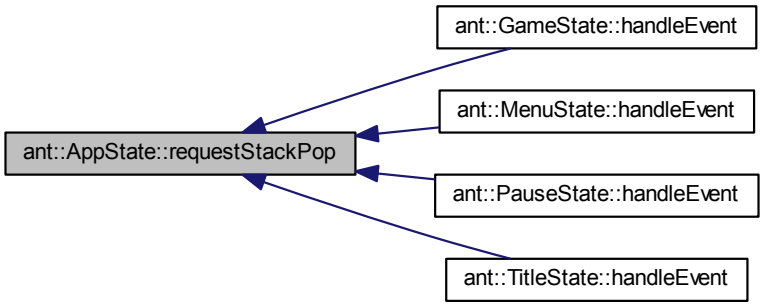
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.31.4.6 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected], [inherited]

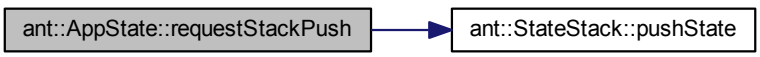
Pide al stack que a un estado nuevo.

Parameters

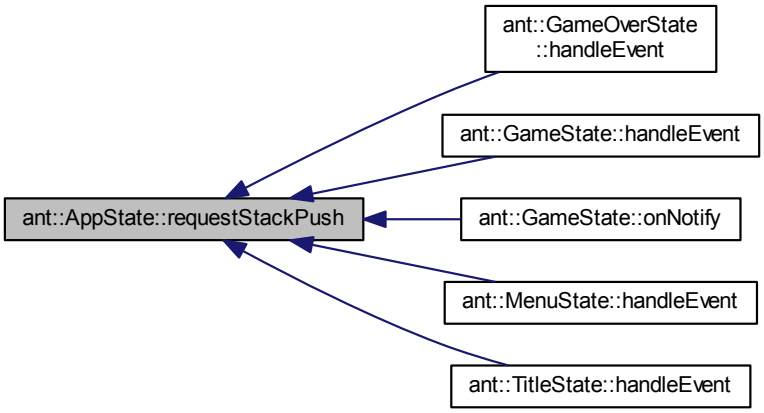
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

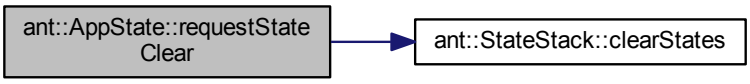


9.31.4.7 void ant::AppState::requestStateClear ( ) [protected],[inherited]

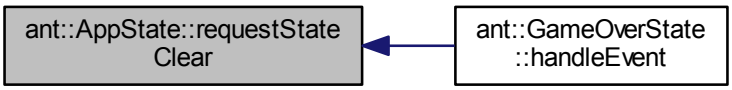
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.31.4.8 bool ant::PauseState::update (sf::Time *dt*) [virtual]

Actualiza.

Parameters

dt	sf::Time delta time

Implements ant::AppState.

Definition at line 27 of file PauseState.cpp.

9.31.5 Member Data Documentation

9.31.5.1 std::string ant::PauseState::font [private]

Definition at line 40 of file PauseState.hpp.

9.31.5.2 bool ant::PauseState::mShowText [private]

Definition at line 41 of file PauseState.hpp.

9.31.5.3 sf::Text ant::PauseState::mText [private]

Definition at line 39 of file PauseState.hpp.

9.31.5.4 sf::Time ant::PauseState::mTextEffectTime [private]

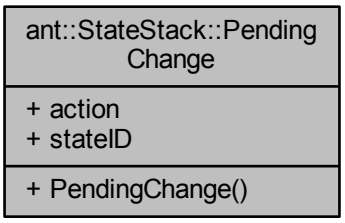
Definition at line 42 of file PauseState.hpp.

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

- PauseState.hpp
- PauseState.cpp

9.32 ant::StateStack::PendingChange Struct Reference

Collaboration diagram for ant::StateStack::PendingChange:



Public Member Functions

- PendingChange (Action action, AppStates::ID stateID=AppStates::None)

Public Attributes

- Action action
- AppStates::ID stateID

9.32.1 Detailed Description

Definition at line 93 of file StateStack.hpp.

9.32.2 Constructor & Destructor Documentation

9.32.2.1 ant::StateStack::PendingChange::PendingChange (Action *action*, AppStates::ID *stateID* = AppStates::None) [explicit]

Definition at line 63 of file StateStack.cpp.

9.32.3 Member Data Documentation

9.32.3.1 Action ant::StateStack::PendingChange::action

Definition at line 95 of file StateStack.hpp.

9.32.3.2 AppStates::ID ant::StateStack::PendingChange::stateID

Definition at line 96 of file StateStack.hpp.

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

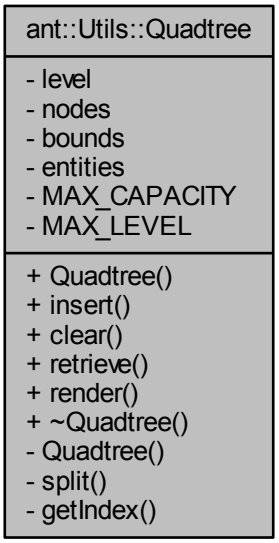
- StateStack.hpp
- StateStack.cpp

9.33 ant::Utils::Quadtree Class Reference

Clase para reducir el numero de test de collisiones.

```
#include <Quadtree.hpp>
```

Collaboration diagram for ant::Utils::Quadtree:



Public Types

- typedef std::list< Entity * > list

  Contenedor que usa el Quadtree.

Public Member Functions

- Quadtree (sf::FloatRect bounds)

  Constructor principal, necesita que se especifique el rectangulo que Ocupara.
- void insert (Entity *e)

  Agrega un puntero a una entidad, la posiciona en el nodo correcto, si no cabe en ningun nodo inferior se mantiene en el padre.
- void clear ()

  Limpia el quadtree y sus nodos.
- list retrieve (list &entities, Entity *e)

  Agrega las entidades del mismo sector que la entidad especificada a la lista(la misma entidad especificada es agregada tambien)
- void render (sf::RenderWindow &win)

  Metodo para ver las divisiones en la pantalla.
- ~Quadtree ()

Private Member Functions

- Quadtree (int level, sf::FloatRect bounds)

  Constructor privado para la division de los nodos.
- void split ()

Metodo para dividir el quadtree.

- int getIndex (Entity *e)

  Obtiene el indice(nodo donde esta) de la entidad especificada.

Private Attributes

- unsigned int level
- std::array< std::unique_ptr < Quadtree >, 4 > nodes
- sf::FloatRect bounds
- list entities

Static Private Attributes

- static const unsigned int MAX_CAPACITY = 10
- static const unsigned int MAX_LEVEL = 1000

9.33.1 Detailed Description

Clase para reducir el numero de test de collisiones.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Todo Posible cambio por Quadtree<T> tarda 1ms mas.

Definition at line 21 of file Quadtree.hpp.

9.33.2 Member Typedef Documentation

9.33.2.1 typedef std::list<Entity*> ant::Utils::Quadtree::list

Contenedor que usa el Quadtree.

Definition at line 26 of file Quadtree.hpp.

9.33.3 Constructor & Destructor Documentation

9.33.3.1 ant::Utils::Quadtree::Quadtree (sf::FloatRect *bounds*)

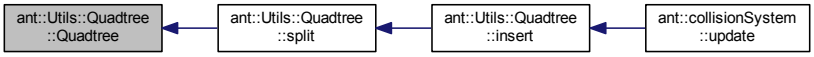
Constructor principal, necesita que se especifique el rectangulo que Ocupara.

Parameters

bounds	sf::FloatRect Rectangulo marcando la zona de este quadtree.

Definition at line 52 of file Quadtree.cpp.

Here is the caller graph for this function:



9.33.3.2 ant::Utils::Quadtree::~Quadtree ( )

Definition at line 126 of file Quadtree.cpp.

9.33.3.3 ant::Utils::Quadtree::Quadtree (int *level*, sf::FloatRect *bounds*) [private]

Constructor privado para la division de los nodos.

Parameters

level	int numero del nivel.
bounds	sf::FloatRect rectangulo de la division.

Definition at line 44 of file Quadtree.cpp.

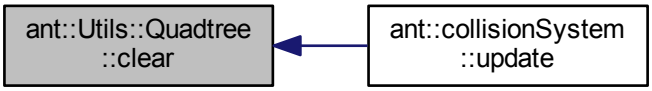
9.33.4 Member Function Documentation

9.33.4.1 void ant::Utils::Quadtree::clear ( )

Limpia el quadtree y sus nodos.

Definition at line 86 of file Quadtree.cpp.

Here is the caller graph for this function:



9.33.4.2 int ant::Utils::Quadtree::getIndex (Entity * *e*) [private]

Obtiene el indice(nodo donde esta) de la entidad especificada.

Parameters

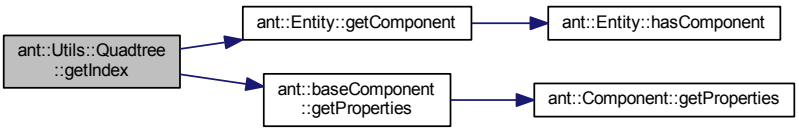
e	Entity∗ Entidad para sacar su indice.

Returns

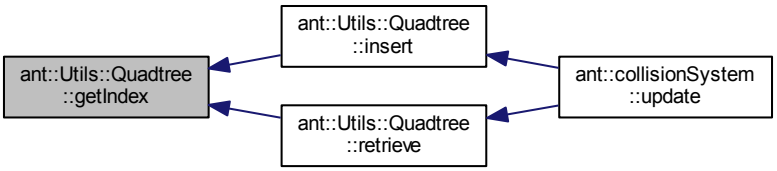
int devuelve el indice(0-3) -1 si esta en el mismo.

Definition at line 16 of file Quadtree.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.33.4.3 void ant::Utils::Quadtree::insert (Entity ∗ *e*)

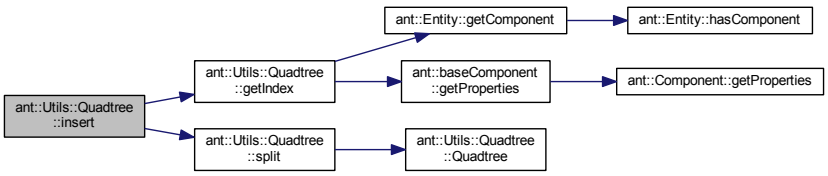
Agrega un puntero a una entidad, la posiciona en el nodo correcto, si no cabe en ningun nodo inferior se mantiene en el padre.

Parameters

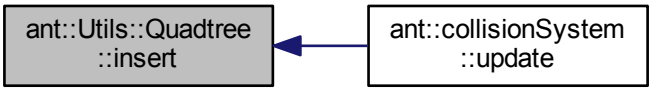
e	Entity ∗

Definition at line 60 of file Quadtree.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.33.4.4 void ant::Utils::Quadtree::render (sf::RenderWindow & *win*)

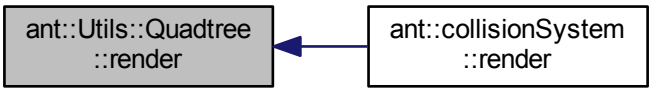
Metodo para ver las divisiones en la pantalla.

Parameters

win	sf::RenderWindow& ventana a la cual dibujar.

Definition at line 105 of file Quadtree.cpp.

Here is the caller graph for this function:



9.33.4.5 Quadtree::list ant::Utils::Quadtree::retrieve (list & *entities*, Entity * *e*)

Agrega las entidades del mismo sector que la entidad especificada a la lista(la misma entidad especificada es agregada tambien)

Parameters

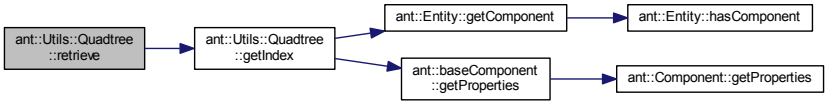
entities	Quadtree::list& lista a la que agregan las entidades del mismo sector de la especificada.
e	Entity * Entidad a comprobar.

Returns

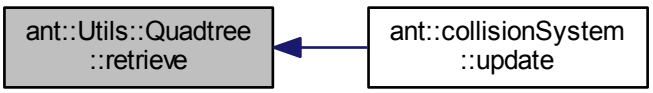
Quadtree::list lista de entidades

Definition at line 95 of file Quadtree.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



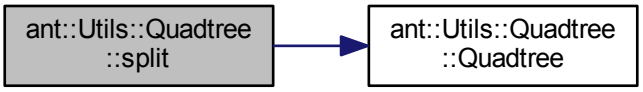
9.33.4.6 void ant::Utils::Quadtree::split ( ) [private]

Metodo para dividir el quadtree.

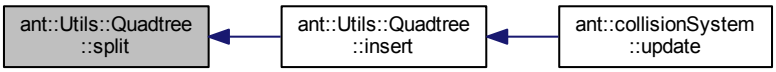
Inicializa los nodos del quadtree y se empiezan a usar.

Definition at line 6 of file Quadtree.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.33.5 Member Data Documentation

9.33.5.1 sf::FloatRect ant::Utils::Quadtree::bounds [private]

Definition at line 66 of file Quadtree.hpp.

9.33.5.2 list ant::Utils::Quadtree::entities [private]

Definition at line 67 of file Quadtree.hpp.

9.33.5.3 unsigned int ant::Utils::Quadtree::level [private]

Definition at line 64 of file Quadtree.hpp.

9.33.5.4 const unsigned int ant::Utils::Quadtree::MAX_CAPACITY = 10 [static], [private]

Definition at line 62 of file Quadtree.hpp.

9.33.5.5 const unsigned int ant::Utils::Quadtree::MAX_LEVEL = 1000 [static], [private]

Definition at line 63 of file Quadtree.hpp.

9.33.5.6 std::array<std::unique_ptr<Quadtree>,4> ant::Utils::Quadtree::nodes [private]

Definition at line 65 of file Quadtree.hpp.

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

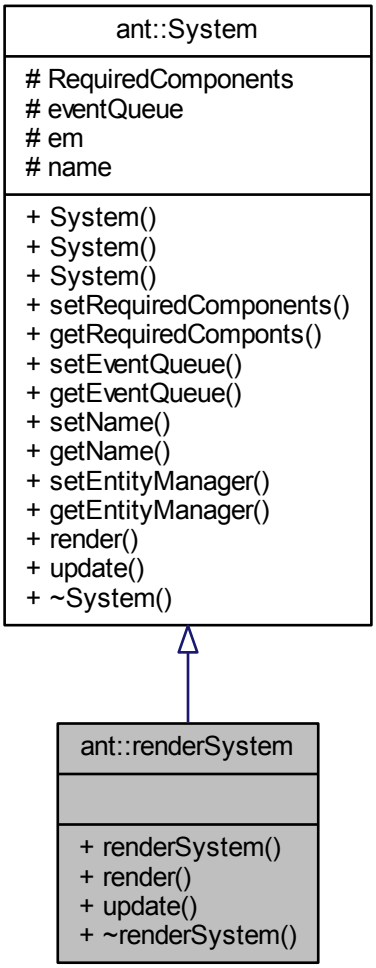
- Quadtree.hpp
- Quadtree.cpp

9.34 ant::renderSystem Class Reference

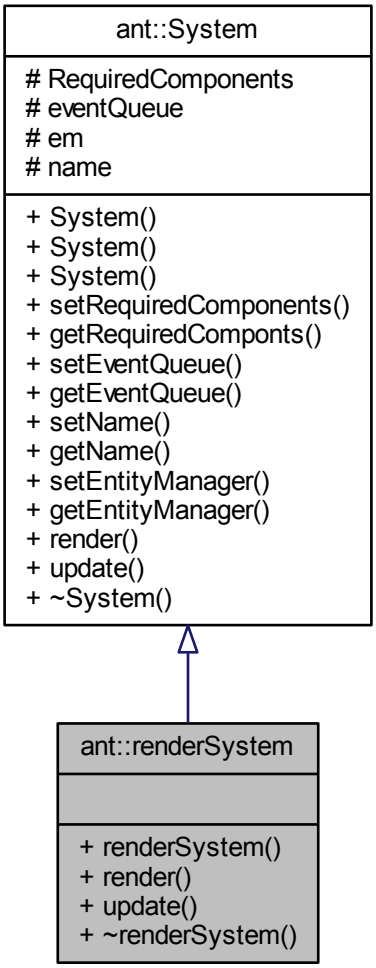
Sistema encargado de mostrar las entidades que tengan los componentes necesarios.

```
#include <renderSystem.hpp>
```

Inheritance diagram for ant::renderSystem:



Collaboration diagram for ant::renderSystem:



Public Member Functions

- renderSystem ()

  Constructor por defecto.
- virtual void render (sf::RenderWindow &win)

  Renderiza las Entity con los componentes Transform y Sprite, Destructable.
- virtual void update (sf::Time dt)

  No hace nada.
- ~renderSystem ()
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.

- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

9.34.1 Detailed Description

Sistema encargado de mostrar las entidades que tengan los componentes necesarios.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file renderSystem.hpp.

9.34.2 Constructor & Destructor Documentation

9.34.2.1 ant::renderSystem::renderSystem ( )

Constructor por defecto.

Definition at line 3 of file renderSystem.cpp.

9.34.2.2 ant::renderSystem::~renderSystem ( )

Definition at line 37 of file renderSystem.cpp.

9.34.3 Member Function Documentation

9.34.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.34.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline], [inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.34.3.3 std::string ant::System::getName ( ) const [inline], [inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.34.3.4 long int ant::System::getRequiredComponts ( ) [inline], [inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.34.3.5 void ant::renderSystem::render (sf::RenderWindow & *win*) [virtual]

Renderiza las Entity con los componentes Transform y Sprite, Destructable.

Parameters

win	sf::RenderWindow & Ventana a la cual dibujar.

Implements ant::System.

Definition at line 7 of file renderSystem.cpp.

9.34.3.6 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.34.3.7 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.34.3.8 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.34.3.9 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.34.3.10 void ant::renderSystem::update (sf::Time *dt*) [virtual]

No hace nada.

Implements ant::System.

Definition at line 34 of file renderSystem.cpp.

9.34.4 Member Data Documentation

9.34.4.1 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.34.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.34.4.3 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.34.4.4 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

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

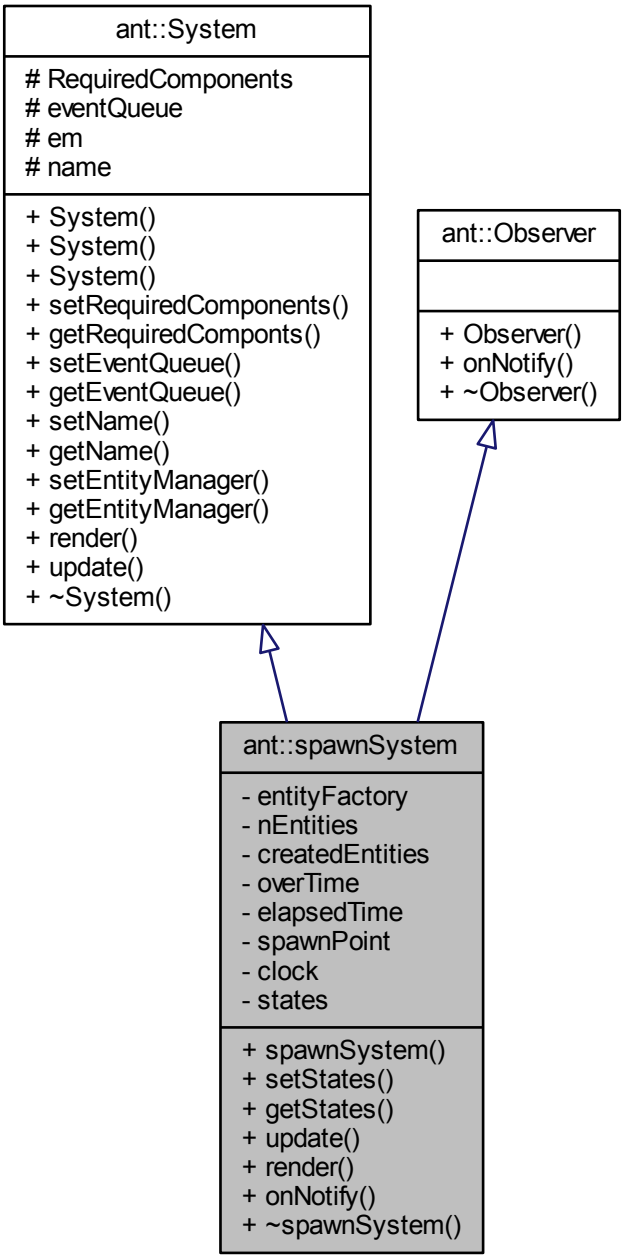
- renderSystem.hpp
- renderSystem.cpp

9.35 ant::spawnSystem Class Reference

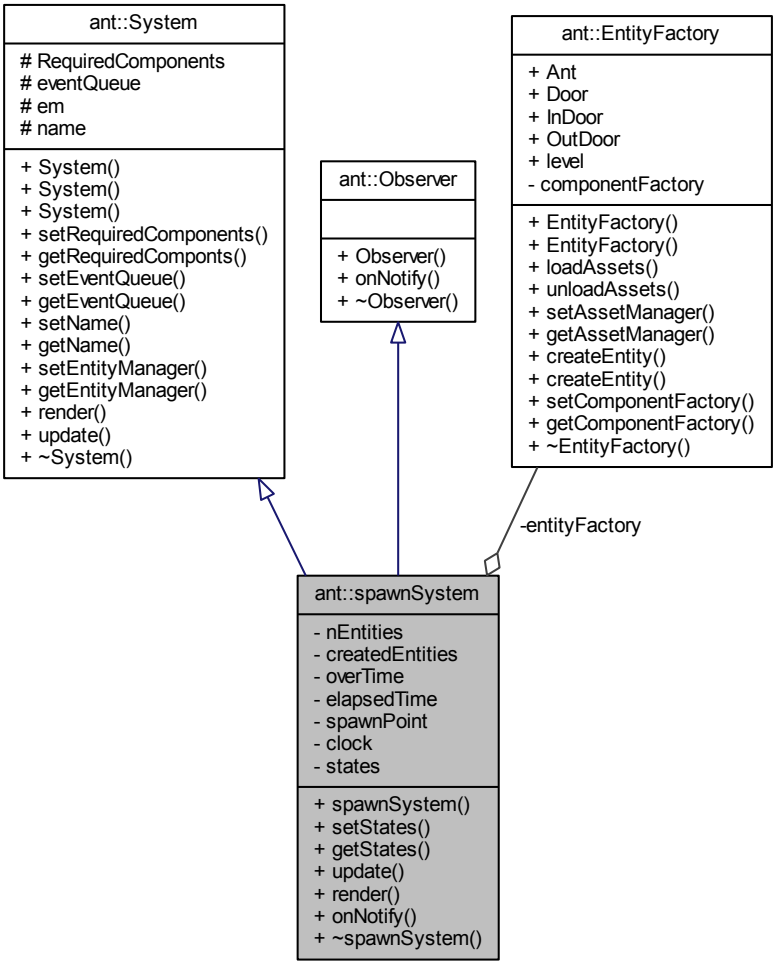
Sistema encargado de crear las entidades sobre el tiempo establecido.

```
#include <spawnSystem.hpp>
```

Inheritance diagram for ant::spawnSystem:



Collaboration diagram for ant::spawnSystem:



Public Member Functions

- spawnSystem (int nEntities, EntityFactory *ef, sf::Time ot, sf::Vector2f spawnPoint, long int state)

  constructor por defecto.
- void setStates (long int state)

  Establece el estado por defecto que pondra a las entidades.
- const long int getStates () const

  Obtiene los estados que pone por defecto al crear las entidades.
- void update (sf::Time dt)

  Cada x segundos crea un entidad en el sitio establecido.
- void render (sf::RenderWindow &win)

  no hace nada.
- void onNotify (std::shared_ptr< baseEvent > e)

  Recibe eventos para cambiar el tiempo o el numero de entidades que tiene que crear.
- ~spawnSystem ()
- void setRequiredComponents (long int req)

Setter para los requisitos necesarios.

- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

Private Attributes

- EntityFactory * entityFactory
- int nEntities
- int createdEntities
- sf::Time overTime
- sf::Time elapsedTime
- sf::Vector2f spawnPoint
- sf::Clock clock
- long int states

9.35.1 Detailed Description

Sistema encargado de crear las entidades sobre el tiempo establecido.

Crea las entidades en la posicion de las entidades con ComponentsMask::COMPONENT_IN.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 18 of file spawnSystem.hpp.

9.35.2 Constructor & Destructor Documentation

9.35.2.1 ant::spawnSystem::spawnSystem (int *nEntities*, EntityFactory * *ef*, sf::Time *ot*, sf::Vector2f *spawnPoint*, long int *state*)

constructor por defecto.

Parameters

nEntities	int numero de entidades a crear.
ot	sf::Time Tiempo para crear las entidades cada X tiempo.
ef	EntityFactory * Para crear las entidades.
spawnPoint	sf::Vector2f sitio donde saldran las entidades.
state	long int Estados iniciales para cuando las entidades son creadas.

Definition at line 4 of file spawnSystem.cpp.

9.35.2.2 ant::spawnSystem::~spawnSystem ( )

Definition at line 56 of file spawnSystem.cpp.

9.35.3 Member Function Documentation

9.35.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline],[inherited]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.35.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline],[inherited]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.35.3.3 std::string ant::System::getName ( ) const [inline],[inherited]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.35.3.4 long int ant::System::getRequiredComponts ( ) [inline],[inherited]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.35.3.5 const long int ant::spawnSystem::getStates ( ) const [inline]

Obtiene los estados que pone por defecto al crear las entidades.

Returns

long int

Definition at line 41 of file spawnSystem.hpp.

9.35.3.6 void ant::spawnSystem::onNotify (std::shared_ptr< baseEvent > *e*) [virtual]

Recibe eventos para cambiar el tiempo o el numero de entidades que tiene que crear.

Parameters

e	std::shared_ptr<baseEvent>

Implements ant::Observer.

Definition at line 38 of file spawnSystem.cpp.

9.35.3.7 void ant::spawnSystem::render (sf::RenderWindow & *win*) [virtual]

no hace nada.

Implements ant::System.

Definition at line 35 of file spawnSystem.cpp.

9.35.3.8 void ant::System::setEntityManager (std::shared_ptr< EntityManager > *em*) [inherited]

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.35.3.9 void ant::System::setEventQueue (std::shared_ptr< EventQueue > *eq*) [inherited]

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.35.3.10 void ant::System::setName (std::string *name*) [inherited]

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.35.3.11 void ant::System::setRequiredComponents (long int *req*) [inherited]

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.35.3.12 void ant::spawnSystem::setStates (long int *state*)

Establece el estado por defecto que pondra a las entidades.

Parameters

state	long int estados de las entidades `setStates(States::FALLING | STATES::ONFIRE);`

Definition at line 19 of file spawnSystem.cpp.

9.35.3.13 void ant::spawnSystem::update (sf::Time *dt*) [virtual]

Cada x segundos crea un entidad en el sitio establecido.

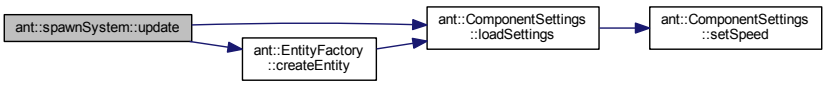
Parameters

dt	sf::Time delta time

Implements ant::System.

Definition at line 22 of file spawnSystem.cpp.

Here is the call graph for this function:



9.35.4 Member Data Documentation

9.35.4.1 sf::Clock ant::spawnSystem::clock [private]

Definition at line 65 of file spawnSystem.hpp.

9.35.4.2 int ant::spawnSystem::createdEntities [private]

Definition at line 61 of file spawnSystem.hpp.

9.35.4.3 sf::Time ant::spawnSystem::elapsedTime [private]

Definition at line 63 of file spawnSystem.hpp.

9.35.4.4 std::shared_ptr<EntityManager> ant::System::em [protected],[inherited]

Definition at line 86 of file System.hpp.

9.35.4.5 EntityFactory* ant::spawnSystem::entityFactory [private]

Definition at line 59 of file spawnSystem.hpp.

9.35.4.6 std::shared_ptr<EventQueue> ant::System::eventQueue [protected],[inherited]

Definition at line 85 of file System.hpp.

9.35.4.7 std::string ant::System::name [protected],[inherited]

Definition at line 87 of file System.hpp.

9.35.4.8 int ant::spawnSystem::nEntities [private]

Definition at line 60 of file spawnSystem.hpp.

9.35.4.9 sf::Time ant::spawnSystem::overTime [private]

Definition at line 62 of file spawnSystem.hpp.

9.35.4.10 long int ant::System::RequiredComponents [protected],[inherited]

Definition at line 84 of file System.hpp.

9.35.4.11 sf::Vector2f ant::spawnSystem::spawnPoint [private]

Definition at line 64 of file spawnSystem.hpp.

9.35.4.12 long int ant::spawnSystem::states [private]

Definition at line 66 of file spawnSystem.hpp.

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

- spawnSystem.hpp
- spawnSystem.cpp

9.36 States Class Reference

Estados de las entidades.

```
#include <States.hpp>
```

Collaboration diagram for States:

Static Public Attributes

- static const long int NONE = 0
- static const long int GROUND = 1 << 0
- static const long int FALLING = 1 << 1
- static const long int CLIMBING = 1 << 2
- static const long int BUILDING = 1 << 3
- static const long int ONFIRE = 1 << 4
- static const long int SAVED = 1 << 5
- static const long int UNSAVED = 1 << 6

9.36.1 Detailed Description

Estados de las entidades.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 9 of file States.hpp.

9.36.2 Member Data Documentation

9.36.2.1 const long int States::BUILDING = 1 << 3 [static]

Definition at line 15 of file States.hpp.

9.36.2.2 const long int States::CLIMBING = 1 << 2 [static]

Definition at line 14 of file States.hpp.

9.36.2.3 const long int States::FALLING = 1 << 1 [static]

Definition at line 13 of file States.hpp.

9.36.2.4 const long int States::GROUND = 1 << 0 [static]

Definition at line 12 of file States.hpp.

9.36.2.5 const long int States::NONE = 0 [static]

Definition at line 11 of file States.hpp.

9.36.2.6 const long int States::ONFIRE = 1 << 4 [static]

Definition at line 16 of file States.hpp.

9.36.2.7 const long int States::SAVED = 1 << 5 [static]

Definition at line 17 of file States.hpp.

9.36.2.8 const long int States::UNSAVED = 1 << 6 [static]

Definition at line 18 of file States.hpp.

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

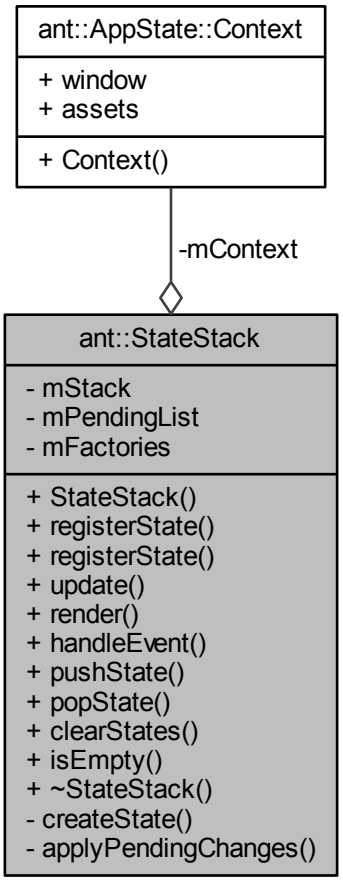
- States.hpp

9.37 ant::StateStack Class Reference

Clase para almacenar los estados del juego.

```
#include <StateStack.hpp>
```

Collaboration diagram for ant::StateStack:



Classes

- struct PendingChange

Public Types

- enum Action { Push, Pop, Clear }

  enumeracion para efectuar acciones.

Public Member Functions

- StateStack (AppState::Context context)

  Constructor principal.

- template<typename T >
  void registerState (AppStates::ID stateID)

  Registra un estado en la fabrica, el estado tiene que tener 2 argumentos.

- template<typename T >
  void registerState (AppStates::ID stateID, std::string msg)

  Registra un estado en la fabrica, el estado tiene que tener 3 argumentos.
- void update (sf::Time dt)

  Actualiza los estados que tiene.
- void render ()

  Renderiza los estados.
- void handleEvent (const sf::Event &event)

  maneja la input de los estados(teclado/ raton).
- void pushState (AppStates::ID stateID)

  A un estado al Stack.
- void popState ()

  Elimina el ultimo estado ado.
- void clearStates ()

  Elimina todos los estados.
- bool isEmpty () const

  Devuelve si el Stack esta vacio.
- ~StateStack ()

Private Member Functions

- AppState::uPtr createState (AppStates::ID stateID)

  Crea el estado especificado.
- void applyPendingChanges ()

  Aplica los cambios pendientes al Stack.

Private Attributes

- std::vector< AppState::uPtr > mStack
- std::vector< PendingChange > mPendingList
- AppState::Context mContext
- std::map< AppStates::ID,
  std::function< AppState::uPtr()> > mFactories

9.37.1 Detailed Description

Clase para almacenar los estados del juego.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 18 of file StateStack.hpp.

9.37.2 Member Enumeration Documentation

9.37.2.1 enum ant::StateStack::Action

enumeracion para efectuar acciones.

Enumerator

Push

Pop

Clear

Definition at line 23 of file StateStack.hpp.

9.37.3 Constructor & Destructor Documentation

9.37.3.1 ant::StateStack::StateStack (AppState::Context *context*) [explicit]

Constructor principal.

Parameters

context	AppState::Context contexto de ejecucion.

Definition at line 4 of file StateStack.cpp.

9.37.3.2 ant::StateStack::~StateStack ( )

Definition at line 66 of file StateStack.cpp.

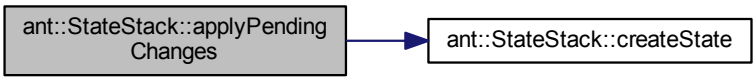
9.37.4 Member Function Documentation

9.37.4.1 void ant::StateStack::applyPendingChanges ( ) [private]

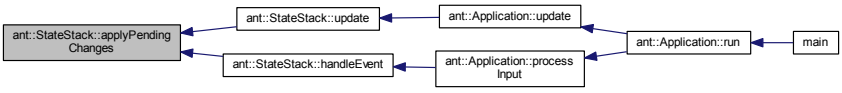
Aplica los cambios pendientes al Stack.

Definition at line 47 of file StateStack.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.37.4.2 void ant::StateStack::clearStates ( )

Elimina todos los estados.

Definition at line 36 of file StateStack.cpp.

Here is the caller graph for this function:

9.37.4.3 AppState::uPtr ant::StateStack::createState (AppStates::ID *stateID*) [private]

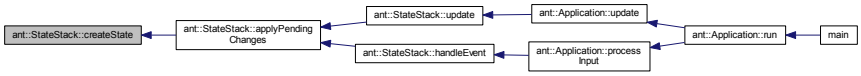
Crea el estado especificado.

Parameters

stateID	AppStates::ID

Definition at line 42 of file StateStack.cpp.

Here is the caller graph for this function:



9.37.4.4 void ant::StateStack::handleEvent (const sf::Event & *event*)

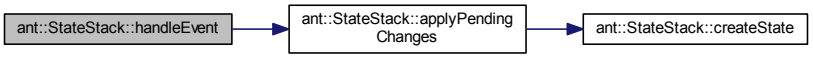
maneja la input de los estados(teclado/ raton).

Parameters

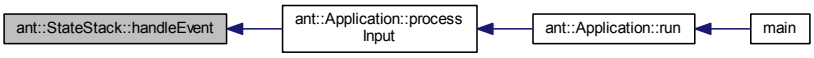
event	sf::Event

Definition at line 22 of file StateStack.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.37.4.5 bool ant::StateStack::isEmpty ( ) const

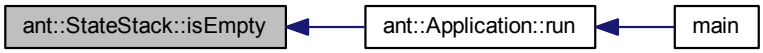
Devuelve si el Stack esta vacio.

Returns

bool

Definition at line 39 of file StateStack.cpp.

Here is the caller graph for this function:

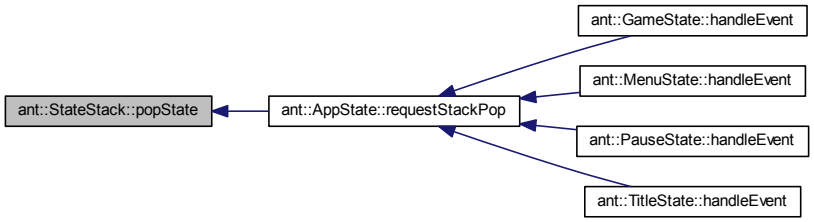


9.37.4.6 void ant::StateStack::popState ( )

Elimina el ultimo estado ado.

Definition at line 33 of file StateStack.cpp.

Here is the caller graph for this function:



9.37.4.7 void ant::StateStack::pushState (AppStates::ID *stateID*)

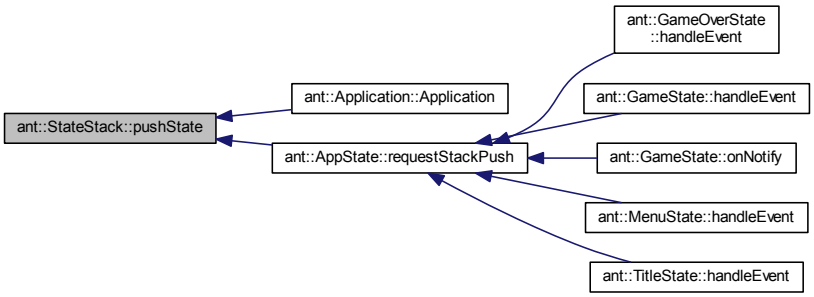
A un estado al Stack.

Parameters

stateID	AppStates::ID

Definition at line 30 of file StateStack.cpp.

Here is the caller graph for this function:



9.37.4.8 template<typename T > void ant::StateStack::registerState (AppStates::ID *stateID*)

Registra un estado en la fabrica, el estado tiene que tener 2 argumentos.

Template Parameters

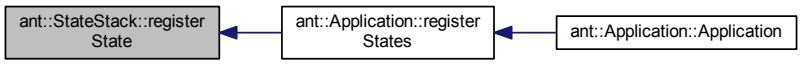
T	clase del estado a crear.

Parameters

stateID	AppState::ID estado a crear.

Definition at line 105 of file StateStack.hpp.

Here is the caller graph for this function:



9.37.4.9 template<typename T > void ant::StateStack::registerState (AppStates::ID *stateID*, std::string *msg*)

Registra un estado en la fabrica, el estado tiene que tener 3 argumentos.

Template Parameters

T	clase del estado a crear.

Parameters

stateID	AppState::ID estado a crear,
msg	std::string mensaje que mostrara el estado.

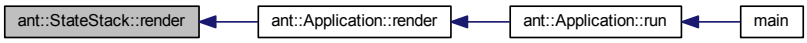
Definition at line 111 of file StateStack.hpp.

9.37.4.10 void ant::StateStack::render ( )

Renderiza los estados.

Definition at line 17 of file StateStack.cpp.

Here is the caller graph for this function:



9.37.4.11 void ant::StateStack::update (sf::Time *dt*)

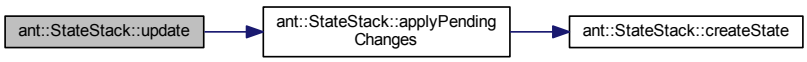
Actualiza los estados que tiene.

Parameters

dt	sf::Time delta time

Definition at line 9 of file StateStack.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

9.37.5 Member Data Documentation

9.37.5.1 AppState::Context ant::StateStack::mContext [private]

Definition at line 101 of file StateStack.hpp.

9.37.5.2 std::map<AppStates::ID, std::function<AppState::uPtr()> > ant::StateStack::mFactories [private]

Definition at line 102 of file StateStack.hpp.

9.37.5.3 std::vector<PendingChange> ant::StateStack::mPendingList [private]

Definition at line 100 of file StateStack.hpp.

9.37.5.4 std::vector<AppState::uPtr> ant::StateStack::mStack [private]

Definition at line 99 of file StateStack.hpp.

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

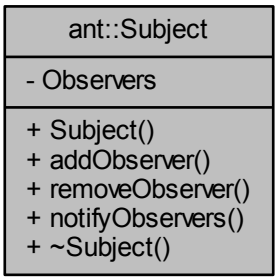
- StateStack.hpp
- StateStack.cpp

9.38 ant::Subject Class Reference

Tema al cual se suscriben los observadores Observer.

```
#include <Subject.hpp>
```

Collaboration diagram for ant::Subject:



Public Member Functions

- Subject ()

  Constructor por defecto.
- void addObserver (Observer *o)

  A un observador.
- void removeObserver (Observer *o)

  Elimina el primer observador que coincida.
- void notifyObservers (std::shared_ptr< baseEvent > e)

  Notifica a los Observadores que esten en suscritos a este tema.
- ~Subject ()

Private Attributes

- std::list< Observer * > Observers

9.38.1 Detailed Description

Tema al cual se suscriben los observadores Observer.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 12 of file Subject.hpp.

9.38.2 Constructor & Destructor Documentation

9.38.2.1 ant::Subject::Subject ( )

Constructor por defecto.

Definition at line 4 of file Subject.cpp.

9.38.2.2 ant::Subject::~Subject ( )

Definition at line 18 of file Subject.cpp.

9.38.3 Member Function Documentation

9.38.3.1 void ant::Subject::addObserver (Observer * *o*)

A un observador.

Parameters

o	Observer*

Definition at line 7 of file Subject.cpp.

9.38.3.2 void ant::Subject::notifyObservers (std::shared_ptr< baseEvent > *e*)

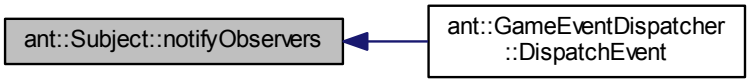
Notifica a los Observadores que esten en suscritos a este tema.

Parameters

e	std::shared_ptr<baseEvent> evento que se envia.

Definition at line 13 of file Subject.cpp.

Here is the caller graph for this function:



9.38.3.3 void ant::Subject::removeObserver (Observer * o)

Elimina el primer observador que coincida.

Parameters

o	Observer* observador a eliminar.

Definition at line 10 of file Subject.cpp.

9.38.4 Member Data Documentation

9.38.4.1 std::list<Observer*> ant::Subject::Observers [private]

Definition at line 35 of file Subject.hpp.

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

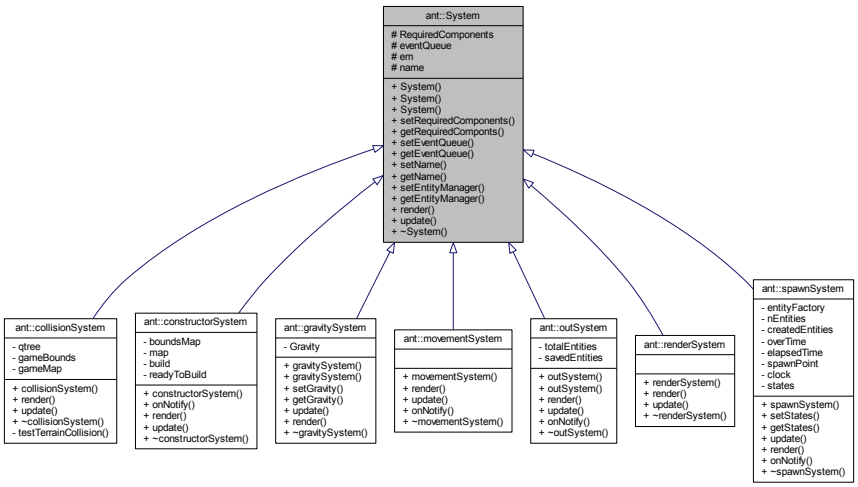
- Subject.hpp
- Subject.cpp

9.39 ant::System Class Reference

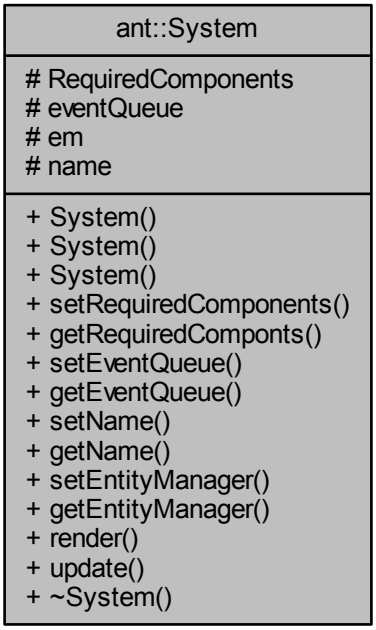
Clase base para los sistemas.

`#include <System.hpp>`

Inheritance diagram for ant::System:



Collaboration diagram for ant::System:



Public Member Functions

- System ()

  Constructor por defecto.
- System (long int req)

Constructor con los requisitos de este sistema.

- System (long int req, std::shared_ptr< EntityManager > em, std::shared_ptr< EventQueue > eventQueue)

  Constructor con los requisitos, EventQueue y EntityManager.
- void setRequiredComponents (long int req)

  Setter para los requisitos necesarios.
- long int getRequiredComponts ()

  Getter de los requisitos para el sistema.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la EventQueue.
- void setName (std::string name)

  Setter Para el nombre del sistema.
- std::string getName () const

  Getter para el nombre del sistema.
- void setEntityManager (std::shared_ptr< EntityManager > em)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.
- virtual void render (sf::RenderWindow &win)=0

  Metodo abstracto para dibujar en la pantalla.
- virtual void update (sf::Time dt)=0

  Metodo abstracto para actualizar entidades.
- virtual ~System ()

Protected Attributes

- long int RequiredComponents
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr< EntityManager > em
- std::string name

9.39.1 Detailed Description

Clase base para los sistemas.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file System.hpp.

9.39.2 Constructor & Destructor Documentation

9.39.2.1 ant::System::System ( )

Constructor por defecto.

Definition at line 3 of file System.cpp.

9.39.2.2 ant::System::System (long int *req*)

Constructor con los requisitos de este sistema.

Parameters

req	long int

Definition at line 7 of file System.cpp.

9.39.2.3 ant::System::System (long int *req*, std::shared_ptr< EntityManager > *em*, std::shared_ptr< EventQueue > *eventQueue*)

Constructor con los requisitos, EventQueue y EntityManager.

Parameters

req	long int
em	std::shared_ptr<EntityManager>
eventQueue	std::shared_ptr<EventQueue>

Definition at line 11 of file System.cpp.

9.39.2.4 ant::System::~System ( ) [virtual]

Definition at line 28 of file System.cpp.

9.39.3 Member Function Documentation

9.39.3.1 std::shared_ptr<EntityManager> ant::System::getEntityManager ( ) [inline]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 71 of file System.hpp.

9.39.3.2 std::shared_ptr<EventQueue> ant::System::getEventQueue ( ) [inline]

Getter para la EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 51 of file System.hpp.

9.39.3.3 std::string ant::System::getName ( ) const [inline]

Getter para el nombre del sistema.

Returns

std::string

Definition at line 61 of file System.hpp.

9.39.3.4 long int ant::System::getRequiredComponts ( ) [inline]

Getter de los requisitos para el sistema.

Definition at line 41 of file System.hpp.

9.39.3.5 virtual void ant::System::render (sf::RenderWindow & *win*) [pure virtual]

Metodo abstracto para dibujar en la pantalla.

Parameters

win	sf::RenderWindow & referencia a la pantalla para dibujar.

Implemented in ant::spawnSystem, ant::constructorSystem, ant::gravitySystem, ant::collisionSystem, ant::outSystem, ant::movementSystem, and ant::renderSystem.

9.39.3.6 void ant::System::setEntityManager (std::shared_ptr< **EntityManager** > *em*)

Setter para el EntityManager.

Parameters

em	std::shared_ptr<EntityManager>

Definition at line 25 of file System.cpp.

9.39.3.7 void ant::System::setEventQueue (std::shared_ptr< **EventQueue** > *eq*)

Setter para la EventQueue.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 19 of file System.cpp.

9.39.3.8 void ant::System::setName (std::string *name*)

Setter Para el nombre del sistema.

Parameters

name	std::string

Definition at line 22 of file System.cpp.

9.39.3.9 void ant::System::setRequiredComponents (long int *req*)

Setter para los requisitos necesarios.

Parameters

req	long int

Definition at line 16 of file System.cpp.

9.39.3.10 virtual void ant::System::update (sf::Time *dt*) [pure virtual]

Metodo abstracto para actualizar entidades.

Parameters

dt	sf::Time delta time para los calculos.

Implemented in ant::collisionSystem, ant::constructorSystem, ant::spawnSystem, ant::gravitySystem, ant::out↩System, ant::movementSystem, and ant::renderSystem.

9.39.4 Member Data Documentation

9.39.4.1 std::shared_ptr<EntityManager> ant::System::em [protected]

Definition at line 86 of file System.hpp.

9.39.4.2 std::shared_ptr<EventQueue> ant::System::eventQueue [protected]

Definition at line 85 of file System.hpp.

9.39.4.3 std::string ant::System::name [protected]

Definition at line 87 of file System.hpp.

9.39.4.4 long int ant::System::RequiredComponents [protected]

Definition at line 84 of file System.hpp.

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

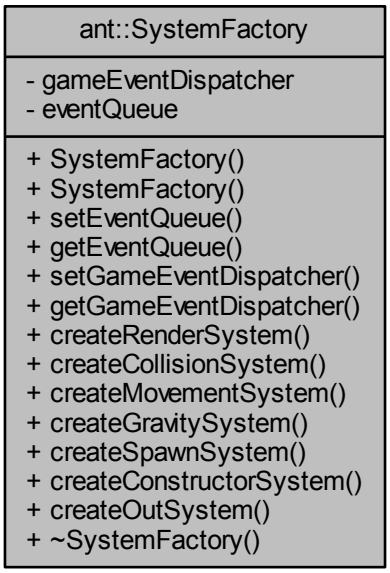
- System.hpp
- System.cpp

9.40 ant::SystemFactory Class Reference

Clase para crear sistemas.

```
#include <SystemFactory.hpp>
```

Collaboration diagram for ant::SystemFactory:



Public Member Functions

- SystemFactory ()

  Constructor por defecto.
- SystemFactory (std::shared_ptr< GameEventDispatcher > ged, std::shared_ptr< EventQueue > eq)

  Constructor para compartir la misma cola y suscritor de eventos.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Setter para la cola de eventos.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para la cola de eventos.
- void setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > ged)

  Setter para el suscritor de eventos.
- std::shared_ptr < GameEventDispatcher > getGameEventDispatcher ()

  Getter para el suscritor de eventos.
- std::shared_ptr< renderSystem > createRenderSystem ()

  Crea un sistema de renderizado.
- std::shared_ptr< collisionSystem > createCollisionSystem (sf::FloatRect bounds, sf::VertexArray *Game↩ Map)

  Crea un sistema de colision.
- std::shared_ptr< movementSystem > createMovementSystem ()

  Crea un sistema de movimiento.
- std::shared_ptr< gravitySystem > createGravitySystem (float gravity)

  Crea un sistema de gravedad.
- std::shared_ptr< spawnSystem > createSpawnSystem (int nEntities, EntityFactory *ef, sf::Time ot, sf::↩ Vector2f spawnPoint, long int state)

Crea un sistema de creacion de entidades.

- std::shared_ptr < constructorSystem > createConstructorSystem (sf::VertexArray *GameMap, sf::FloatRect bounds)

  Crea un sistema de construccion.

- std::shared_ptr< outSystem > createOutSystem (int totalEntities)

  Crea un sistema para comprobar cuando salen del mapa o entran por alguna entidad que tenga Component_OUT.

- ~SystemFactory ()

Private Attributes

- std::shared_ptr < GameEventDispatcher > gameEventDispatcher
- std::shared_ptr< EventQueue > eventQueue

9.40.1 Detailed Description

Clase para crear sistemas.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 20 of file SystemFactory.hpp.

9.40.2 Constructor & Destructor Documentation

9.40.2.1 ant::SystemFactory::SystemFactory ( )

Constructor por defecto.

Definition at line 4 of file SystemFactory.cpp.

9.40.2.2 ant::SystemFactory::SystemFactory (std::shared_ptr< GameEventDispatcher > *ged*, std::shared_ptr< EventQueue > *eq*)

Constructor para compartir la misma cola y suscritor de eventos.

Parameters

ged	std::shared_ptr<GameEventDispatcher>
eq	std::shared_ptr<EventQueue>

Definition at line 7 of file SystemFactory.cpp.

9.40.2.3 ant::SystemFactory::~SystemFactory ( )

Definition at line 58 of file SystemFactory.cpp.

9.40.3 Member Function Documentation

9.40.3.1 std::shared_ptr< collisionSystem > ant::SystemFactory::createCollisionSystem (sf::FloatRect *bounds*, sf::VertexArray * *GameMap*)

Crea un sistema de colision.

Parameters

GameMap	sf::VertexArray ∗ Terreno del juego.
bounds	sf::FloatRect rectangulo para zona que ocupa.

Returns

std::shared_ptr<collisionSystem>

Definition at line 21 of file SystemFactory.cpp.

9.40.3.2 std::shared_ptr< constructorSystem > ant::SystemFactory::createConstructorSystem (sf::VertexArray ∗ *GameMap*, sf::FloatRect *bounds*)

Crea un sistema de construccion.

See also

constructorSystem::constructorSystem

Returns

std::shared_ptr<constructorSystem>

Definition at line 43 of file SystemFactory.cpp.

9.40.3.3 std::shared_ptr< gravitySystem > ant::SystemFactory::createGravitySystem (float *gravity*)

Crea un sistema de gravedad.

See also

gravitySystem::gravitySystem

Parameters

gravity	float

Returns

std::shared_ptr<gravitySystem>

Definition at line 32 of file SystemFactory.cpp.

9.40.3.4 std::shared_ptr< movementSystem > ant::SystemFactory::createMovementSystem ( )

Crea un sistema de movimiento.

See also

movementSystem::movementSystem

Returns

std::shared_ptr<movementSystem>

Definition at line 26 of file SystemFactory.cpp.

9.40.3.5 std::shared_ptr< outSystem > ant::SystemFactory::createOutSystem (int *totalEntities*)

Crea un sistema para comprobar cuando salen del mapa o entran por alguna entidad que tenga Component_OUT.

See also

outSystem::outSystem

Returns

std::shared_ptr<outSystem>

Definition at line 51 of file SystemFactory.cpp.

9.40.3.6 std::shared_ptr< renderSystem > ant::SystemFactory::createRenderSystem ( )

Crea un sistema de renderizado.

Returns

std::shared_ptr<renderSystem>

Definition at line 16 of file SystemFactory.cpp.

9.40.3.7 std::shared_ptr< spawnSystem > ant::SystemFactory::createSpawnSystem (int *nEntities*, EntityFactory * *ef*, sf::Time *ot*, sf::Vector2f *spawnPoint*, long int *state*)

Crea un sistema de creacion de entidades.

See also

spawnSystem::spawnSystem

Returns

std::shared_ptr<spawnSystem>

Definition at line 37 of file SystemFactory.cpp.

9.40.3.8 std::shared_ptr<EventQueue> ant::SystemFactory::getEventQueue ( ) [inline]

Getter para la cola de eventos.

Returns

std::shared_ptr<EventQueue>

Definition at line 41 of file SystemFactory.hpp.

9.40.3.9 std::shared_ptr<GameEventDispatcher> ant::SystemFactory::getGameEventDispatcher ( ) [inline]

Getter para el suscritor de eventos.

Returns

std::shared_ptr<GameEventDispatcher>

Definition at line 51 of file SystemFactory.hpp.

9.40.3.10 void ant::SystemFactory::setEventQueue (std::shared_ptr< EventQueue > *eq*)

Setter para la cola de eventos.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 10 of file SystemFactory.cpp.

9.40.3.11 void ant::SystemFactory::setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > *ged*)

Setter para el suscritor de eventos.

Parameters

ged	std::shared_ptr<GameEventDispatcher>

Definition at line 13 of file SystemFactory.cpp.

9.40.4 Member Data Documentation

9.40.4.1 std::shared_ptr<EventQueue> ant::SystemFactory::eventQueue [private]

Definition at line 99 of file SystemFactory.hpp.

9.40.4.2 std::shared_ptr<GameEventDispatcher> ant::SystemFactory::gameEventDispatcher [private]

Definition at line 98 of file SystemFactory.hpp.

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

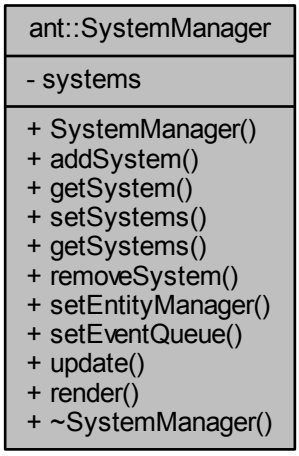
- SystemFactory.hpp
- SystemFactory.cpp

9.41 ant::SystemManager Class Reference

Clase contenedor para los systemas.

`#include <SystemManager.hpp>`

Collaboration diagram for ant::SystemManager:



Public Types

- typedef std::map< std::string, std::shared_ptr< System > > map

  Contenedor usado por la clase.

Public Member Functions

- SystemManager ()

  Constructor por defecto.
- bool addSystem (std::shared_ptr< System > s)

  A un sistema al manager.
- std::shared_ptr< System > getSystem (std::string name)

  Devuelve el sistema con el nombre especificado.
- void setSystems (map systems)

  Setter para poner el contenedor con todos los sistemas.
- SystemManager::map & getSystems ()

  Getter para obtener el contenedor de los sistemas.
- void removeSystem (std::string name)

  Elimina el sistema del manager.
- void setEntityManager (std::shared_ptr< EntityManager > entityManager)

  Establece el mismo EntityManager a todos los sistemas que esten agregados.
- void setEventQueue (std::shared_ptr< EventQueue > eventQueue)

  Establece la misma EventQueue a todos los sistemas que esten agregados.
- void update (sf::Time dt)

  Hace que todos los sistemas apliquen sus actualizaciones a sus EntityManager.
- void render (sf::RenderWindow &win)

  Hace que todos los sistemas renderizen a las Entidades que tengan que ser renderizadas.
- ~SystemManager ()

Private Attributes

- map systems

9.41.1 Detailed Description

Clase contenedor para los systemas.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 15 of file SystemManager.hpp.

9.41.2 Member Typedef Documentation

9.41.2.1 typedef std::map<std::string,std::shared_ptr<System> > ant::SystemManager::map

Contenedor usado por la clase.

Definition at line 20 of file SystemManager.hpp.

9.41.3 Constructor & Destructor Documentation

9.41.3.1 ant::SystemManager::SystemManager ( )

Constructor por defecto.

Definition at line 4 of file SystemManager.cpp.

9.41.3.2 ant::SystemManager::~SystemManager ( )

Definition at line 38 of file SystemManager.cpp.

9.41.4 Member Function Documentation

9.41.4.1 bool ant::SystemManager::addSystem (std::shared_ptr< System > *s*)

A un sistema al manager.

Parameters

s	std::shared_ptr<System>

Returns

bool

Definition at line 5 of file SystemManager.cpp.

9.41.4.2 std::shared_ptr< System > ant::SystemManager::getSystem (std::string *name*)

Devuelve el sistema con el nombre especificado.

Parameters

name	std::string nombre del sistema.

Returns

std::shared_ptr<System>

Definition at line 9 of file SystemManager.cpp.

9.41.4.3 SystemManager::map& ant::SystemManager::getSystems ( ) [inline]

Getter para obtener el contenedor de los sistemas.

Returns

SystemManager::map

Definition at line 46 of file SystemManager.hpp.

9.41.4.4 void ant::SystemManager::removeSystem (std::string *name*)

Elimina el sistema del manager.

Parameters

name	std::string nombre del sistema.

Definition at line 15 of file SystemManager.cpp.

9.41.4.5 void ant::SystemManager::render (sf::RenderWindow & *win*)

Hace que todos los sistemas renderizen a las Entidades que tengan que ser renderizadas.

Este metodo necesita ir entre sf::RenderWindow::clear() y sf::RenderWindow::display()

Parameters

win	sf::RenderWindow &

Definition at line 33 of file SystemManager.cpp.

9.41.4.6 void ant::SystemManager::setEntityManager (std::shared_ptr< EntityManager > *entityManager*)

Establece el mismo EntityManager a todos los sistemas que esten agregados.

Parameters

entityManager	std::shared_ptr<EntityManager>

Definition at line 18 of file SystemManager.cpp.

9.41.4.7 void ant::SystemManager::setEventQueue (std::shared_ptr< EventQueue > *eventQueue*)

Establece la misma EventQueue a todos los sistemas que esten agregados.

Parameters

eventQueue	std::shared_ptr<EventQueue>

Definition at line 23 of file SystemManager.cpp.

9.41.4.8 void ant::SystemManager::setSystems (map *systems*)

Setter para poner el contenedor con todos los sistemas.

Parameters

systems	map

Definition at line 12 of file SystemManager.cpp.

9.41.4.9 void ant::SystemManager::update (sf::Time *dt*)

Hace que todos los sistemas apliquen sus actualizaciones a sus EntityManager.

Parameters

dt	sf::Time delta time

Definition at line 28 of file SystemManager.cpp.

9.41.5 Member Data Documentation

9.41.5.1 map ant::SystemManager::systems [private]

Definition at line 77 of file SystemManager.hpp.

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

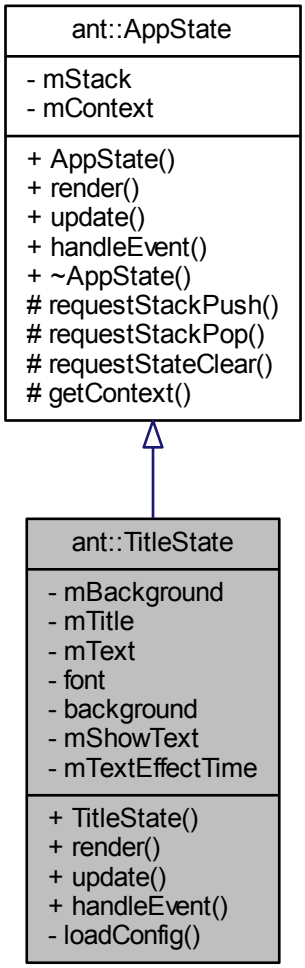
- SystemManager.hpp
- SystemManager.cpp

9.42 ant::TitleState Class Reference

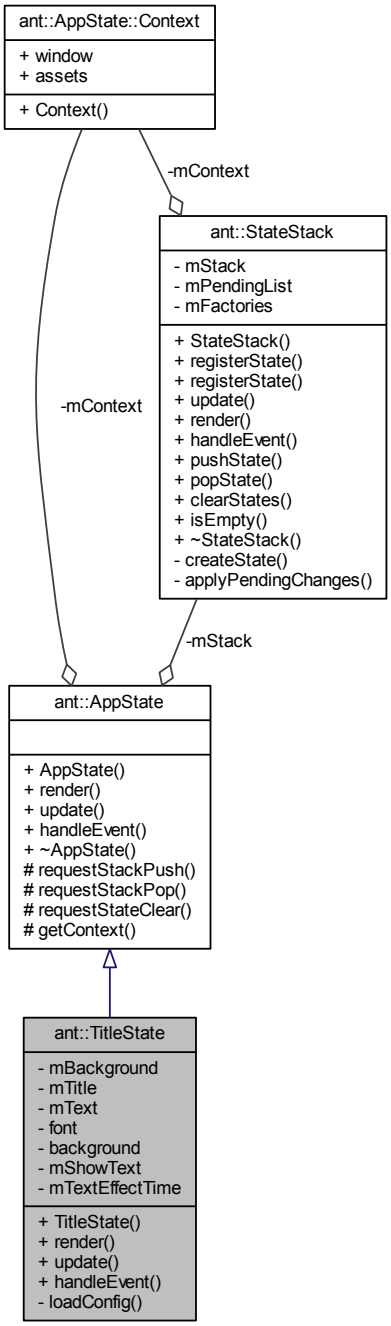
Muestra el titulo de la aplicacion.

```
#include <TitleState.hpp>
```

Inheritance diagram for ant::TitleState:



Collaboration diagram for ant::TitleState:



Public Types

- typedef std::unique_ptr< AppState > uPtr

  Puntero unico de AppState.

- typedef std::shared_ptr< AppState > sPtr

  Puntero Compartido de AppState.

Public Member Functions

- TitleState (StateStack &stack, Context context)

  Constructor por defecto.
- virtual void render ()

  Renderiza el estado en pantalla.
- virtual bool update (sf::Time dt)

  Actualiza.
- virtual bool handleEvent (const sf::Event &event)

  Maneja el input de teclado/raton.

Protected Member Functions

- void requestStackPush (AppStates::ID stateID)

  Pide al stack que a un estado nuevo.
- void requestStackPop ()

  Pide al stack que quite el ultimo estado ado.
- void requestStateClear ()

  Pide al stack que se limpie.
- Context getContext () const

  Devuelve el contexto.

Private Member Functions

- void loadConfig (const std::string &filename)

  Carga la configuracion de este estado.

Private Attributes

- sf::Sprite mBackground
- sf::Text mTitle
- sf::Text mText
- std::string font
- std::string background
- bool mShowText
- sf::Time mTextEffectTime

9.42.1 Detailed Description

Muestra el titulo de la aplicacion.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1a

Definition at line 14 of file TitleState.hpp.

9.42.2 Member Typedef Documentation

9.42.2.1 typedef std::shared_ptr<AppState> ant::AppState::sPtr [inherited]

Puntero Compartido de AppState.

Definition at line 26 of file AppState.hpp.

9.42.2.2 typedef std::unique_ptr<AppState> ant::AppState::uPtr [inherited]

Puntero unico de AppState.

Definition at line 22 of file AppState.hpp.

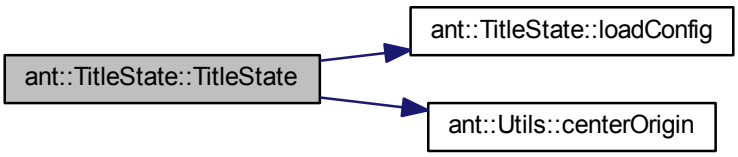
9.42.3 Constructor & Destructor Documentation

9.42.3.1 ant::TitleState::TitleState (StateStack & *stack*, Context *context*)

Constructor por defecto.

Definition at line 4 of file TitleState.cpp.

Here is the call graph for this function:



9.42.4 Member Function Documentation

9.42.4.1 AppState::Context ant::AppState::getContext ( ) const [protected], [inherited]

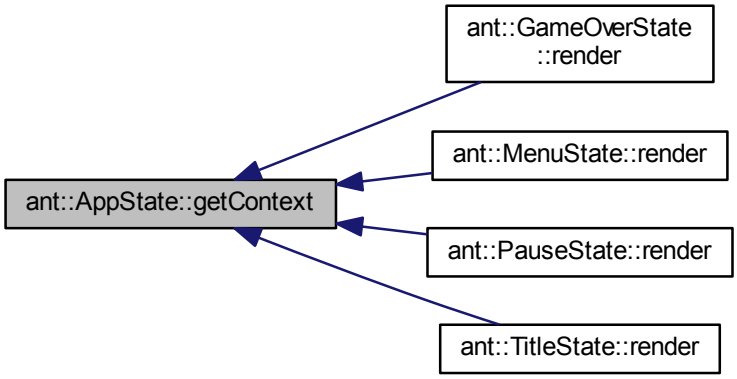
Devuelve el contexto.

Returns

AppState::Context

Definition at line 21 of file AppState.cpp.

Here is the caller graph for this function:



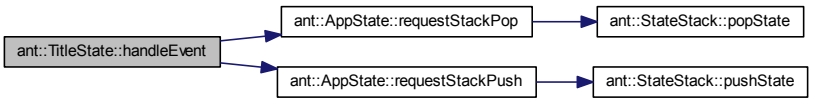
9.42.4.2 bool ant::TitleState::handleEvent (const sf::Event & *event*) [virtual]

Maneja el input de teclado/raton.

Implements ant::AppState.

Definition at line 48 of file TitleState.cpp.

Here is the call graph for this function:



9.42.4.3 void ant::TitleState::loadConfig (const std::string & *filename*) [private]

Carga la configuracion de este estado.

Definition at line 30 of file TitleState.cpp.

Here is the caller graph for this function:

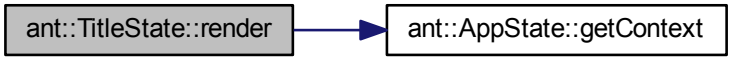
9.42.4.4 void ant::TitleState::render ( ) [virtual]

Renderiza el estado en pantalla.

Implements ant::AppState.

Definition at line 22 of file TitleState.cpp.

Here is the call graph for this function:

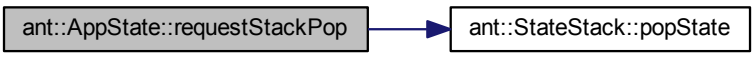


9.42.4.5 void ant::AppState::requestStackPop ( ) [protected],[inherited]

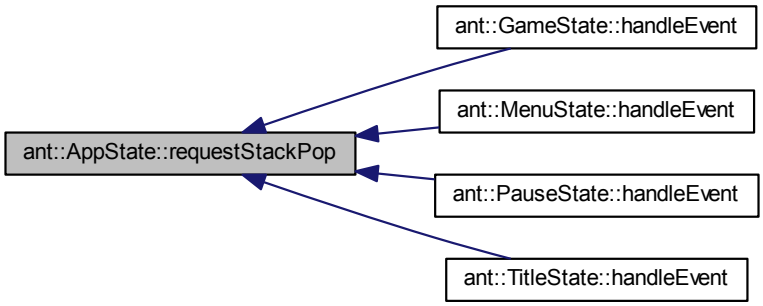
Pide al stack que quite el ultimo estado ado.

Definition at line 15 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.6 void ant::AppState::requestStackPush (AppStates::ID *stateID*) [protected],[inherited]

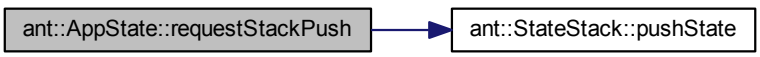
Pide al stack que a un estado nuevo.

Parameters

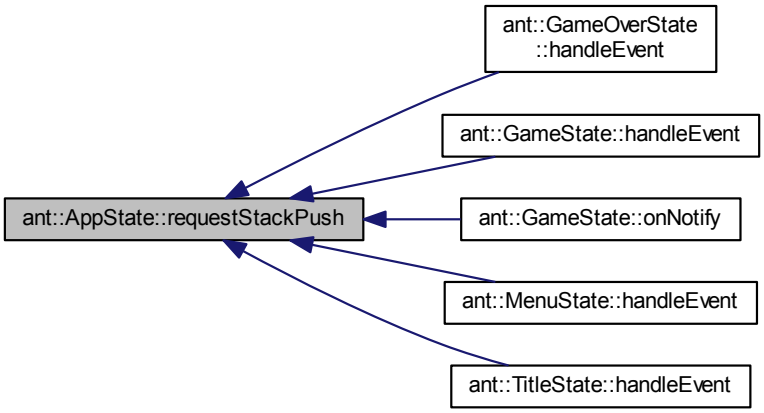
stateID	AppStates::ID id del estado.

Definition at line 12 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

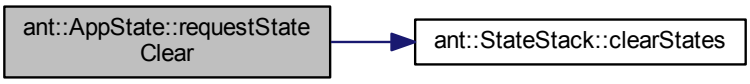


9.42.4.7 void ant::AppState::requestStateClear ( ) [protected],[inherited]

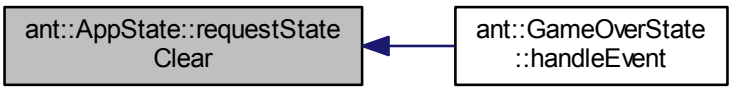
Pide al stack que se limpie.

Definition at line 18 of file AppState.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.8 bool ant::TitleState::update (sf::Time *dt*) [virtual]

Actualiza.

Implements ant::AppState.

Definition at line 40 of file TitleState.cpp.

9.42.5 Member Data Documentation

9.42.5.1 std::string ant::TitleState::background [private]

Definition at line 42 of file TitleState.hpp.

9.42.5.2 std::string ant::TitleState::font [private]

Definition at line 41 of file TitleState.hpp.

9.42.5.3 sf::Sprite ant::TitleState::mBackground [private]

Definition at line 38 of file TitleState.hpp.

9.42.5.4 bool ant::TitleState::mShowText [private]

Definition at line 43 of file TitleState.hpp.

9.42.5.5 sf::Text ant::TitleState::mText [private]

Definition at line 40 of file TitleState.hpp.

9.42.5.6 sf::Time ant::TitleState::mTextEffectTime [private]

Definition at line 44 of file TitleState.hpp.

9.42.5.7 sf::Text ant::TitleState::mTitle [private]

Definition at line 39 of file TitleState.hpp.

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

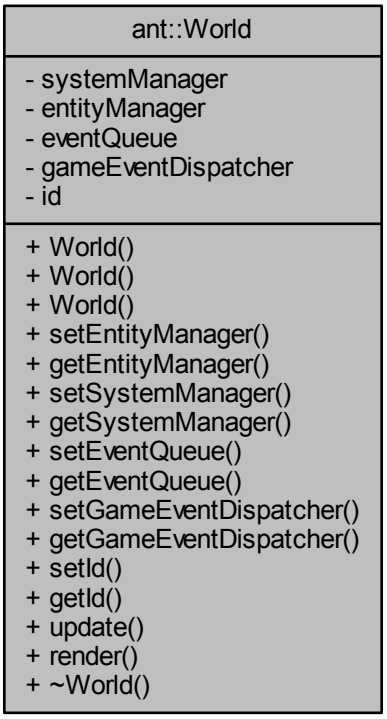
- TitleState.hpp
- TitleState.cpp

9.43 ant::World Class Reference

Clase para crear escenarios.

```
#include <World.hpp>
```

Collaboration diagram for ant::World:



Public Member Functions

- World ()

  Constructor por defecto.
- World (long int id)

  Constructor con el id del World.
- World (long int id, std::shared_ptr< EntityManager > entityManager, std::unique_ptr< SystemManager > systemManager, std::shared_ptr< EventQueue > eventQueue)

  Constructor con el id del World, EntityManager,SystemManager y EventQueue.
- void setEntityManager (std::shared_ptr< EntityManager > entityManager)

  Setter para el EntityManager.
- std::shared_ptr< EntityManager > getEntityManager ()

  Getter para el EntityManager.
- void setSystemManager (std::unique_ptr< SystemManager > systemManager)

  Setter para el SystemManager.
- SystemManager * getSystemManager ()

  Getter para el SystemManager.
- void setEventQueue (std::shared_ptr< EventQueue > eventQueue)

  Setter para la EventQueue.
- std::shared_ptr< EventQueue > getEventQueue ()

  Getter para el EventQueue.
- void setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > ged)

Setter para el GameEventDispatcher.

- std::shared_ptr < GameEventDispatcher > getGameEventDispatcher ()

  Getter para el GameEventDispatcher.
- void setId (long int id)

  Setter para el id del mundo.
- const long int getId () const

  Getter del id del mundo.
- void update (sf::Time dt)

  Actualiza todas las entidades del EntityManager con los sistemas que tenga en su SystemManager.
- void render (sf::RenderWindow &win)

  Renderiza todas las entidades del EntityManager que tengan los componentes necesarios.
- ~World ()

Private Attributes

- std::unique_ptr< SystemManager > systemManager
- std::shared_ptr< EntityManager > entityManager
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr < GameEventDispatcher > gameEventDispatcher
- long int id

9.43.1 Detailed Description

Clase para crear escenarios.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 15 of file World.hpp.

9.43.2 Constructor & Destructor Documentation

9.43.2.1 ant::World::World ( )

Constructor por defecto.

Definition at line 4 of file World.cpp.

9.43.2.2 ant::World::World (long int *id*)

Constructor con el id del World.

Parameters

id	long int id para este World.

Definition at line 9 of file World.cpp.

9.43.2.3 ant::World::World (long int *id*, std::shared_ptr< EntityManager > *entityManager*, std::unique_ptr< SystemManager > *systemManager*, std::shared_ptr< EventQueue > *eventQueue*)

Constructor con el id del World, EntityManager,SystemManager y EventQueue.

Parameters

id	long int id del mundo
entityManager	std::shared_ptr<EntityManager>
systemManager	std::shared_ptr<SystemManager>
eventQueue	std::shared_ptr<EventQueue>

Definition at line 15 of file World.cpp.

9.43.2.4 ant::World::~World ( )

Definition at line 44 of file World.cpp.

9.43.3 Member Function Documentation

9.43.3.1 std::shared_ptr<EntityManager> ant::World::getEntityManager ( ) [inline]

Getter para el EntityManager.

Returns

std::shared_ptr<EntityManager>

Definition at line 43 of file World.hpp.

9.43.3.2 std::shared_ptr<EventQueue> ant::World::getEventQueue ( ) [inline]

Getter para el EventQueue.

Returns

std::shared_ptr<EventQueue>

Definition at line 63 of file World.hpp.

9.43.3.3 std::shared_ptr<GameEventDispatcher> ant::World::getGameEventDispatcher ( ) [inline]

Getter para el GameEventDispatcher.

Returns

std::shared_ptr<GameEventDispatcher>

Definition at line 73 of file World.hpp.

9.43.3.4 const long int ant::World::getId ( ) const [inline]

Getter del id del mundo.

Returns

const long int

Definition at line 83 of file World.hpp.

9.43.3.5 SystemManager* ant::World::getSystemManager ( ) [inline]

Getter para el SystemManager.

Returns

SystemManager *

Definition at line 53 of file World.hpp.

9.43.3.6 void ant::World::render (sf::RenderWindow & *win*)

Renderiza todas las entidades del EntityManager que tengan los componentes necesarios.

Parameters

win	sf::RenderWindow &

Definition at line 41 of file World.cpp.

9.43.3.7 void ant::World::setEntityManager (std::shared_ptr< EntityManager > *entityManager*)

Setter para el EntityManager.

Parameters

entityManager	std::shared_ptr<EntityManager>

Definition at line 21 of file World.cpp.

9.43.3.8 void ant::World::setEventQueue (std::shared_ptr< EventQueue > *eventQueue*)

Setter para la EventQueue.

Parameters

eventQueue	std::shared_ptr<EventQueue>

Definition at line 29 of file World.cpp.

9.43.3.9 void ant::World::setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > *ged*)

Setter para el GameEventDispatcher.

Parameters

ged	std::shared_ptr<GameEventDispatcher>

Definition at line 32 of file World.cpp.

9.43.3.10 void ant::World::setId (long int *id*)

Setter para el id del mundo.

Parameters

id	long int

Definition at line 35 of file World.cpp.

9.43.3.11 void ant::World::setSystemManager (std::unique_ptr< SystemManager > *systemManager*)

Setter para el SystemManager.

Parameters

systemManager	std::unique_ptr<SystemManager>

Definition at line 25 of file World.cpp.

9.43.3.12 void ant::World::update (sf::Time *dt*)

Actualiza todas las entidades del EntityManager con los sistemas que tenga en su SystemManager.

Parameters

dt	sf::Time delta time

Definition at line 38 of file World.cpp.

9.43.4 Member Data Documentation

9.43.4.1 std::shared_ptr<EntityManager> ant::World::entityManager [private]

Definition at line 97 of file World.hpp.

9.43.4.2 std::shared_ptr<EventQueue> ant::World::eventQueue [private]

Definition at line 98 of file World.hpp.

9.43.4.3 std::shared_ptr<GameEventDispatcher> ant::World::gameEventDispatcher [private]

Definition at line 99 of file World.hpp.

9.43.4.4 long int ant::World::id [private]

Definition at line 100 of file World.hpp.

9.43.4.5 std::unique_ptr<SystemManager> ant::World::systemManager [private]

Definition at line 96 of file World.hpp.

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

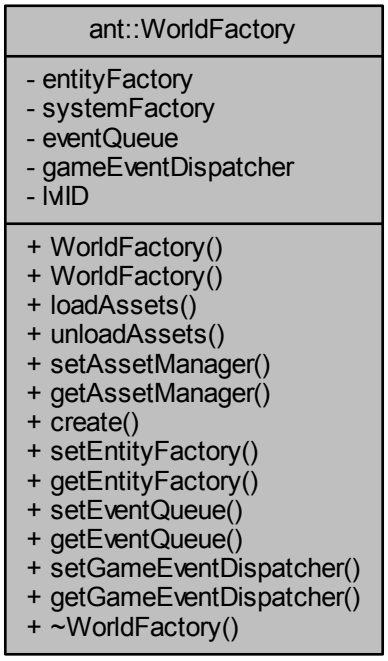
- World.hpp
- World.cpp

9.44 ant::WorldFactory Class Reference

Clase para crear mundos.

`#include <WorldFactory.hpp>`

Collaboration diagram for ant::WorldFactory:



Public Member Functions

- WorldFactory ()

  Constructor por defecto.
- WorldFactory (std::shared_ptr< GameEventDispatcher > ged, std::shared_ptr< EventQueue > eq)

  Constructor con GameEventDispatcher y EventQueue.
- bool loadAssets (const std::string &json)

  Carga los assets necesarios para crear los mundos.
- bool unloadAssets ()

  Descarga los assets que tenga.

- void setAssetManager (std::shared_ptr< AssetManager > assets)

  Establece el AssetManager que usara.
- std::shared_ptr< AssetManager > getAssetManager ()

  Obtiene el AssetManager que usa.
- std::unique_ptr< World > create (const std::string &name, const std::string &background, const sf::Image &lvl, int nEntities, sf::Time overTime, sf::FloatRect bounds)

  Crea un nivel segun la imagen especificada.
- void setEntityFactory (std::shared_ptr< EntityFactory > ef)

  Establece el EntityFactory que usara.
- std::shared_ptr< EntityFactory > getEntityFactory ()

  Obtiene el EntityFactory que usa.
- void setEventQueue (std::shared_ptr< EventQueue > eq)

  Establece la EventQueue que usaran los sistemas.
- std::shared_ptr< EventQueue > getEventQueue ()

  Obtiene la EventQueue que usa.
- void setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > ged)

  Establece el GameEventDispatcher que usara.
- std::shared_ptr
  < GameEventDispatcher > getGameEventDispatcher ()

  Obtiene el GameEventDispatcher que usa.
- ~WorldFactory ()

Private Attributes

- std::shared_ptr< EntityFactory > entityFactory
- std::shared_ptr< SystemFactory > systemFactory
- std::shared_ptr< EventQueue > eventQueue
- std::shared_ptr
  < GameEventDispatcher > gameEventDispatcher
- int lvlID

9.44.1 Detailed Description

Clase para crear mundos.

Utilizara el GameEventDispatcher, EventQueue que tenga para los mundos generados.

Author

Cristian Glez `Cristian.glez.m@gmail.com`

Version

0.1

Definition at line 18 of file WorldFactory.hpp.

9.44.2 Constructor & Destructor Documentation

9.44.2.1 ant::WorldFactory::WorldFactory ( )

Constructor por defecto.

Definition at line 4 of file WorldFactory.cpp.

9.44.2.2 ant::WorldFactory::WorldFactory (std::shared_ptr< GameEventDispatcher > *ged*, std::shared_ptr< EventQueue > *eq*)

Constructor con GameEventDispatcher y EventQueue.

Parameters

ged	std::shared_ptr<GameEventDispatcher>
eq	std::shared_ptr<EventQueue>

Definition at line 11 of file WorldFactory.cpp.

9.44.2.3 ant::WorldFactory::~WorldFactory ( )

Definition at line 110 of file WorldFactory.cpp.

9.44.3 Member Function Documentation

9.44.3.1 std::unique_ptr< World > ant::WorldFactory::create (const std::string & *name*, const std::string & *background*, const sf::Image & *lvl*, int *nEntities*, sf::Time *overTime*, sf::FloatRect *bounds*)

Crea un nivel segun la imagen especificada.

La imagen especificada tiene que tener:

Colores

1. Color blanco es ignorado.
2. Color negro es suelo.
3. Color rojo es la puerta de entrada al nivel.
4. Color verde es la puerta de salida del nivel.

Parameters

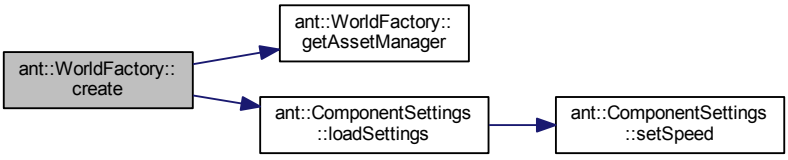
name	std::string nombre del nivel.
background	std::string nombre de la textura del fondo.
lvl	sf::Image imagen a cargar con la informacion del nivel.
nEntities	int numero de entidades que tendra el nivel.
overTime	sf::Time tiempo maximo para pasarse el nivel.
bounds	sf::FloatRect rectangulo con los limites del mundo(Tienen que ser iguales a la imagenes de fondo.)

Returns

std::unique_ptr<World> World creado

Definition at line 27 of file WorldFactory.cpp.

Here is the call graph for this function:



9.44.3.2 std::shared_ptr<AssetManager> ant::WorldFactory::getAssetManager ( ) [inline]

Obtiene el AssetManager que usa.

Returns

std::shared_ptr<AssetManager>

Definition at line 50 of file WorldFactory.hpp.

Here is the caller graph for this function:

9.44.3.3 std::shared_ptr<EntityFactory> ant::WorldFactory::getEntityFactory ( ) [inline]

Obtiene el EntityFactory que usa.

Returns

std::shared_ptr<EntityFactory>

Definition at line 80 of file WorldFactory.hpp.

9.44.3.4 std::shared_ptr<EventQueue> ant::WorldFactory::getEventQueue ( ) [inline]

Obtiene la EventQueue que usa.

Returns

std::shared_ptr<EventQueue>

Definition at line 90 of file WorldFactory.hpp.

9.44.3.5 std::shared_ptr<GameEventDispatcher> ant::WorldFactory::getGameEventDispatcher ( ) [inline]

Obtiene el GameEventDispatcher que usa.

Returns

std::shared_ptr<GameEventDispatcher>

Definition at line 100 of file WorldFactory.hpp.

9.44.3.6 bool ant::WorldFactory::loadAssets (const std::string & *json*)

Carga los assets necesarios para crear los mundos.

Parameters

json	const std::string & fichero json con los datos de configuracion.

Returns

bool

Definition at line 18 of file WorldFactory.cpp.

9.44.3.7 void ant::WorldFactory::setAssetManager (std::shared_ptr< AssetManager > *assets*)

Establece el AssetManager que usara.

Parameters

assets	std::shared_ptr<AssetManager>

Definition at line 24 of file WorldFactory.cpp.

9.44.3.8 void ant::WorldFactory::setEntityFactory (std::shared_ptr< EntityFactory > *ef*)

Establece el EntityFactory que usara.

Parameters

ef	std::shared_ptr<EntityFactory>

Definition at line 101 of file WorldFactory.cpp.

9.44.3.9 void ant::WorldFactory::setEventQueue (std::shared_ptr< EventQueue > *eq*)

Establece la EventQueue que usaran los sistemas.

Parameters

eq	std::shared_ptr<EventQueue>

Definition at line 104 of file WorldFactory.cpp.

9.44.3.10 void ant::WorldFactory::setGameEventDispatcher (std::shared_ptr< GameEventDispatcher > *ged*)

Establece el GameEventDispatcher que usara.

Parameters

ged	std::shared_ptr<GameEventDispatcher>

Definition at line 107 of file WorldFactory.cpp.

9.44.3.11 bool ant::WorldFactory::unloadAssets ( )

Descarga los assets que tenga.

Returns

bool

Definition at line 21 of file WorldFactory.cpp.

9.44.4 Member Data Documentation

9.44.4.1 std::shared_ptr<EntityFactory> ant::WorldFactory::entityFactory [private]

Definition at line 103 of file WorldFactory.hpp.

9.44.4.2 std::shared_ptr<EventQueue> ant::WorldFactory::eventQueue [private]

Definition at line 105 of file WorldFactory.hpp.

9.44.4.3 std::shared_ptr<GameEventDispatcher> ant::WorldFactory::gameEventDispatcher [private]

Definition at line 106 of file WorldFactory.hpp.

9.44.4.4 int ant::WorldFactory::lvlID [private]

Definition at line 107 of file WorldFactory.hpp.

9.44.4.5 std::shared_ptr<SystemFactory> ant::WorldFactory::systemFactory [private]

Definition at line 104 of file WorldFactory.hpp.

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

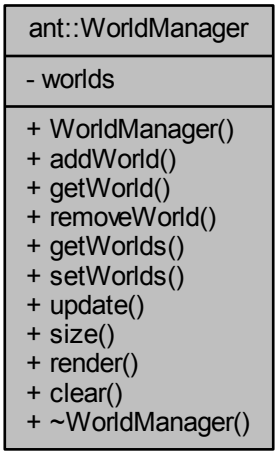
- WorldFactory.hpp
- WorldFactory.cpp

9.45 ant::WorldManager Class Reference

Clase Contenedor para los mundos.

```
#include <WorldManager.hpp>
```

Collaboration diagram for ant::WorldManager:



Public Types

- typedef std::map< long int, std::unique_ptr< World > > map

  Contenedor usado por WorldManager.
- typedef map::iterator iterator

  Iterador del Contenedor.

Public Member Functions

- WorldManager ()

  Constructor por defecto.
- void addWorld (std::unique_ptr< World > w)

  A un World al contenedor.
- World * getWorld (long int id)

  Obtiene un "naked Pointer" del mundo especificado. Si no se encuentra devuelve un nullptr.
- void removeWorld (long int id)

  Elimina el mundo especificado.
- map & getWorlds ()

  Devuelve el contendor usado.
- void setWorlds (map worlds)

  Establece el contenedor.
- void update (long int id, sf::Time dt)

  Actualiza el mundo especificado por id.
- int size ()

  Devuelve el numero de mundos que tiene.
- void render (long int id, sf::RenderWindow &win)

  Renderiza el mundo especificado por el id.
- void clear ()

Limpia los mundos que contiene.

- ~WorldManager ()

Private Attributes

- map worlds

9.45.1 Detailed Description

Clase Contenedor para los mundos.

Author

Cristian Glez Cristian.glez.m@gmail.com

Version

0.1

Definition at line 14 of file WorldManager.hpp.

9.45.2 Member Typedef Documentation

9.45.2.1 typedef map::iterator ant::WorldManager::iterator

Iterador del Contenedor.

Definition at line 23 of file WorldManager.hpp.

9.45.2.2 typedef std::map<long int,std::unique_ptr<World> > ant::WorldManager::map

Contenedor usado por WorldManager.

Definition at line 19 of file WorldManager.hpp.

9.45.3 Constructor & Destructor Documentation

9.45.3.1 ant::WorldManager::WorldManager ( )

Constructor por defecto.

Definition at line 4 of file WorldManager.cpp.

9.45.3.2 ant::WorldManager::~WorldManager ( )

Definition at line 40 of file WorldManager.cpp.

9.45.4 Member Function Documentation

9.45.4.1 void ant::WorldManager::addWorld (std::unique_ptr< World > *w*)

A un World al contenedor.

Parameters

w	std::unique_ptr<World>

Definition at line 7 of file WorldManager.cpp.

9.45.4.2 void ant::WorldManager::clear ( )

Limpia los mundos que contiene.

Definition at line 37 of file WorldManager.cpp.

9.45.4.3 World * ant::WorldManager::getWorld (long int *id*)

Obtiene un "naked Pointer" del mundo especificado. Si no se encuentra devuelve un nullptr.

Parameters

id	long int id del mundo

Returns

World *

Definition at line 10 of file WorldManager.cpp.

9.45.4.4 map& ant::WorldManager::getWorlds ( ) [inline]

Devuelve el contendor usado.

Returns

map

Definition at line 49 of file WorldManager.hpp.

9.45.4.5 void ant::WorldManager::removeWorld (long int *id*)

Elimina el mundo especificado.

Parameters

id	long int

Definition at line 13 of file WorldManager.cpp.

9.45.4.6 void ant::WorldManager::render (long int *id*, sf::RenderWindow & *win*)

Renderiza el mundo especificado por el id.

Si se pone -1 Renderizara todos los mundos, en pantalla se vera el ultimo.

Parameters

id	long int id del mundo (-1 para renderizar a todos)

win	sf::RenderWindow & ventana a la cual dibujar.

Definition at line 28 of file WorldManager.cpp.

9.45.4.7 void ant::WorldManager::setWorlds (map *worlds*)

Establece el contenedor.

Parameters

worlds	map

Definition at line 16 of file WorldManager.cpp.

9.45.4.8 int ant::WorldManager::size ( ) [inline]

Devuelve el numero de mundos que tiene.

Definition at line 67 of file WorldManager.hpp.

9.45.4.9 void ant::WorldManager::update (long int *id*, sf::Time *dt*)

Actualiza el mundo especificado por id.

Si se pone -1 actualiza a todos los mundos que contenga.

Parameters

id	long int id del mundo (-1 para actualizar todos)
dt	sf::Time delta time

Definition at line 19 of file WorldManager.cpp.

9.45.5 Member Data Documentation

9.45.5.1 map ant::WorldManager::worlds [private]

Definition at line 83 of file WorldManager.hpp.

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

- WorldManager.hpp
- WorldManager.cpp

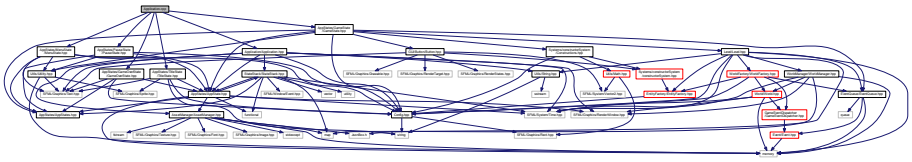
Chapter 10

File Documentation

10.1 Application.cpp File Reference

```
#include <Application/Application.hpp>
#include <AppStates/TitleState/TitleState.hpp>
#include <AppStates/MenuState/MenuState.hpp>
#include <AppStates/GameState/GameState.hpp>
#include <AppStates/PauseState/PauseState.hpp>
#include <AppStates/GameOverState/GameOverState.hpp>
```

Include dependency graph for Application.cpp:



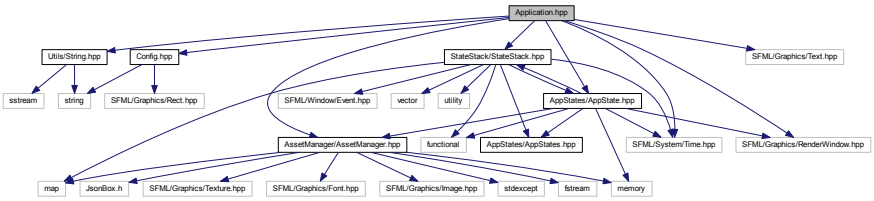
Namespaces

- ant

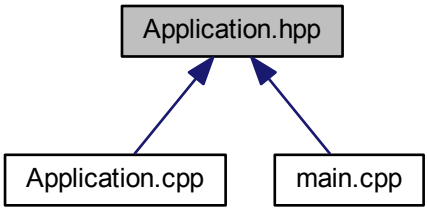
10.2 Application.hpp File Reference

```
#include <Config.hpp>
#include <Utils/String.hpp>
#include <AssetManager/AssetManager.hpp>
#include <StateStack/StateStack.hpp>
#include <AppStates/AppState.hpp>
#include <SFML/System/Time.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for Application.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Application

  Clase de aplicacion.

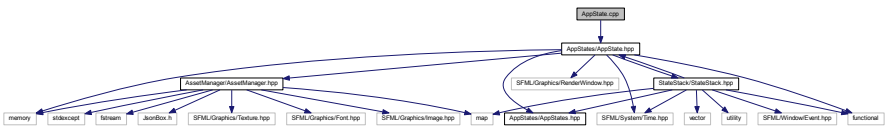
Namespaces

- ant

10.3 AppState.cpp File Reference

`#include <AppStates/AppState.hpp>`

Include dependency graph for AppState.cpp:



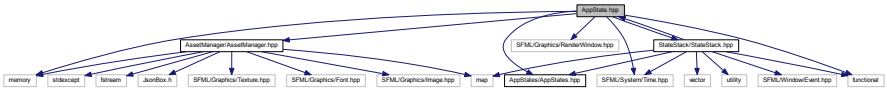
Namespaces

- ant

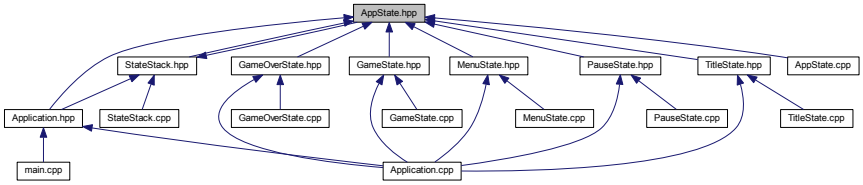
10.4 AppState.hpp File Reference

```
#include <AssetManager/AssetManager.hpp>
#include <AppStates/AppStates.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
#include <SFML/System/Time.hpp>
#include <memory>
#include <functional>
#include <StateStack/StateStack.hpp>
```

Include dependency graph for AppState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::AppState

  Clase abstracta de estado de aplicacion.
- struct ant::AppState::Context

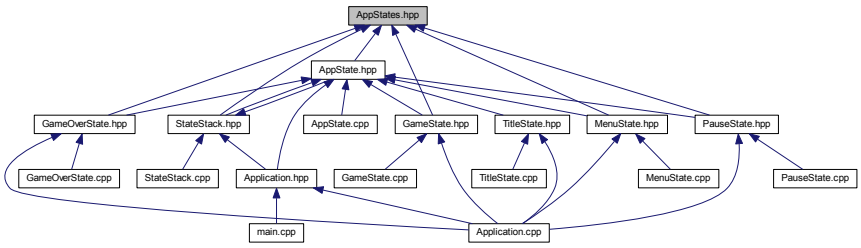
  Clase para compartir datos esenciales a los estados.

Namespaces

- ant

10.5 AppStates.hpp File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

- ant
- ant::AppStates

  Tiene los ID de los estados de la aplicacion.

Enumerations

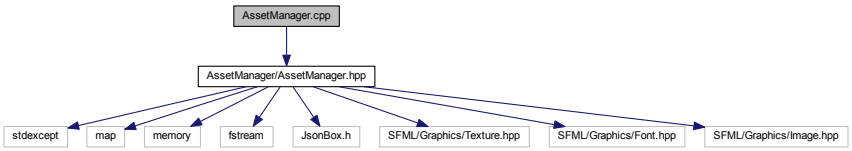
- enum ant::AppStates::ID {
  ant::AppStates::None, ant::AppStates::Title, ant::AppStates::Menu, ant::AppStates::Game, ant::AppStates::Pause, ant::AppStates::GameFailed, ant::AppStates::GameSucceed }

  ID de los estados de la aplicacion.

10.6 AssetManager.cpp File Reference

```
#include <AssetManager/AssetManager.hpp>
```

Include dependency graph for AssetManager.cpp:



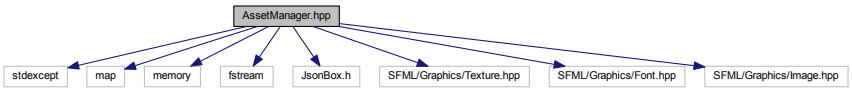
Namespaces

- ant

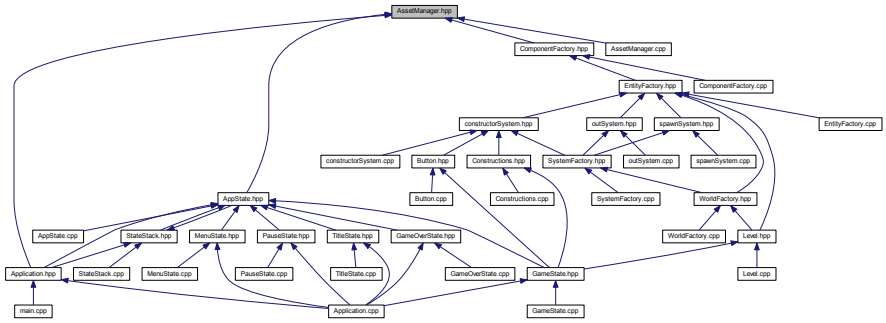
10.7 AssetManager.hpp File Reference

```
#include <stdexcept>
#include <map>
#include <memory>
#include <fstream>
#include <JsonBox.h>
#include <SFML/Graphics/Texture.hpp>
#include <SFML/Graphics/Font.hpp>
#include <SFML/Graphics/Image.hpp>
```

Include dependency graph for AssetManager.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::AssetManager

  Clase para tener todos los assets(imagenes,texturas,etc) del juego.

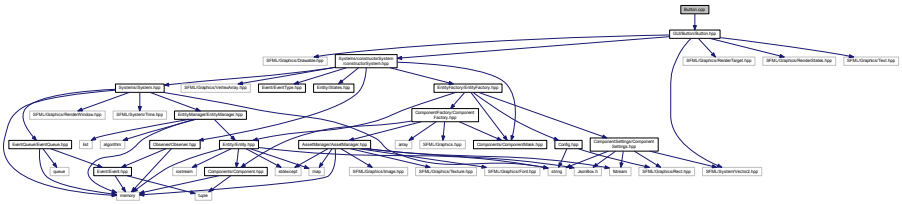
Namespaces

- ant

10.8 Button.cpp File Reference

```
#include <GUI/Button/Button.hpp>
```

Include dependency graph for Button.cpp:



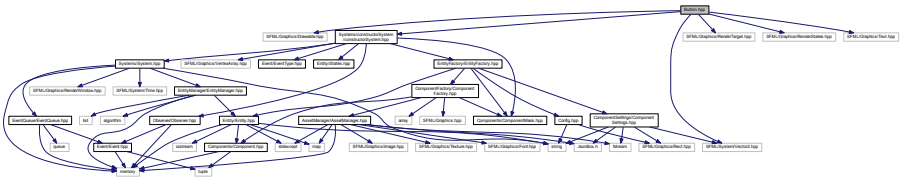
Namespaces

- ant
- ant::GUI

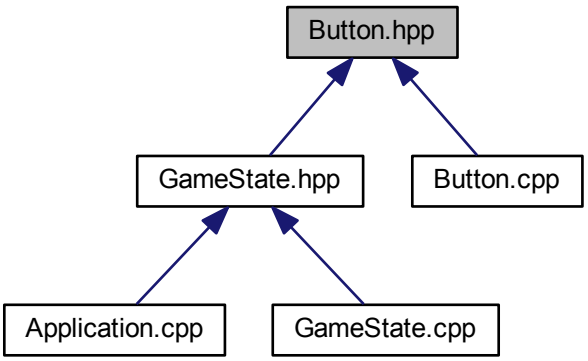
10.9 Button.hpp File Reference

```
#include <SFML/Graphics/Drawable.hpp>
#include <Systems/constructorSystem/constructorSystem.hpp>
#include <SFML/System/Vector2.hpp>
#include <SFML/Graphics/RenderTarget.hpp>
#include <SFML/Graphics/RenderStates.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for Button.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::GUI::Button

  Clase de un simple boton.

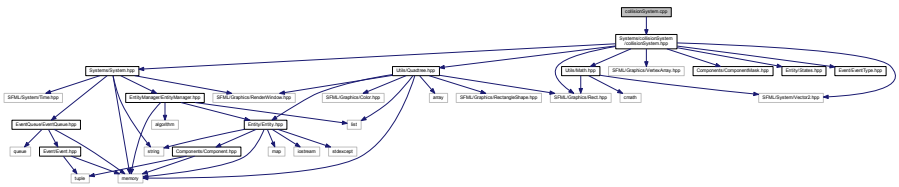
Namespaces

- ant
- ant::GUI

10.10 collisionSystem.cpp File Reference

```
#include <Systems/collisionSystem/collisionSystem.hpp>
```

Include dependency graph for collisionSystem.cpp:



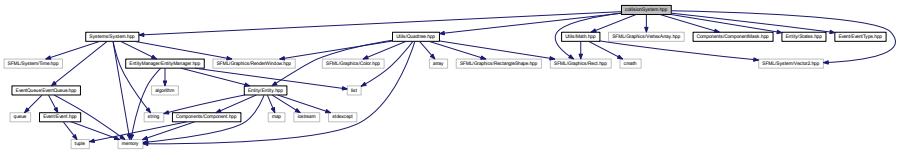
Namespaces

- ant

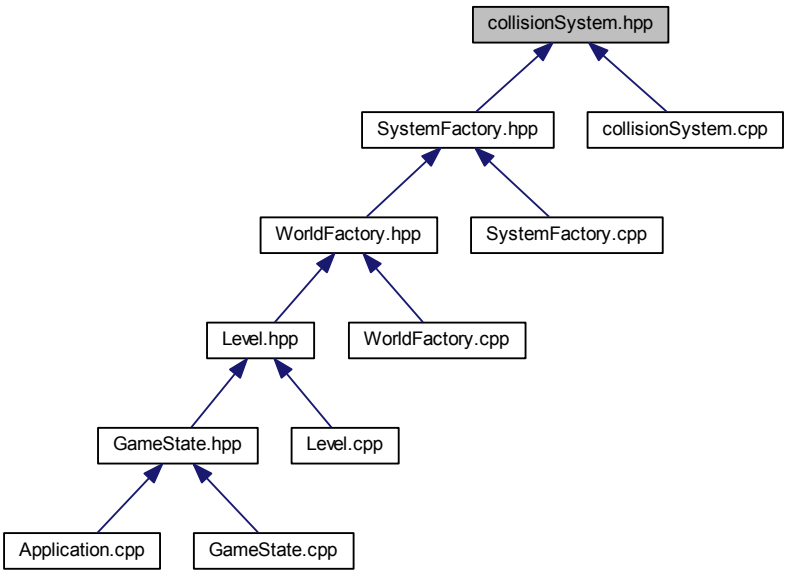
10.11 collisionSystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <Utils/Quadtree.hpp>
#include <SFML/System/Vector2.hpp>
#include <SFML/Graphics/Rect.hpp>
#include <SFML/Graphics/VertexArray.hpp>
#include <Components/ComponentMask.hpp>
#include <Entity/States.hpp>
#include <Event/EventType.hpp>
#include <Utils/Math.hpp>
```

Include dependency graph for collisionSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::collisionSystem

  Sistema que comprueba las colisiones entre entidades.

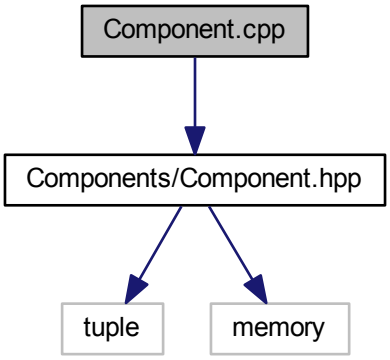
Namespaces

- ant

10.12 Component.cpp File Reference

```
#include <Components/Component.hpp>
```

Include dependency graph for Component.cpp:



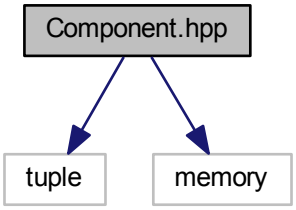
Namespaces

- ant

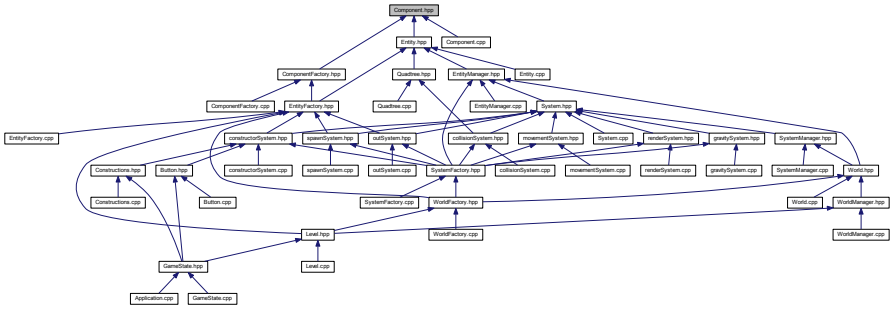
10.13 Component.hpp File Reference

```
#include <tuple>
#include <memory>
```

Include dependency graph for Component.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::baseComponent

  Clase base de componente.
- class ant::Component< T >

  Clase especializada de baseComponent.

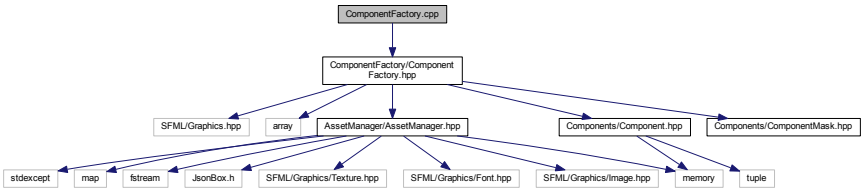
Namespaces

- ant

10.14 ComponentFactory.cpp File Reference

```
#include <ComponentFactory/ComponentFactory.hpp>
```

Include dependency graph for ComponentFactory.cpp:



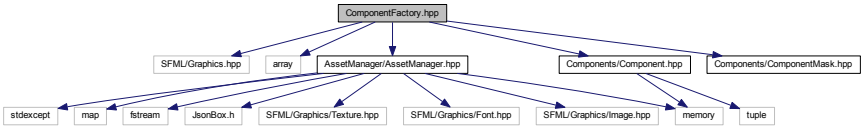
Namespaces

- ant

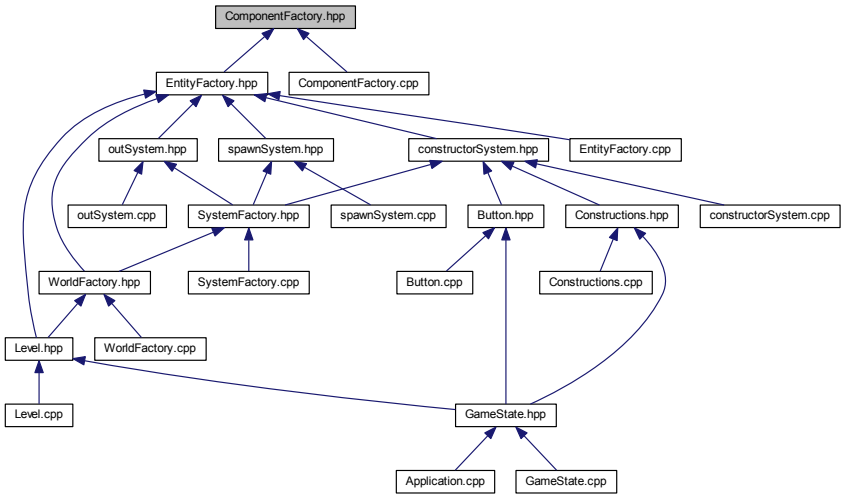
10.15 ComponentFactory.hpp File Reference

```
#include <SFML/Graphics.hpp>
#include <array>
#include <AssetManager/AssetManager.hpp>
#include <Components/Component.hpp>
#include <Components/ComponentMask.hpp>
```

Include dependency graph for ComponentFactory.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::ComponentFactory

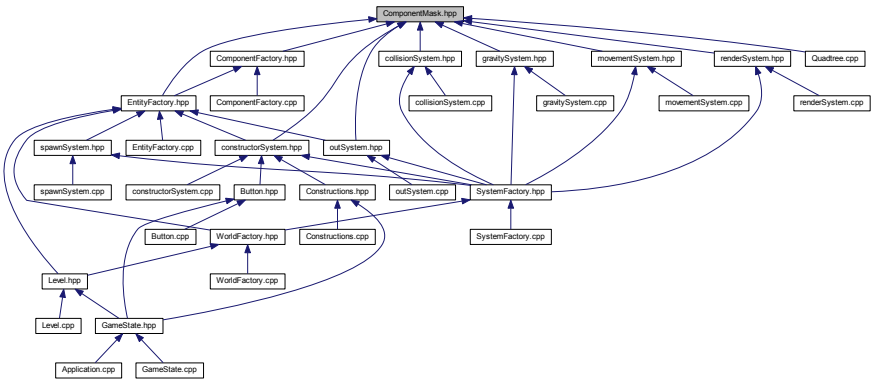
  Clase para crear los componentes de las entidades.

Namespaces

- ant

10.16 ComponentMask.hpp File Reference

This graph shows which files directly or indirectly include this file:



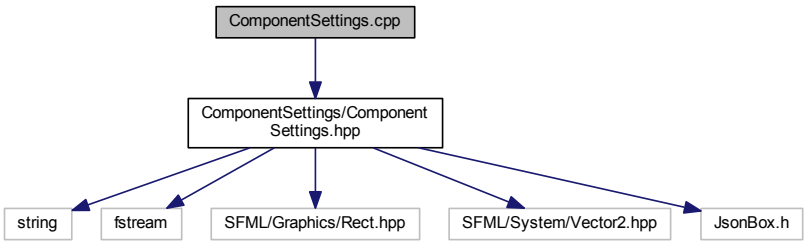
Classes

- class ComponentsMask

  tipos de componentes.

10.17 ComponentSettings.cpp File Reference

`#include <ComponentSettings/ComponentSettings.hpp>`
Include dependency graph for ComponentSettings.cpp:



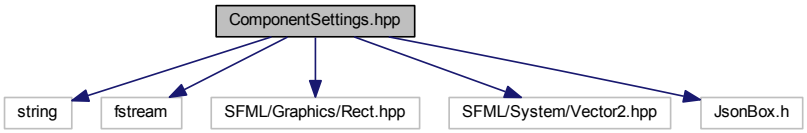
Namespaces

- ant

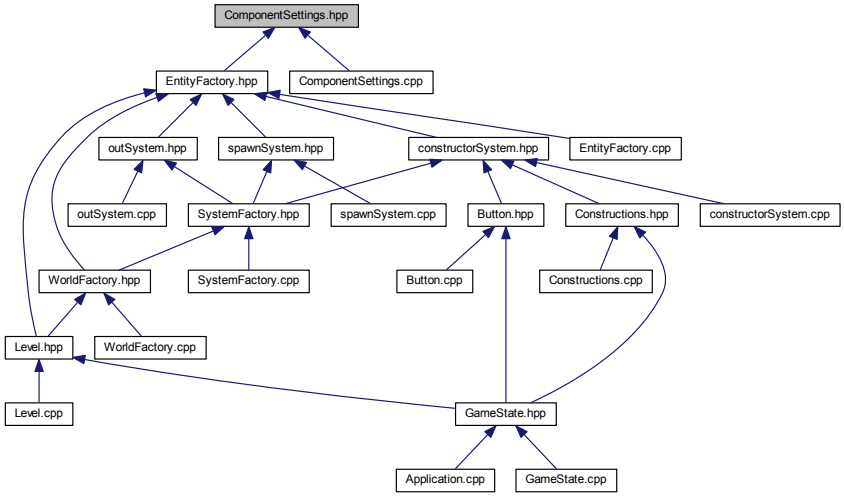
10.18 ComponentSettings.hpp File Reference

```
#include <string>
#include <fstream>
#include <SFML/Graphics/Rect.hpp>
#include <SFML/System/Vector2.hpp>
#include <JsonBox.h>
```

Include dependency graph for ComponentSettings.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::ComponentSettings

  Clase para la configuracion de los componentes en EntityFactory.

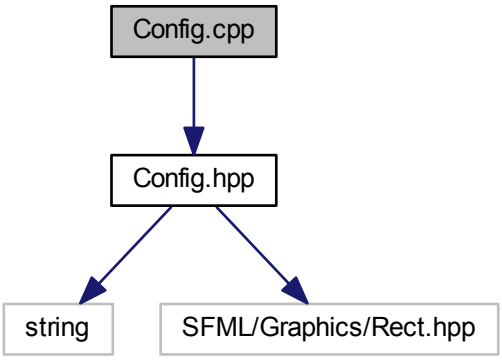
Namespaces

- ant

10.19 Config.cpp File Reference

```
#include <Config.hpp>
```

Include dependency graph for Config.cpp:



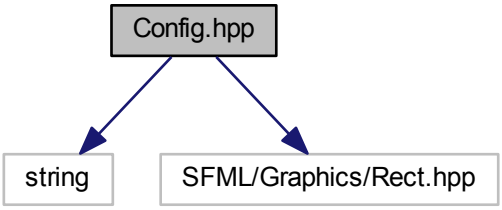
Namespaces

- ant

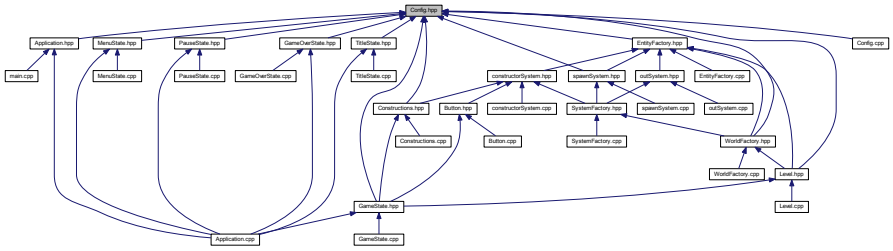
10.20 Config.hpp File Reference

```
#include <string>
#include <SFML/Graphics/Rect.hpp>
```

Include dependency graph for Config.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Config

  Clase de configuracion con constantes de ficheros.

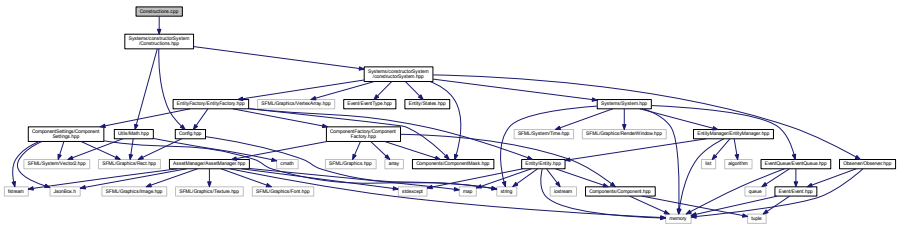
Namespaces

- ant

10.21 Constructions.cpp File Reference

```
#include <Systems/constructorSystem/Constructions.hpp>
```

Include dependency graph for Constructions.cpp:



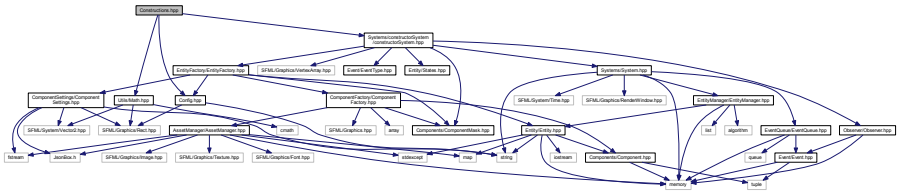
Namespaces

- ant

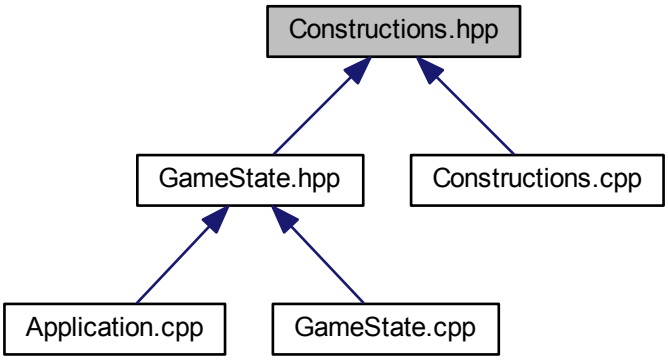
10.22 Constructions.hpp File Reference

```
#include <Systems/constructorSystem/constructorSystem.hpp>
#include <Config.hpp>
#include <Utils/Math.hpp>
```

Include dependency graph for Constructions.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Constructions

  Clase estatica con los comandos para las hormigas.

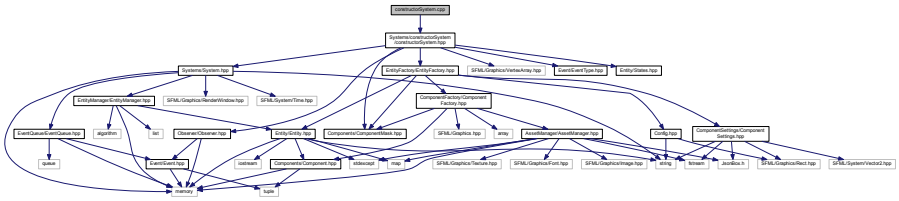
Namespaces

- ant

10.23 constructorSystem.cpp File Reference

`#include <Systems/constructorSystem/constructorSystem.hpp>`

Include dependency graph for constructorSystem.cpp:



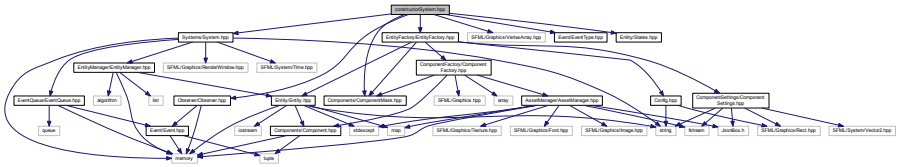
Namespaces

- ant

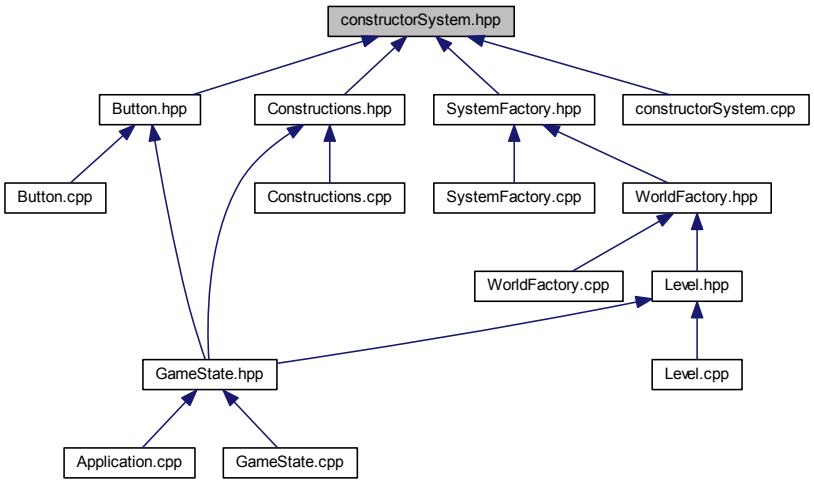
10.24 constructorSystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <Observer/Observer.hpp>
#include <Components/ComponentMask.hpp>
#include <EntityFactory/EntityFactory.hpp>
#include <SFML/Graphics/VertexArray.hpp>
#include <Event/EventType.hpp>
#include <Entity/States.hpp>
```

Include dependency graph for constructorSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::constructorSystem

  Sistema encargado de que las hormigas construyan etc.

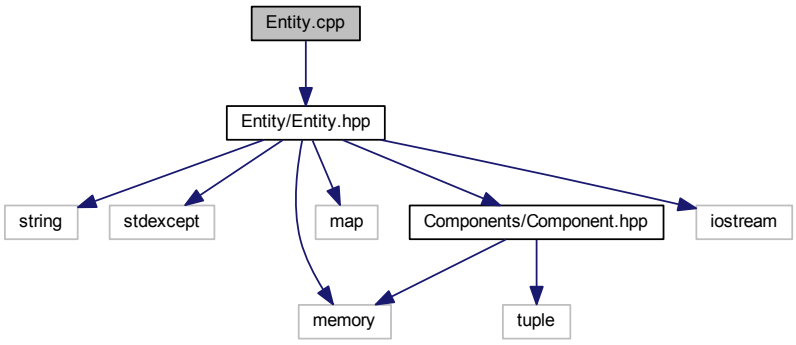
Namespaces

- ant

10.25 Entity.cpp File Reference

```
#include <Entity/Entity.hpp>
```

Include dependency graph for Entity.cpp:



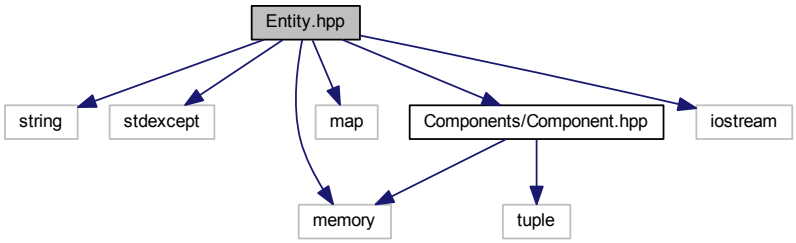
Namespaces

- ant

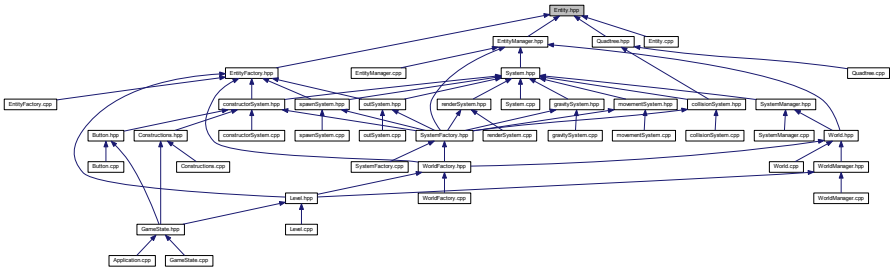
10.26 Entity.hpp File Reference

```
#include <string>
#include <stdexcept>
#include <memory>
#include <map>
#include <Components/Component.hpp>
#include <iostream>
```

Include dependency graph for Entity.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Entity

  Clase para tener los componentes, solo puede tener un componente de cada tipo.

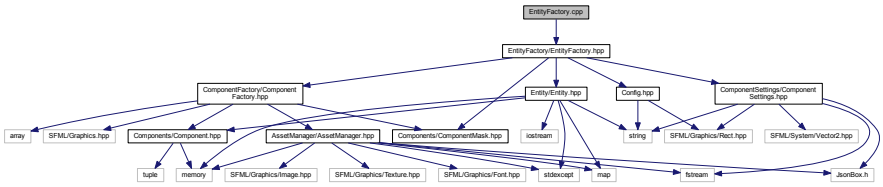
Namespaces

- ant

10.27 EntityFactory.cpp File Reference

```
#include <EntityFactory/EntityFactory.hpp>
```

Include dependency graph for EntityFactory.cpp:



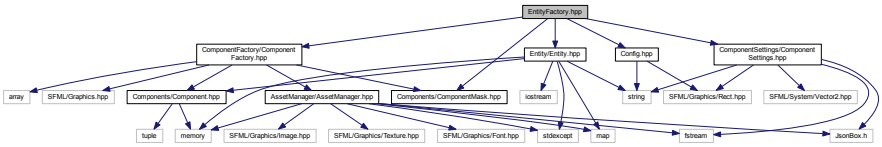
Namespaces

- ant

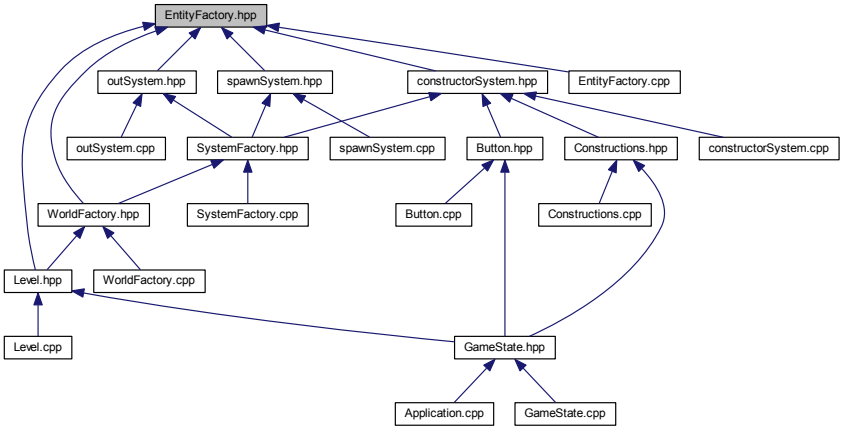
10.28 EntityFactory.hpp File Reference

```
#include <ComponentFactory/ComponentFactory.hpp>
#include <Components/ComponentMask.hpp>
#include <ComponentSettings/ComponentSettings.hpp>
#include <Entity/Entity.hpp>
#include <Config.hpp>
```

Include dependency graph for EntityFactory.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::EntityFactory

  Clase para crear las entidades del juego.

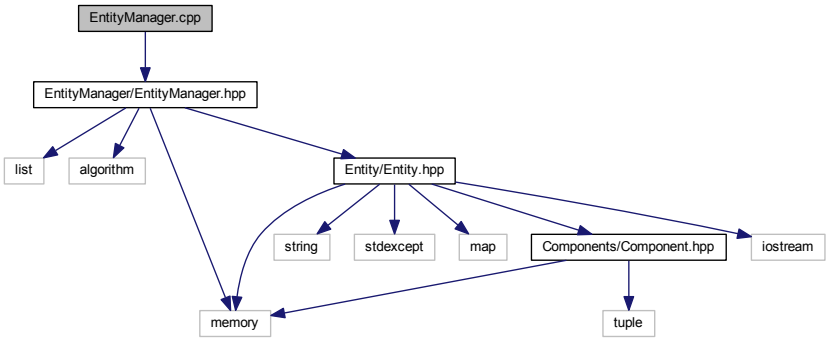
Namespaces

- ant

10.29 EntityManager.cpp File Reference

```
#include <EntityManager/EntityManager.hpp>
```

Include dependency graph for EntityManager.cpp:



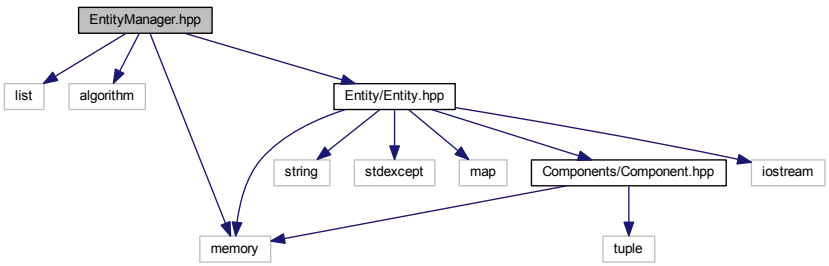
Namespaces

- ant

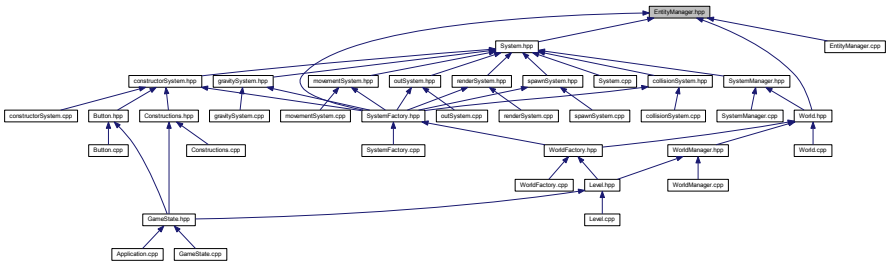
10.30 EntityManager.hpp File Reference

```
#include <list>
#include <algorithm>
#include <memory>
#include <Entity/Entity.hpp>
```

Include dependency graph for EntityManager.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::EntityManager

  Clase Para manejar las entidades.

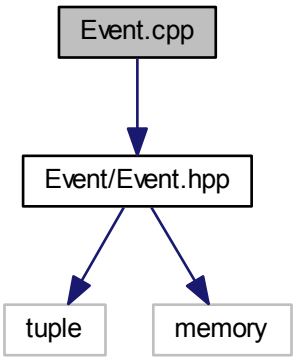
Namespaces

- ant

10.31 Event.cpp File Reference

```
#include <Event/Event.hpp>
```

Include dependency graph for Event.cpp:



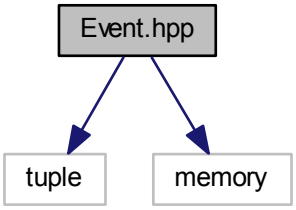
Namespaces

- ant

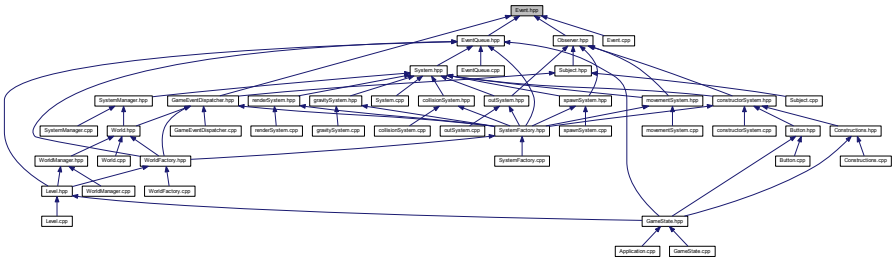
10.32 Event.hpp File Reference

```
#include <tuple>
#include <memory>
```

Include dependency graph for Event.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::baseEvent

  Clase Base de eventos.

- class ant::Event< T >

  Clase Especializada de evento.

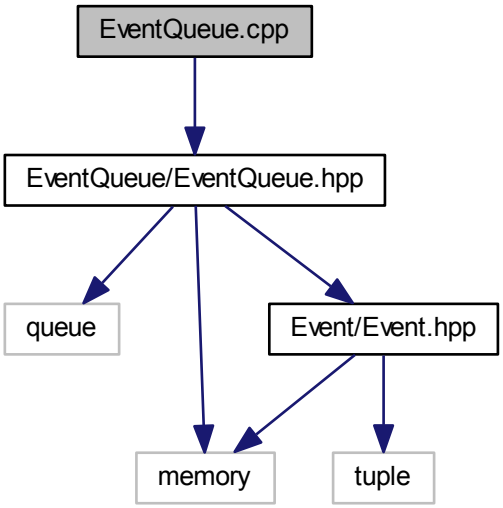
Namespaces

- ant

10.33 EventQueue.cpp File Reference

```
#include <EventQueue/EventQueue.hpp>
```

Include dependency graph for EventQueue.cpp:



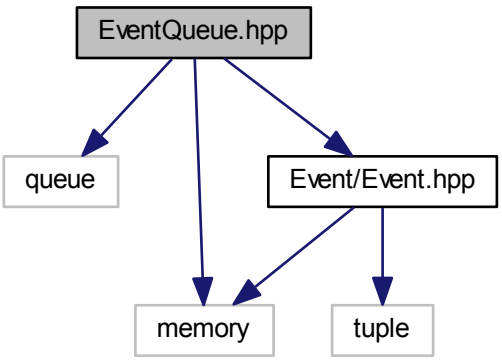
Namespaces

- ant

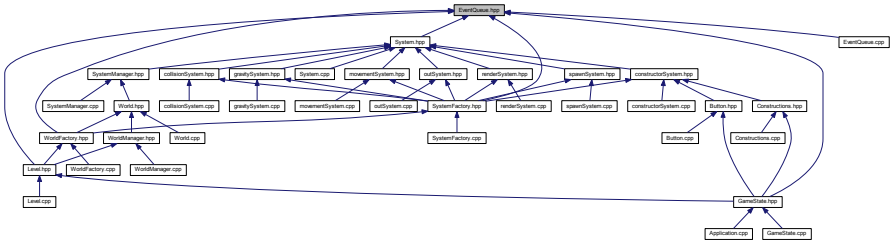
10.34 EventQueue.hpp File Reference

```
#include <queue>
#include <memory>
#include <Event/Event.hpp>
```

Include dependency graph for EventQueue.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::EventQueue

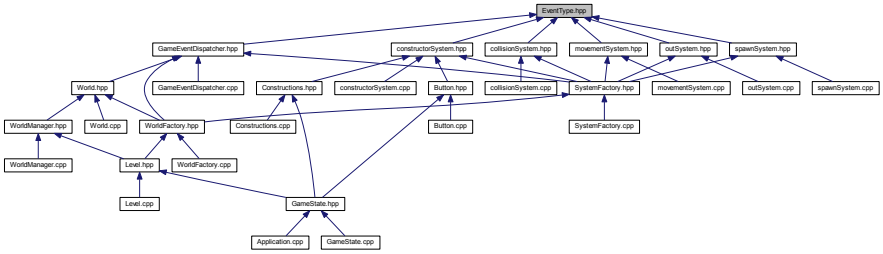
  Clase donde se almacenan los eventos para ser procesados despues.

Namespaces

- ant

10.35 EventType.hpp File Reference

This graph shows which files directly or indirectly include this file:



Classes

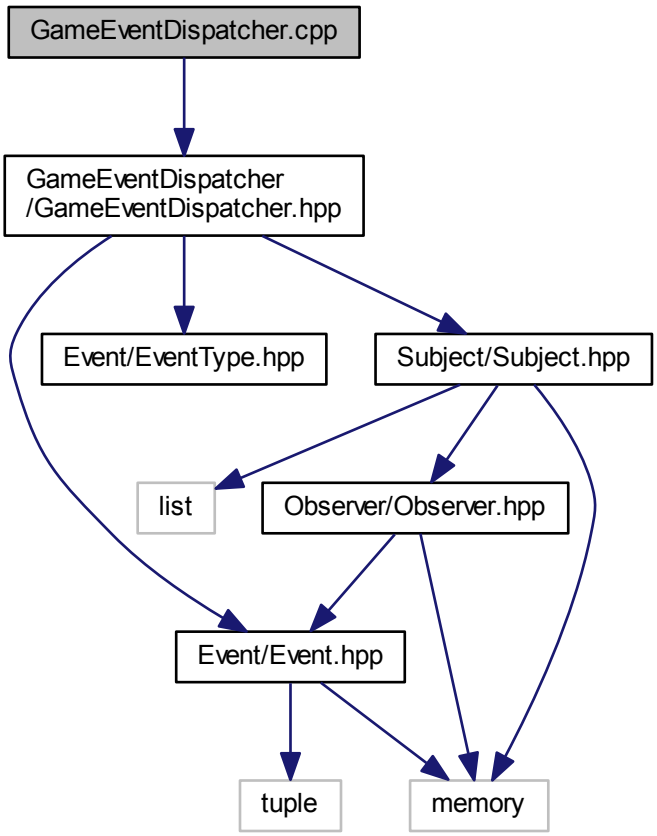
- class EventType

  Tipos de eventos.

10.36 GameEventDispatcher.cpp File Reference

```
#include <GameEventDispatcher/GameEventDispatcher.hpp>
```

Include dependency graph for GameEventDispatcher.cpp:



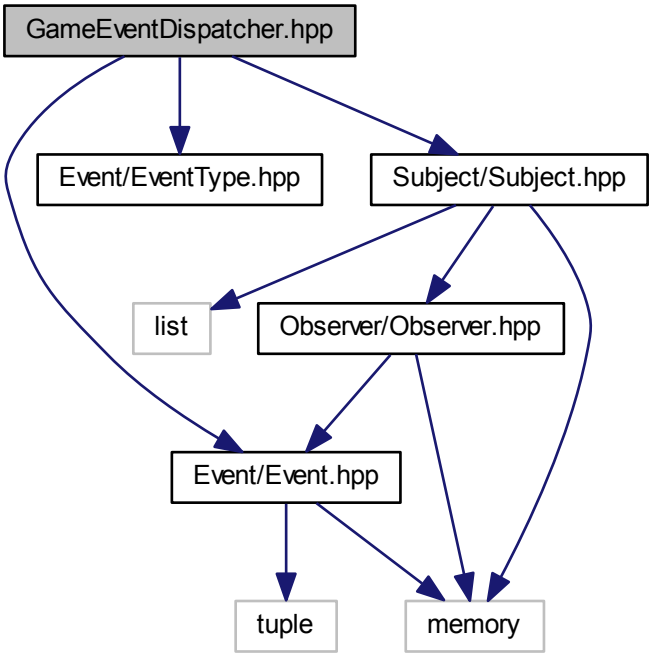
Namespaces

- ant

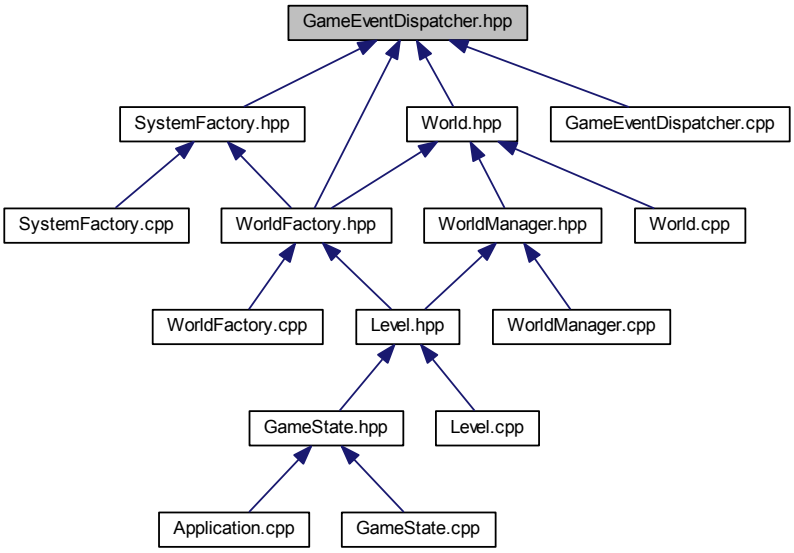
10.37 GameEventDispatcher.hpp File Reference

```
#include <Event/Event.hpp>
#include <Event/EventType.hpp>
#include <Subject/Subject.hpp>
```

Include dependency graph for GameEventDispatcher.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::GameEventDispatcher

  Clase encargada de processar eventos.

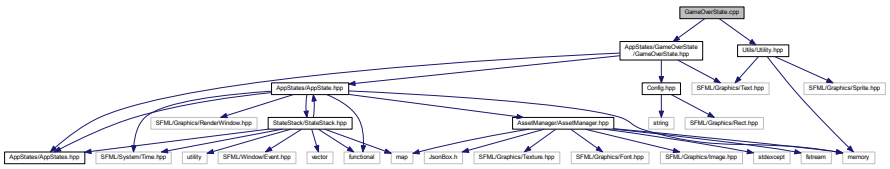
Namespaces

- ant

10.38 GameOverState.cpp File Reference

```
#include <AppStates/GameOverState/GameOverState.hpp>
#include <Utils/Utility.hpp>
```

Include dependency graph for GameOverState.cpp:



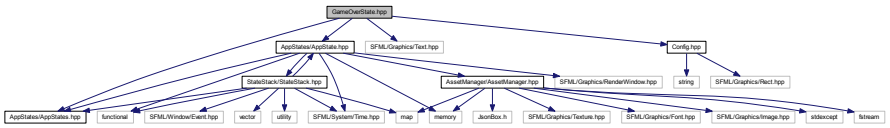
Namespaces

- ant

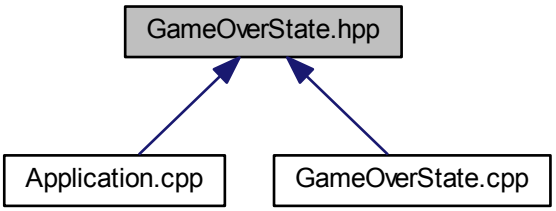
10.39 GameOverState.hpp File Reference

```
#include <AppStates/AppState.hpp>
#include <AppStates/AppStates.hpp>
#include <SFML/Graphics/Text.hpp>
#include <Config.hpp>
```

Include dependency graph for GameOverState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::GameOverState

  Muestra el mensaje de final de juego.

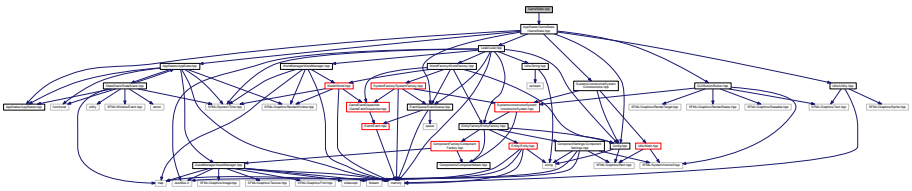
Namespaces

- ant

10.40 GameState.cpp File Reference

```
#include <AppStates/GameState/GameState.hpp>
```

Include dependency graph for GameState.cpp:



Namespaces

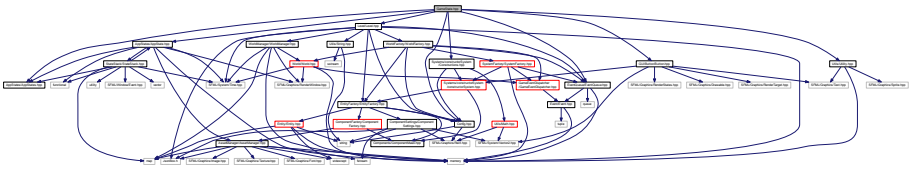
- ant

10.41 GameState.hpp File Reference

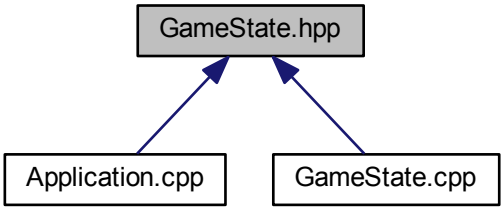
```
#include <AppStates/AppState.hpp>
```

```
#include <AppStates/AppStates.hpp>
#include <Level/Level.hpp>
#include <Config.hpp>
#include <EventQueue/EventQueue.hpp>
#include <Systems/constructorSystem/Constructions.hpp>
#include <Utils/Utility.hpp>
#include <GUI/Button/Button.hpp>
```

Include dependency graph for GameState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::GameState

  Estado de juego, Inicia el juego.

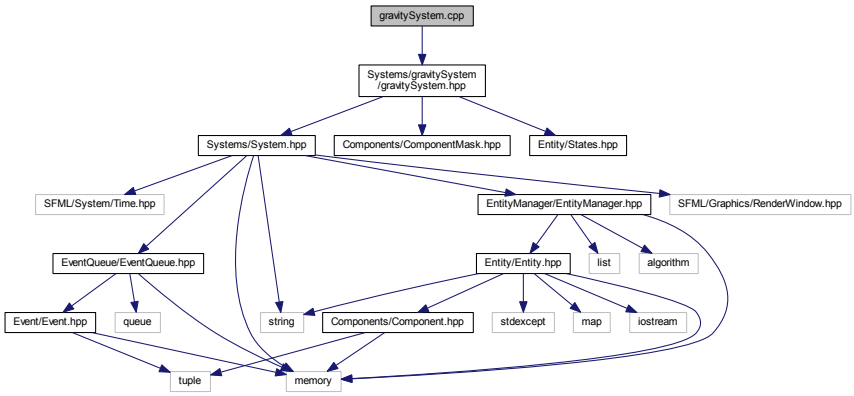
Namespaces

- ant

10.42 gravitySystem.cpp File Reference

```
#include <Systems/gravitySystem/gravitySystem.hpp>
```

Include dependency graph for gravitySystem.cpp:



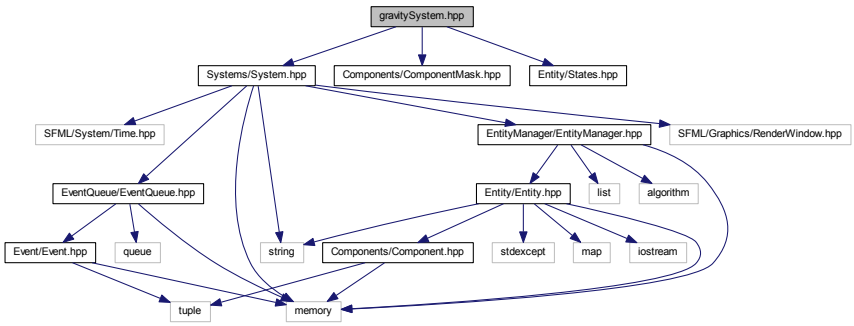
Namespaces

- ant

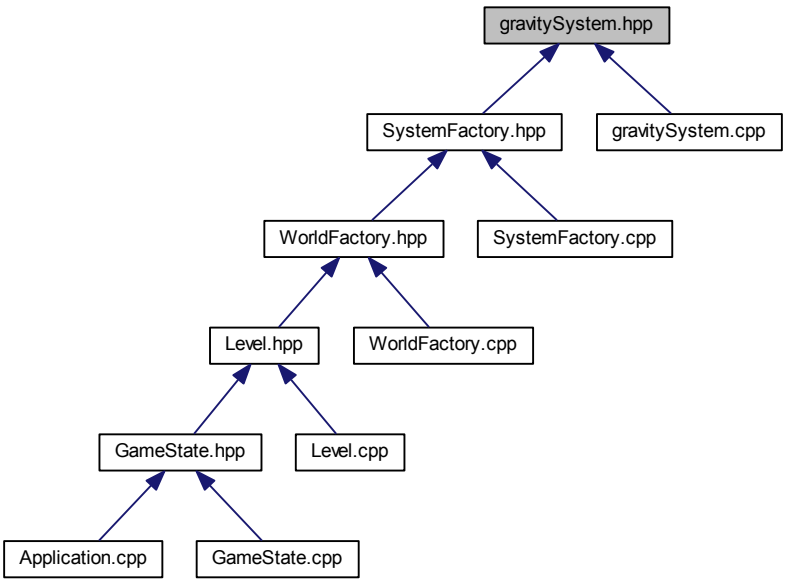
10.43 gravitySystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <Components/ComponentMask.hpp>
#include <Entity/States.hpp>
```

Include dependency graph for gravitySystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::gravitySystem

  Sistema para aplicar la gravedad a las entidades.

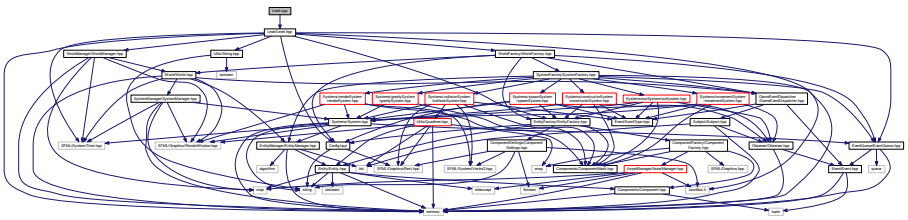
Namespaces

- ant

10.44 Level.cpp File Reference

`#include <Level/Level.hpp>`

Include dependency graph for Level.cpp:



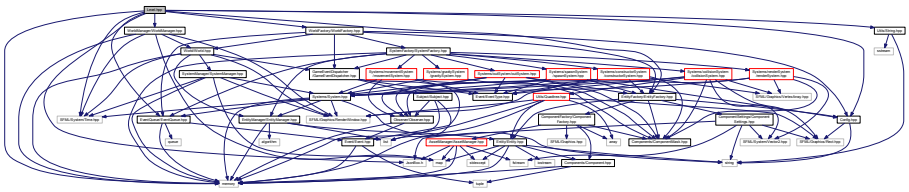
Namespaces

- ant

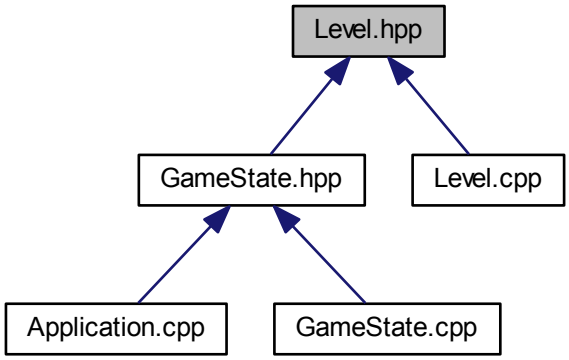
10.45 Level.hpp File Reference

```
#include <memory>
#include <SFML/System/Time.hpp>
#include <Utils/String.hpp>
#include <JsonBox.h>
#include <WorldManager/WorldManager.hpp>
#include <WorldFactory/WorldFactory.hpp>
#include <EventQueue/EventQueue.hpp>
#include <EntityFactory/EntityFactory.hpp>
#include <Config.hpp>
```

Include dependency graph for Level.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Level

  Clase que tiene el nivel del juego.

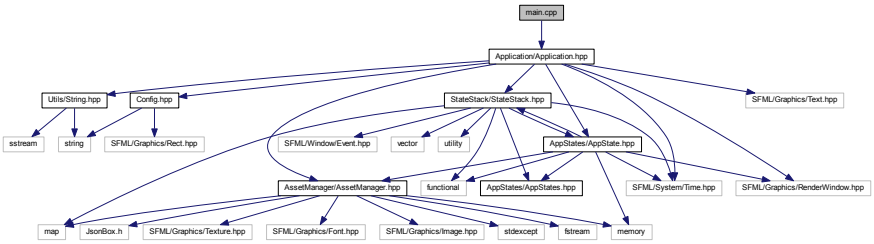
Namespaces

- ant

10.46 main.cpp File Reference

`#include <Application/Application.hpp>`

Include dependency graph for main.cpp:



Functions

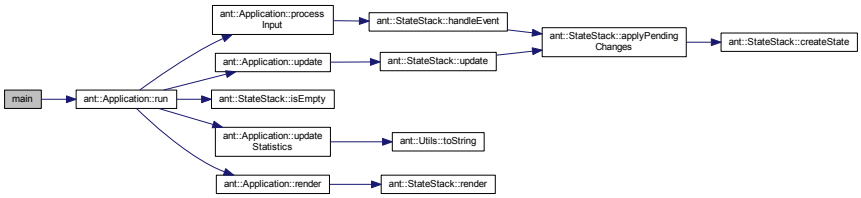
- int main ()

10.46.1 Function Documentation

10.46.1.1 int main ( )

Definition at line 3 of file main.cpp.

Here is the call graph for this function:

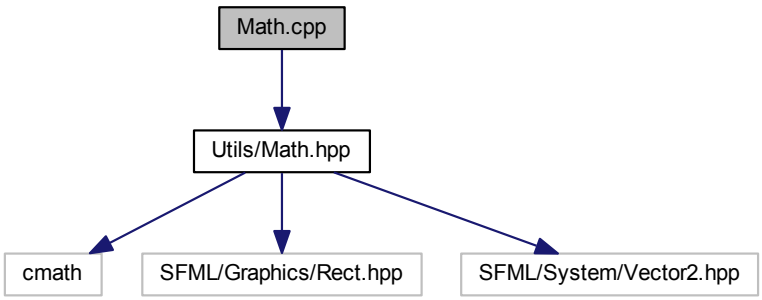


10.47 mainpage.dox File Reference

10.48 Math.cpp File Reference

`#include <Utils/Math.hpp>`

Include dependency graph for Math.cpp:



Namespaces

- ant
- ant::Utils

Functions

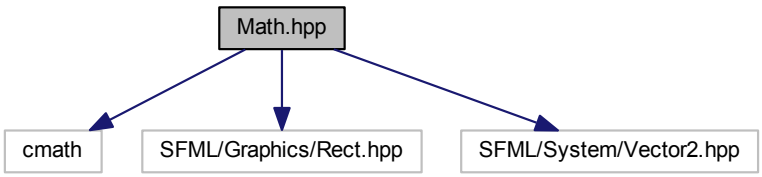
- sf::FloatRect ant::Utils::rotateRect (const sf::FloatRect &rect, const float &angle)
- sf::Vector2f ant::Utils::normalize (const sf::Vector2f &v)
- bool ant::Utils::RectContains (const sf::FloatRect &e, const sf::FloatRect &bounds)

  Comprueba que un rectangulo esta completamente dentro de otro.
- float ant::Utils::getDirection (const sf::Vector2f &currentPosition, const sf::Vector2f &wantedPosition)

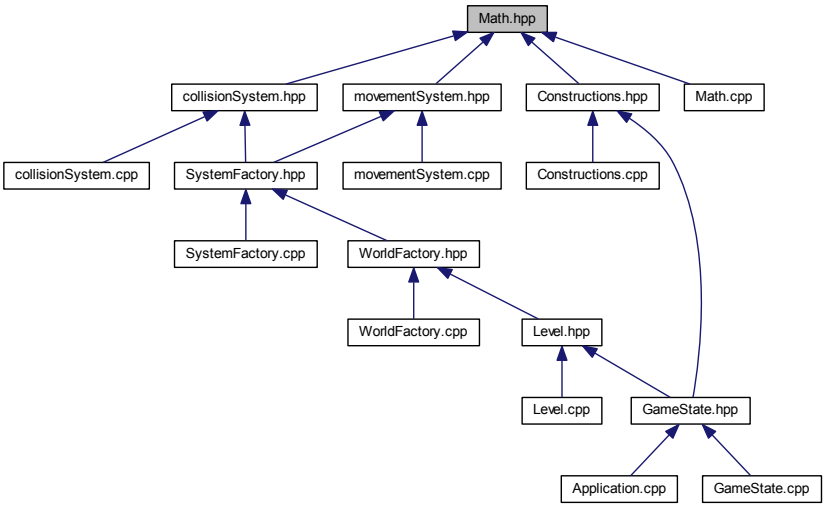
10.49 Math.hpp File Reference

```
#include <cmath>
#include <SFML/Graphics/Rect.hpp>
#include <SFML/System/Vector2.hpp>
```

Include dependency graph for Math.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- ant
- ant::Utils

Functions

- template<typename T >
  T ant::Utils::toRadians (T degrees)

  Convertira el valor introducido a radianes.
- template<typename T >
  T ant::Utils::toDegrees (T radians)
- sf::FloatRect ant::Utils::rotateRect (const sf::FloatRect &rect, const float &angle)
- template<typename T >
  T ant::Utils::getAngle (const sf::Vector2< T > &v)
- sf::Vector2f ant::Utils::normalize (const sf::Vector2f &v)
- template<typename T >
  int ant::Utils::dot (const sf::Vector2< T > &v1, const sf::Vector2< T > &v2)
- bool ant::Utils::RectContains (const sf::FloatRect &e, const sf::FloatRect &bounds)

  Comprueba que un rectangulo esta completamente dentro de otro.
- float ant::Utils::getDirection (const sf::Vector2f &currentPosition, const sf::Vector2f &wantedPosition)

Variables

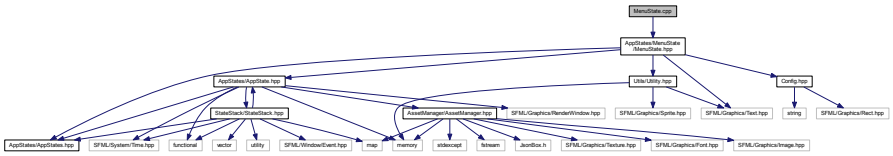
- constexpr double ant::Utils::PI = 3.14159265358979

  Definicion de PI.

10.50 MenuState.cpp File Reference

```
#include <AppStates/MenuState/MenuState.hpp>
```

Include dependency graph for MenuState.cpp:



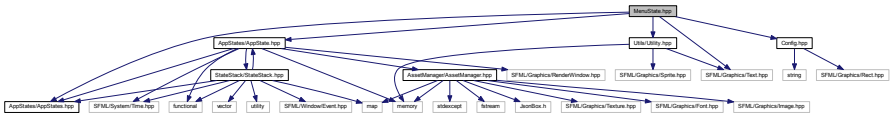
Namespaces

- ant

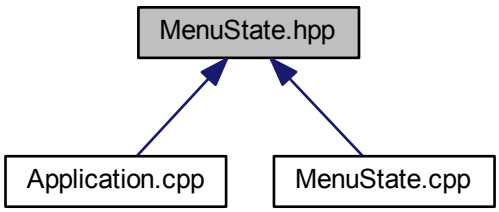
10.51 MenuState.hpp File Reference

```
#include <AppStates/AppState.hpp>
#include <AppStates/AppStates.hpp>
#include <Utils/Utility.hpp>
#include <Config.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for MenuState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::MenuState

  Clase de estado del menu.

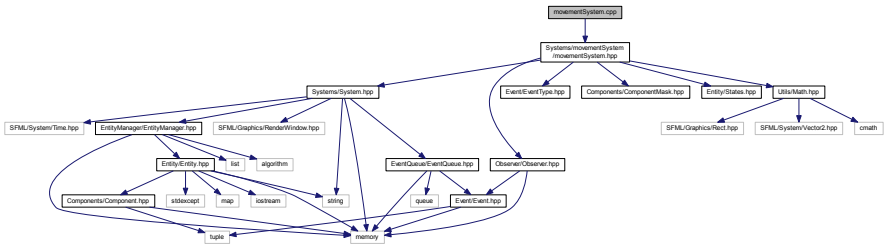
Namespaces

- ant

10.52 movementSystem.cpp File Reference

```
#include <Systems/movementSystem/movementSystem.hpp>
```

Include dependency graph for movementSystem.cpp:



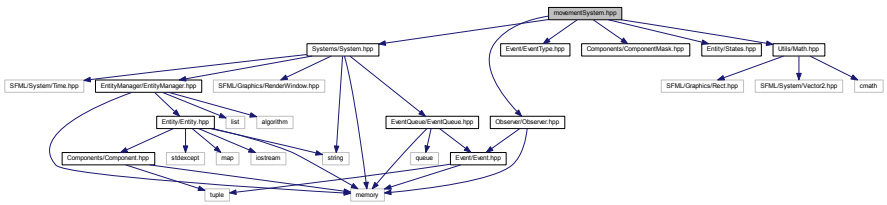
Namespaces

- ant

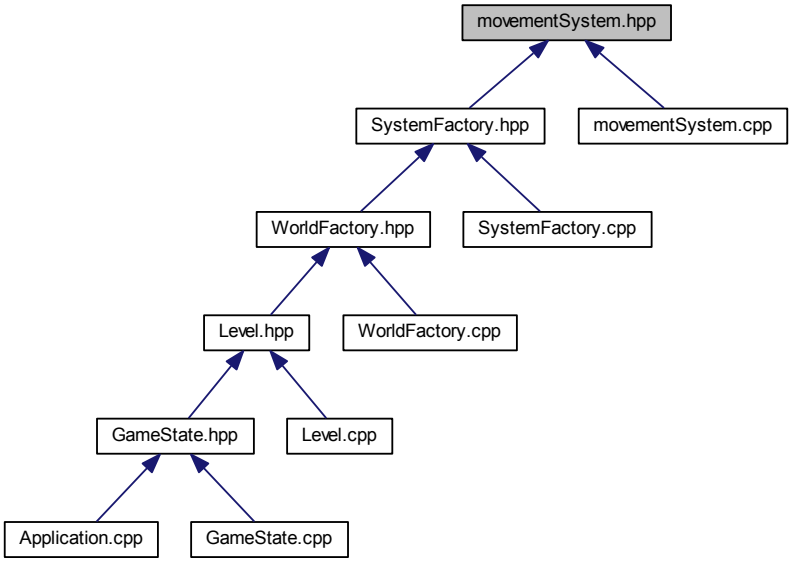
10.53 movementSystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <Observer/Observer.hpp>
#include <Event/EventType.hpp>
#include <Components/ComponentMask.hpp>
#include <Entity/States.hpp>
#include <Utils/Math.hpp>
```

Include dependency graph for movementSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::movementSystem

  Sistema encargado de hacer que las entidades se muevan.

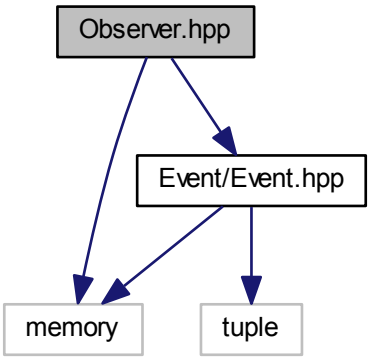
Namespaces

- ant

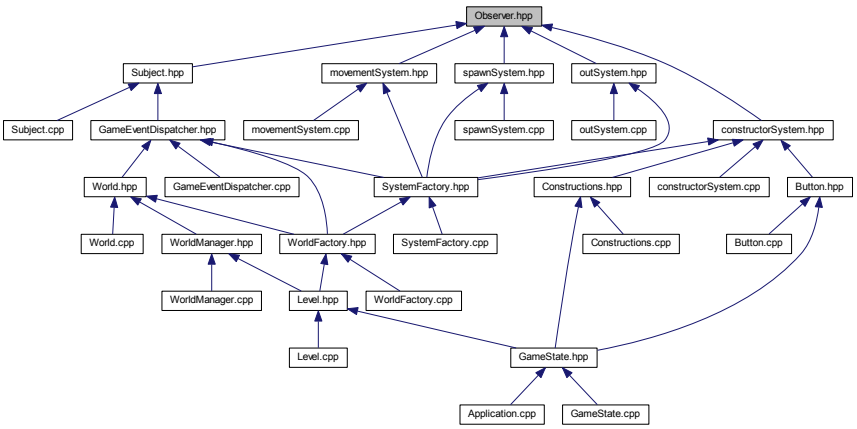
10.54 Observer.hpp File Reference

```
#include <memory>
#include <Event/Event.hpp>
```

Include dependency graph for Observer.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Observer

  Interfaz para los observadores.

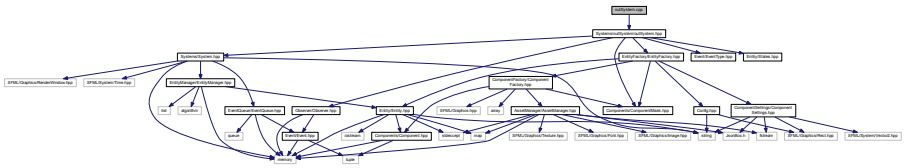
Namespaces

- ant

10.55 outSystem.cpp File Reference

```
#include <Systems/outSystem/outSystem.hpp>
```

Include dependency graph for outSystem.cpp:



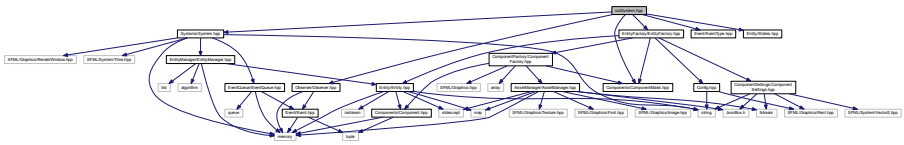
Namespaces

- ant

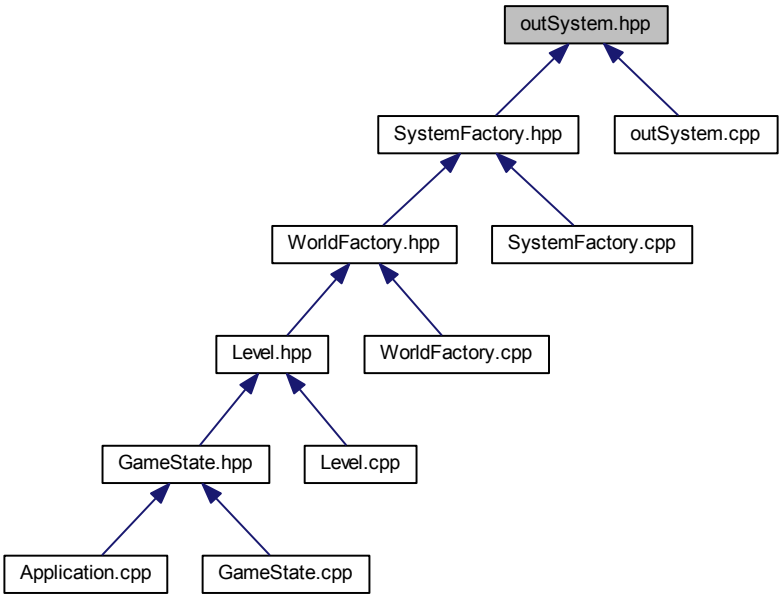
10.56 outSystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <EntityFactory/EntityFactory.hpp>
#include <Observer/Observer.hpp>
#include <Event/EventType.hpp>
#include <Entity/States.hpp>
#include <Components/ComponentMask.hpp>
```

Include dependency graph for outSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::outSystem

  Sistema encargado de quitar a las entidades que chocan con las entidades con componente OUT.

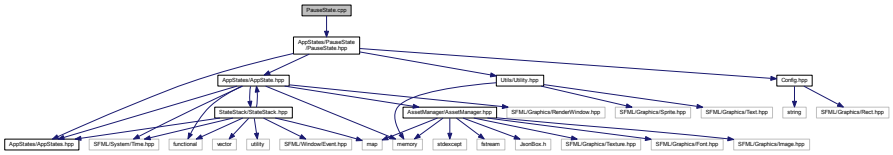
Namespaces

- ant

10.57 PauseState.cpp File Reference

```
#include <AppStates/PauseState/PauseState.hpp>
```

Include dependency graph for PauseState.cpp:



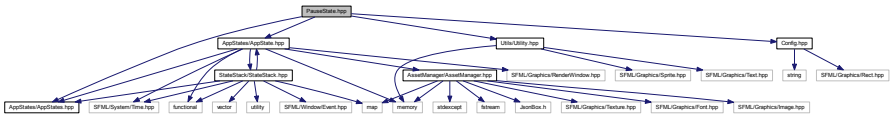
Namespaces

- ant

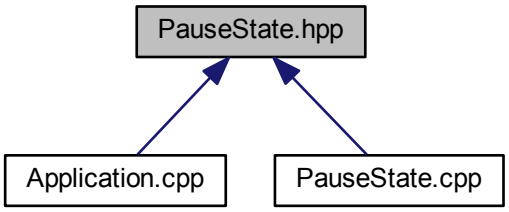
10.58 PauseState.hpp File Reference

```
#include <AppStates/AppState.hpp>
#include <AppStates/AppStates.hpp>
#include <Utils/Utility.hpp>
#include <Config.hpp>
```

Include dependency graph for PauseState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::PauseState

  Estado de pausa.

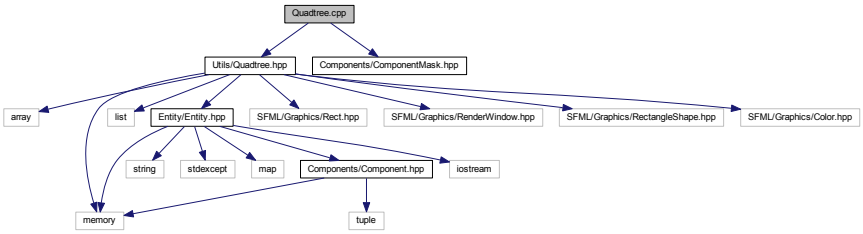
Namespaces

- ant

10.59 Quadtree.cpp File Reference

```
#include <Utils/Quadtree.hpp>
#include <Components/ComponentMask.hpp>
```

Include dependency graph for Quadtree.cpp:



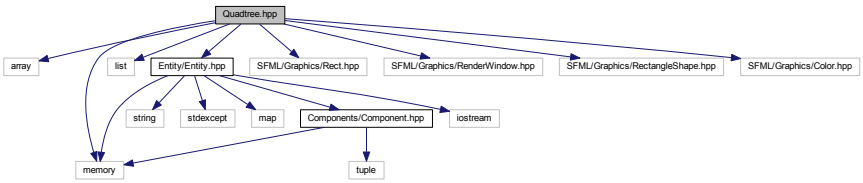
Namespaces

- ant
- ant::Utils

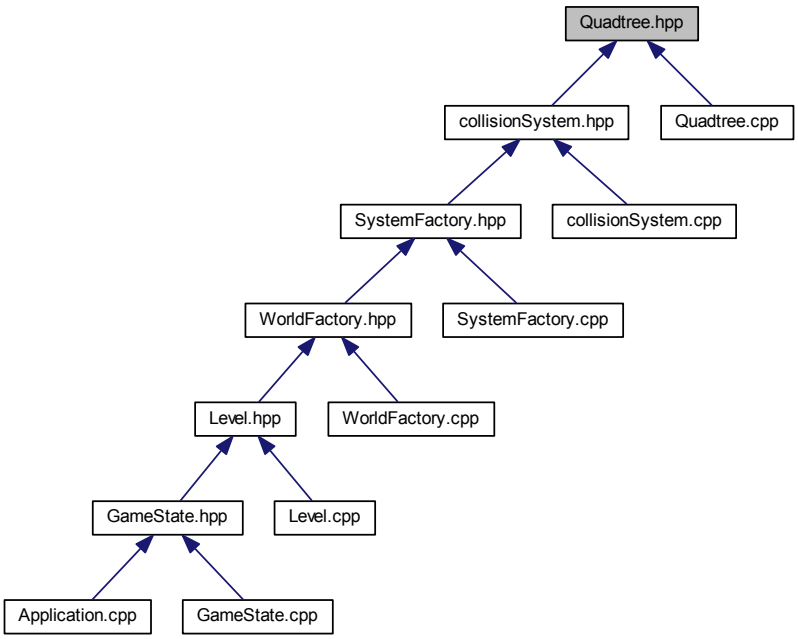
10.60 Quadtree.hpp File Reference

```
#include <array>
#include <memory>
#include <list>
#include <Entity/Entity.hpp>
#include <SFML/Graphics/Rect.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
#include <SFML/Graphics/RectangleShape.hpp>
#include <SFML/Graphics/Color.hpp>
```

Include dependency graph for Quadtree.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Utils::Quadtree

  Clase para reducir el numero de test de collisiones.

Namespaces

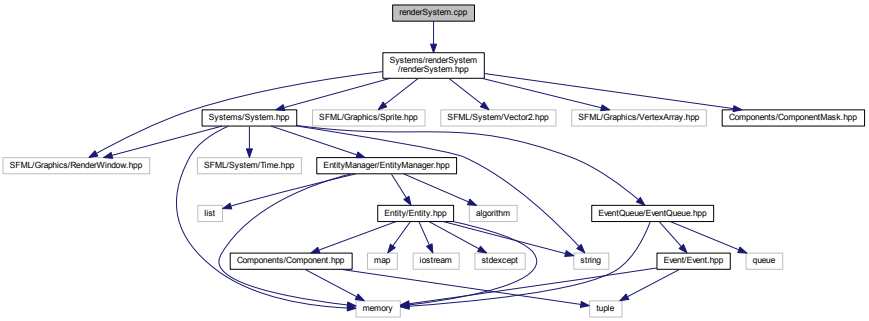
- ant
- ant::Utils

10.61 README.md File Reference

10.62 renderSystem.cpp File Reference

```
#include <Systems/renderSystem/renderSystem.hpp>
```

Include dependency graph for renderSystem.cpp:



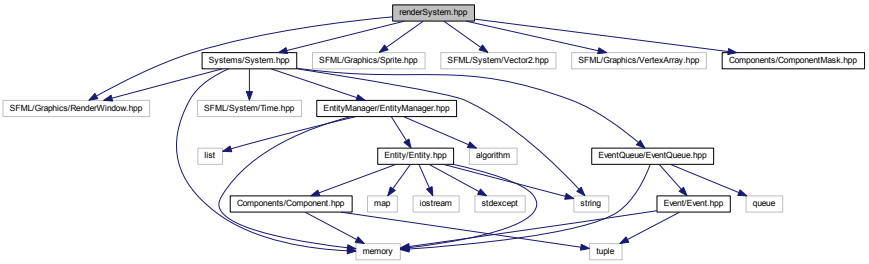
Namespaces

- ant

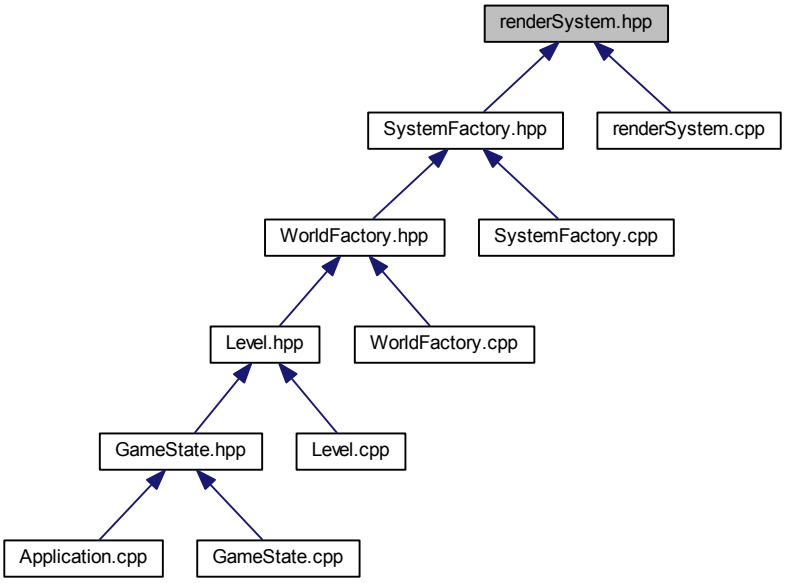
10.63 renderSystem.hpp File Reference

```
#include <SFML/Graphics/RenderWindow.hpp>
#include <SFML/Graphics/Sprite.hpp>
#include <SFML/System/Vector2.hpp>
#include <SFML/Graphics/VertexArray.hpp>
#include <Systems/System.hpp>
#include <Components/ComponentMask.hpp>
```

Include dependency graph for renderSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::renderSystem

  Sistema encargado de mostrar las entidades que tengan los componentes necesarios.

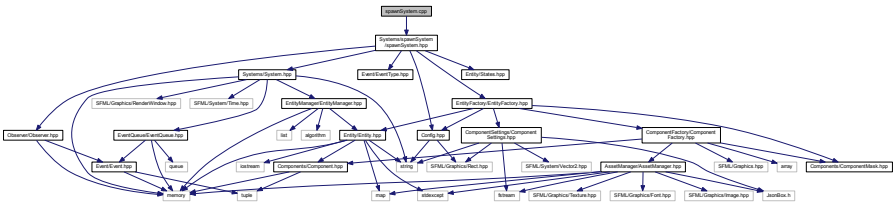
Namespaces

- ant

10.64 spawnSystem.cpp File Reference

```
#include <Systems/spawnSystem/spawnSystem.hpp>
```

Include dependency graph for spawnSystem.cpp:



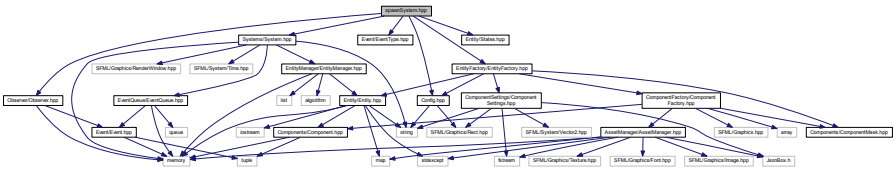
Namespaces

- ant

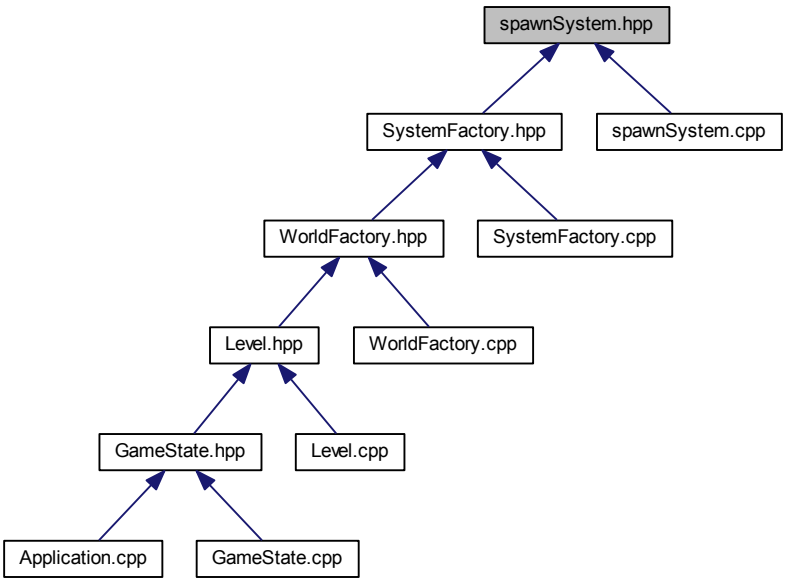
10.65 spawnSystem.hpp File Reference

```
#include <Systems/System.hpp>
#include <Observer/Observer.hpp>
#include <Event/EventType.hpp>
#include <EntityFactory/EntityFactory.hpp>
#include <Entity/States.hpp>
#include <Config.hpp>
```

Include dependency graph for spawnSystem.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::spawnSystem

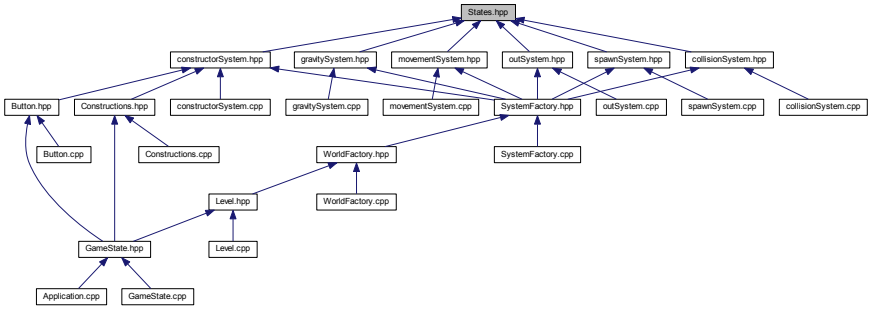
  Sistema encargado de crear las entidades sobre el tiempo establecido.

Namespaces

- ant

10.66 States.hpp File Reference

This graph shows which files directly or indirectly include this file:



Classes

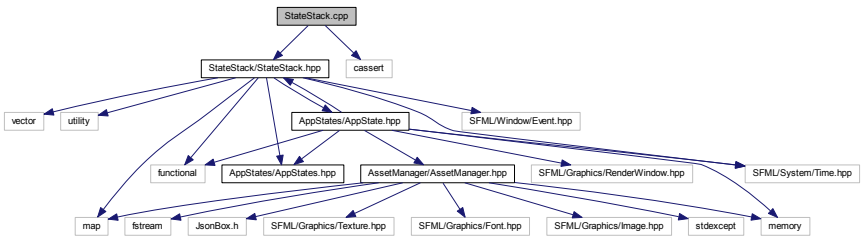
- class States

  Estados de las entidades.

10.67 StateStack.cpp File Reference

```
#include <StateStack/StateStack.hpp>
#include <cassert>
```

Include dependency graph for StateStack.cpp:



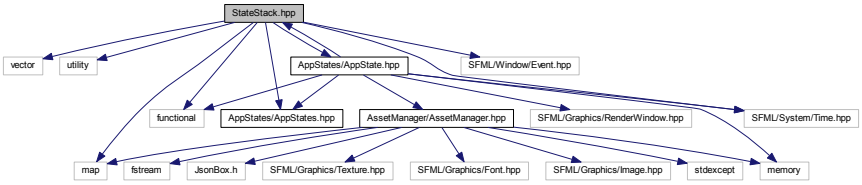
Namespaces

- ant

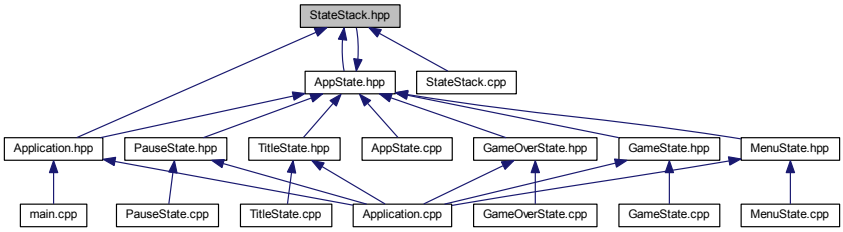
10.68 StateStack.hpp File Reference

```
#include <vector>
#include <utility>
#include <functional>
#include <map>
#include <AppStates/AppStates.hpp>
#include <AppStates/AppState.hpp>
#include <SFML/System/Time.hpp>
#include <SFML/Window/Event.hpp>
```

Include dependency graph for StateStack.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::StateStack

  Clase para almacenar los estados del juego.

- struct ant::StateStack::PendingChange

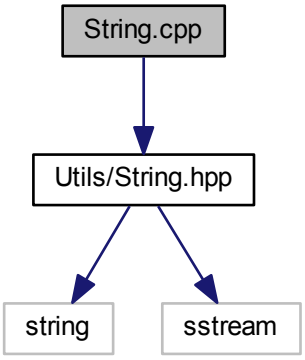
Namespaces

- ant

10.69 String.cpp File Reference

```
#include <Utils/String.hpp>
```

Include dependency graph for String.cpp:



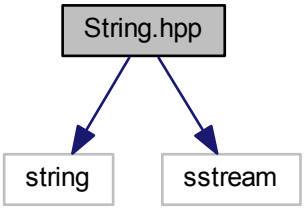
Namespaces

- ant
- ant::Utils

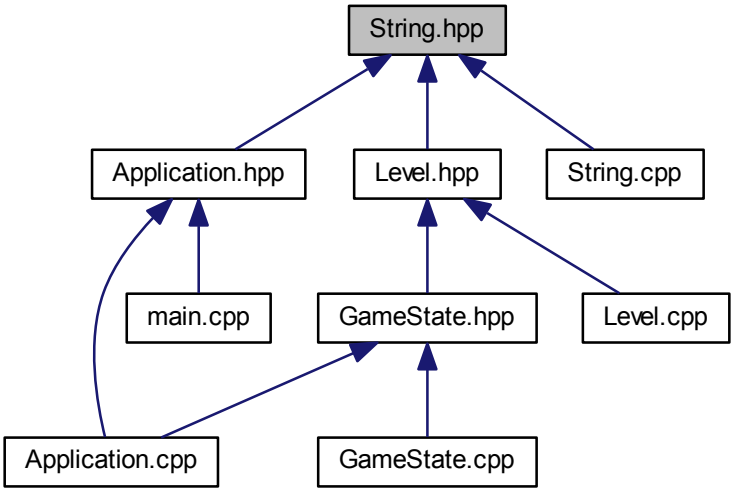
10.70 String.hpp File Reference

```
#include <string>
#include <sstream>
```

Include dependency graph for String.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- ant
- ant::Utils

Functions

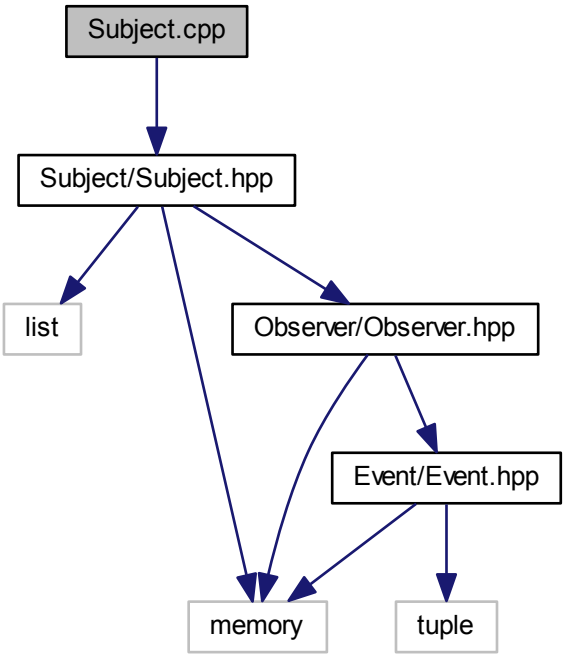
- template<typename T >
  std::string ant::Utils::toString (T s)

  Convierte el Tipo a string.

10.71 Subject.cpp File Reference

```
#include <Subject/Subject.hpp>
```

Include dependency graph for Subject.cpp:



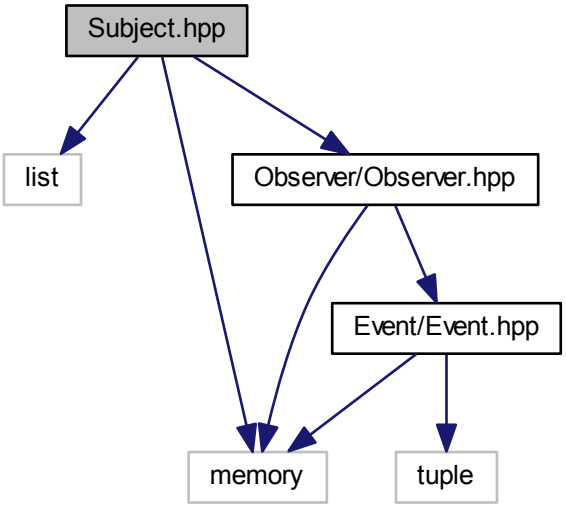
Namespaces

- ant

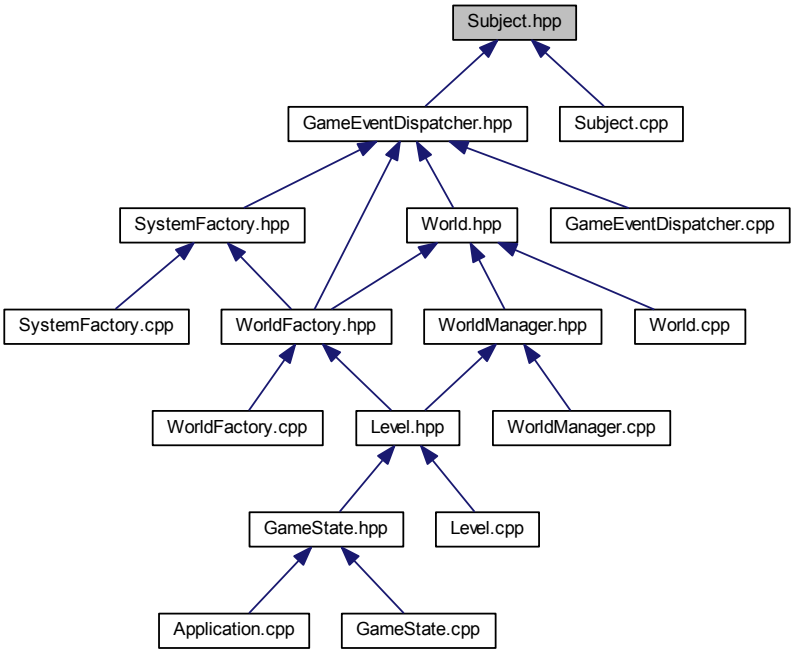
10.72 Subject.hpp File Reference

```
#include <list>
#include <memory>
#include <Observer/Observer.hpp>
```

Include dependency graph for Subject.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::Subject

  Tema al cual se suscriben los observadores Observer.

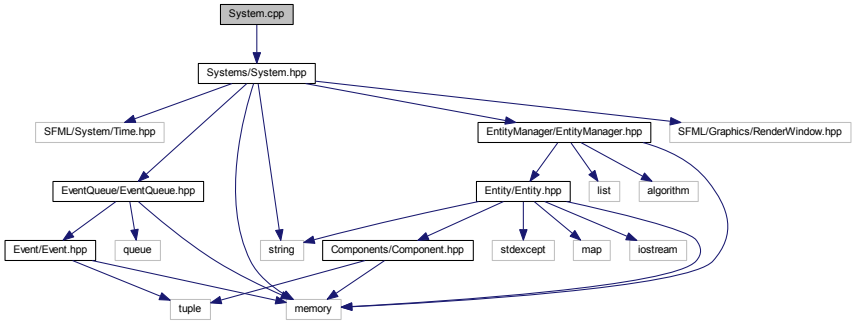
Namespaces

- ant

10.73 System.cpp File Reference

```
#include <Systems/System.hpp>
```

Include dependency graph for System.cpp:



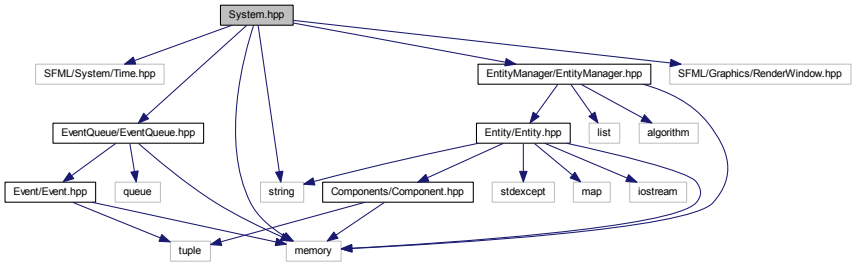
Namespaces

- ant

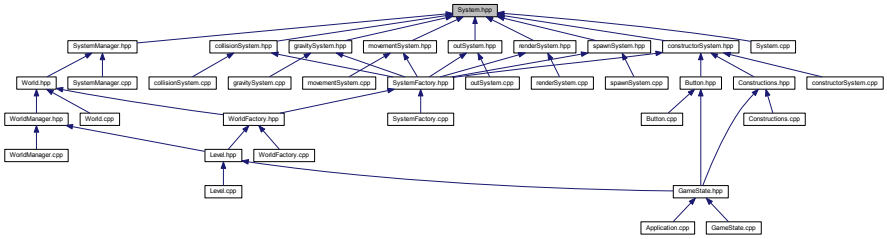
10.74 System.hpp File Reference

```
#include <SFML/System/Time.hpp>
#include <string>
#include <memory>
#include <EventQueue/EventQueue.hpp>
#include <EntityManager/EntityManager.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
```

Include dependency graph for System.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::System

  Clase base para los sistemas.

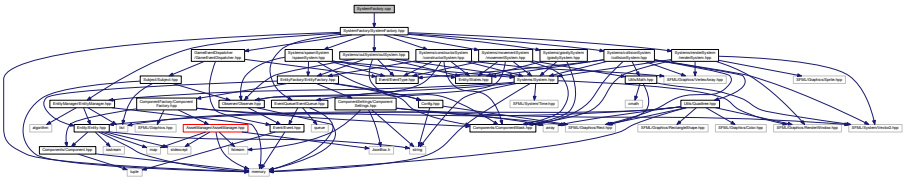
Namespaces

- ant

10.75 SystemFactory.cpp File Reference

```
#include <SystemFactory/SystemFactory.hpp>
```

Include dependency graph for SystemFactory.cpp:



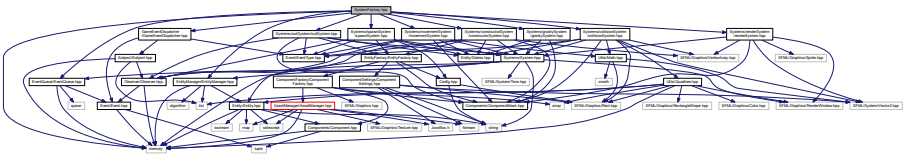
Namespaces

- ant

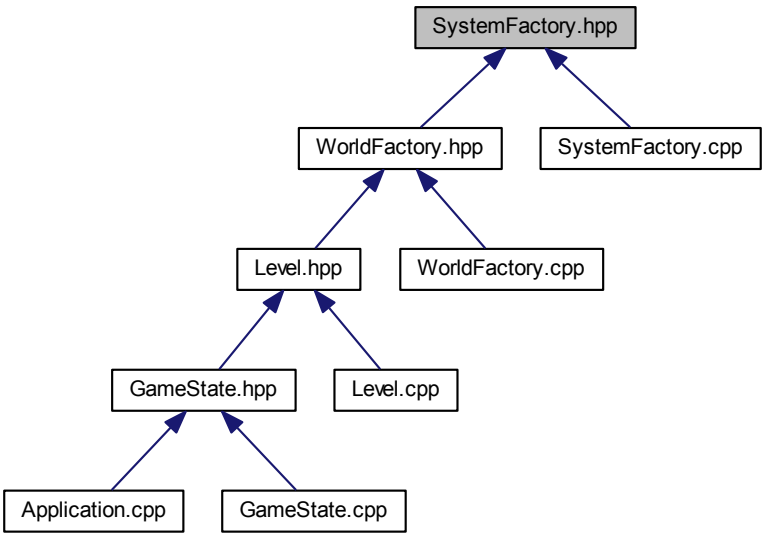
10.76 SystemFactory.hpp File Reference

```
#include <memory>
#include <EntityManager/EntityManager.hpp>
#include <GameEventDispatcher/GameEventDispatcher.hpp>
#include <EventQueue/EventQueue.hpp>
#include <Systems/collisionSystem/collisionSystem.hpp>
#include <Systems/renderSystem/renderSystem.hpp>
#include <Systems/constructorSystem/constructorSystem.hpp>
#include <Systems/movementSystem/movementSystem.hpp>
#include <Systems/gravitySystem/gravitySystem.hpp>
#include <Systems/spawnSystem/spawnSystem.hpp>
#include <Systems/outSystem/outSystem.hpp>
```

Include dependency graph for SystemFactory.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::SystemFactory

  Clase para crear sistemas.

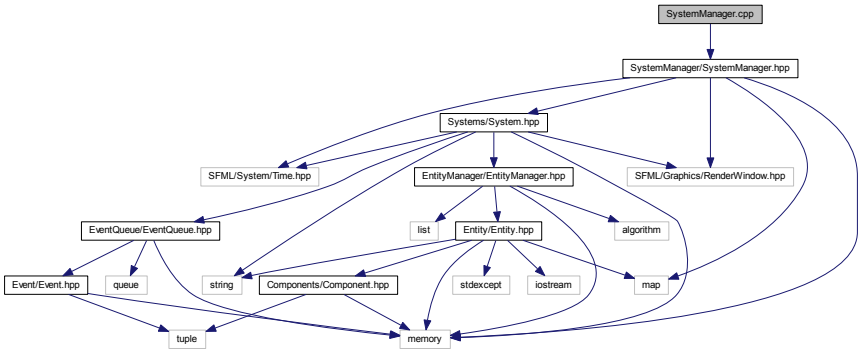
Namespaces

- ant

10.77 SystemManager.cpp File Reference

```
#include <SystemManager/SystemManager.hpp>
```

Include dependency graph for SystemManager.cpp:



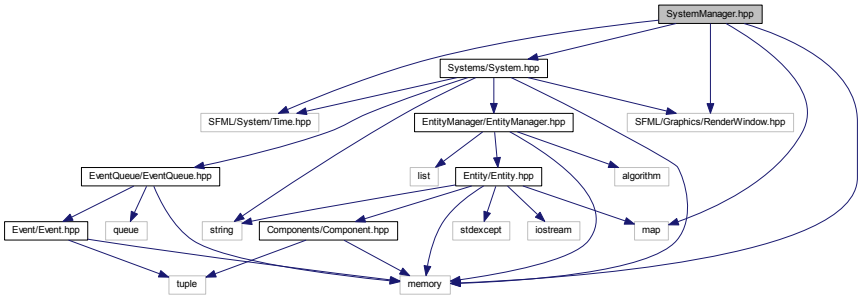
Namespaces

- ant

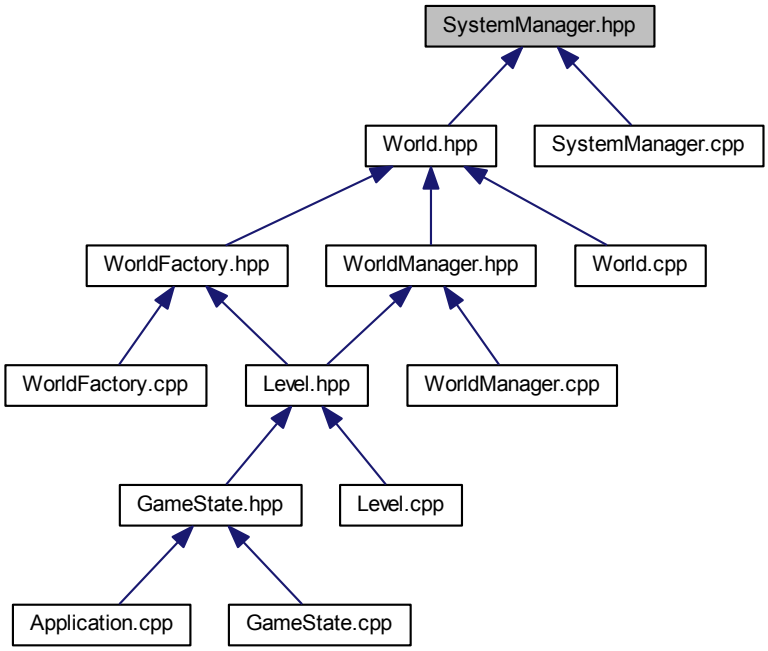
10.78 SystemManager.hpp File Reference

```
#include <map>
#include <memory>
#include <Systems/System.hpp>
#include <SFML/System/Time.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
```

Include dependency graph for SystemManager.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::SystemManager

  Clase contenedor para los systemas.

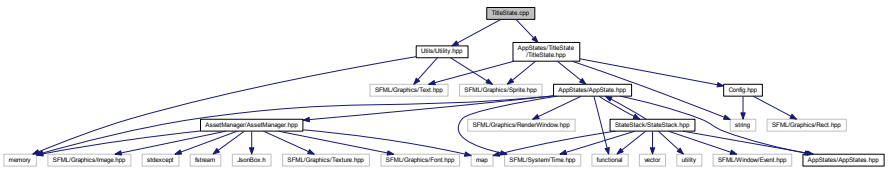
Namespaces

- ant

10.79 TitleState.cpp File Reference

```
#include <AppStates/TitleState/TitleState.hpp>
#include <Utils/Utility.hpp>
```

Include dependency graph for TitleState.cpp:



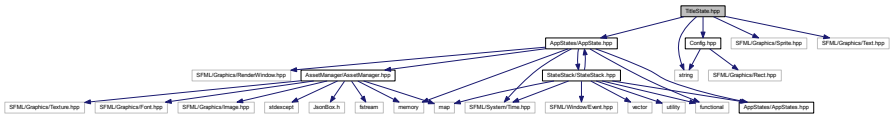
Namespaces

- ant

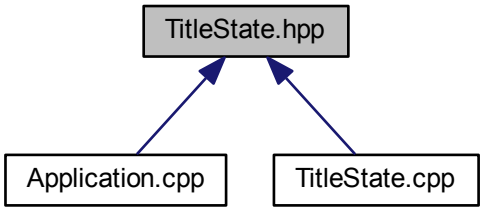
10.80 TitleState.hpp File Reference

```
#include <AppStates/AppState.hpp>
#include <string>
#include <Config.hpp>
#include <SFML/Graphics/Sprite.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for TitleState.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::TitleState

  Muestra el titulo de la aplicacion.

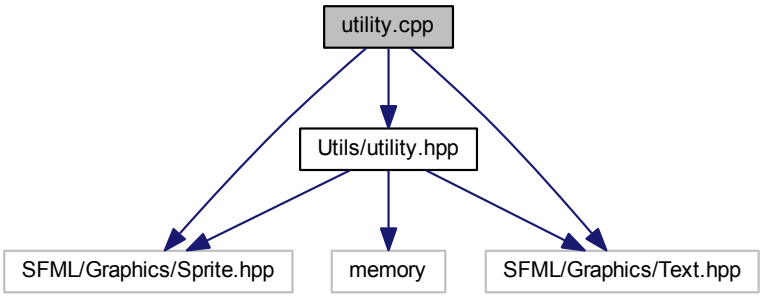
Namespaces

- ant

10.81 utility.cpp File Reference

```
#include <Utils/utility.hpp>
#include <SFML/Graphics/Sprite.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for utility.cpp:



Namespaces

- ant
- ant::Utils

Functions

- void ant::Utils::centerOrigin (sf::Sprite &sprite)

  Centra el sprite.

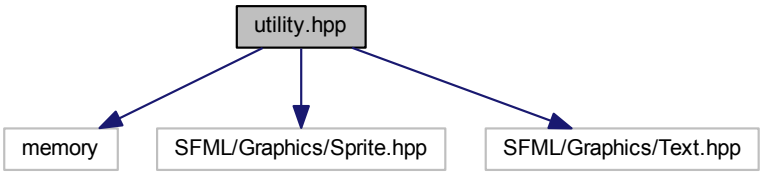
- void ant::Utils::centerOrigin (sf::Text &text)

  Centra el texto.

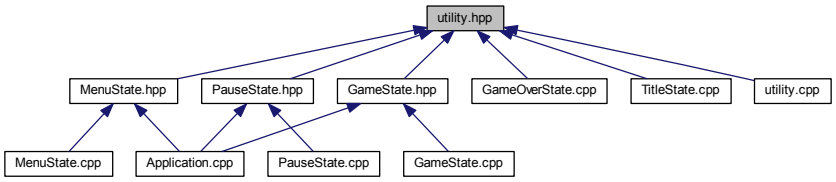
10.82 utility.hpp File Reference

```
#include <memory>
#include <SFML/Graphics/Sprite.hpp>
#include <SFML/Graphics/Text.hpp>
```

Include dependency graph for utility.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- ant
- ant::Utils

Functions

- template<typename T , typename... Args>
  std::unique_ptr< T > ant::Utils::makeUniquePtr (Args &&...args)

  Utilidad para crear un puntero unico.
- template<typename T , typename... Args>
  std::shared_ptr< T > ant::Utils::makeSharedPtr (Args &&...args)

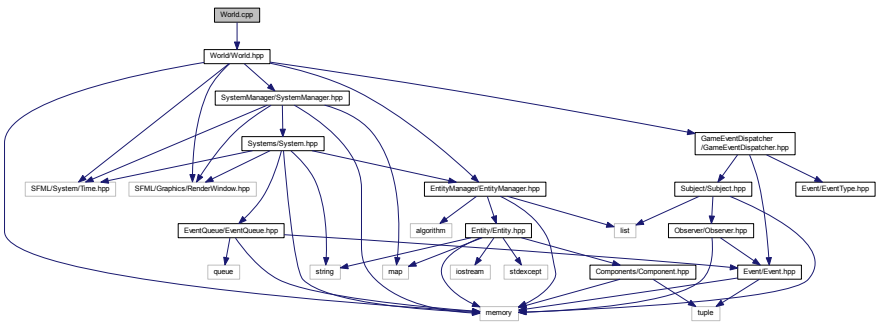
  Utilidad para crear un puntero Compartido.
- void ant::Utils::centerOrigin (sf::Text &text)

  Centra el texto.
- void ant::Utils::centerOrigin (sf::Sprite &sprite)

  Centra el sprite.

10.83 World.cpp File Reference

`#include <World/World.hpp>`
Include dependency graph for World.cpp:



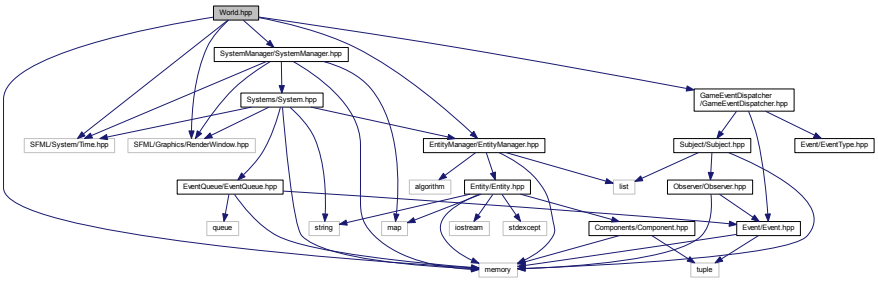
Namespaces

- ant

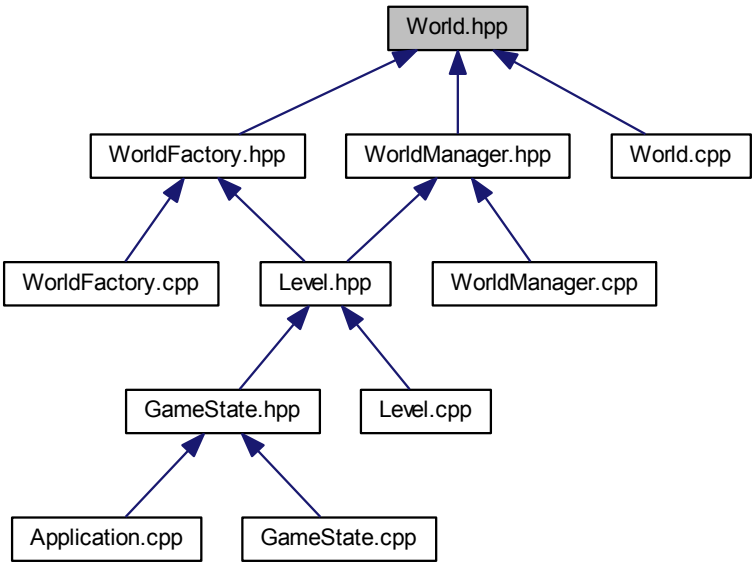
10.84 World.hpp File Reference

```
#include <SFML/System/Time.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
#include <memory>
#include <EntityManager/EntityManager.hpp>
#include <SystemManager/SystemManager.hpp>
#include <GameEventDispatcher/GameEventDispatcher.hpp>
```

Include dependency graph for World.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::World

  Clase para crear escenarios.

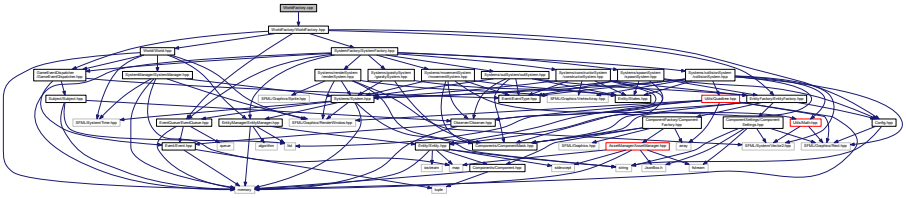
Namespaces

- ant

10.85 WorldFactory.cpp File Reference

```
#include <WorldFactory/WorldFactory.hpp>
```

Include dependency graph for WorldFactory.cpp:



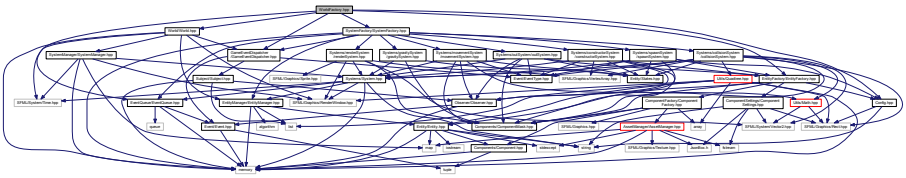
Namespaces

- ant

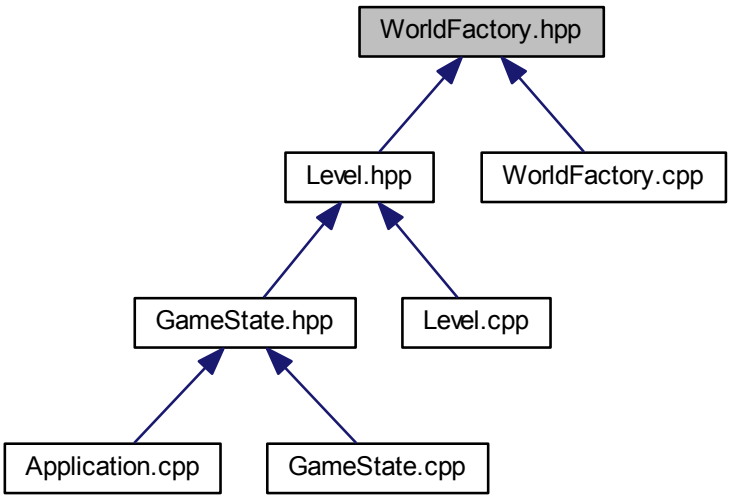
10.86 WorldFactory.hpp File Reference

```
#include <World/World.hpp>
#include <Config.hpp>
#include <EventQueue/EventQueue.hpp>
#include <GameEventDispatcher/GameEventDispatcher.hpp>
#include <EntityFactory/EntityFactory.hpp>
#include <SystemFactory/SystemFactory.hpp>
```

Include dependency graph for WorldFactory.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::WorldFactory

  Clase para crear mundos.

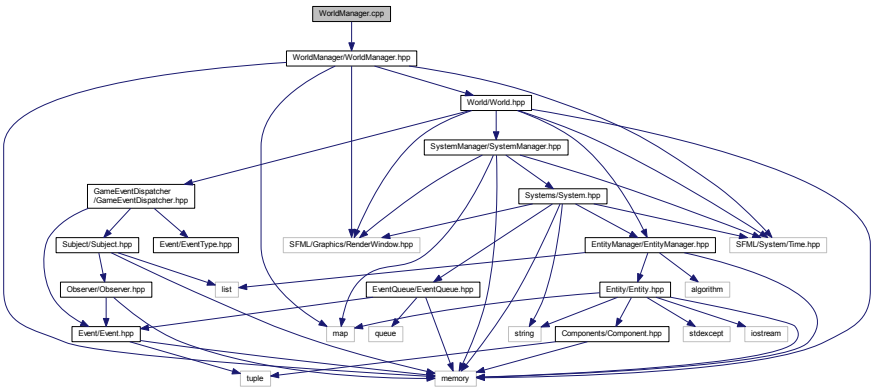
Namespaces

- ant

10.87 WorldManager.cpp File Reference

`#include <WorldManager/WorldManager.hpp>`

Include dependency graph for WorldManager.cpp:



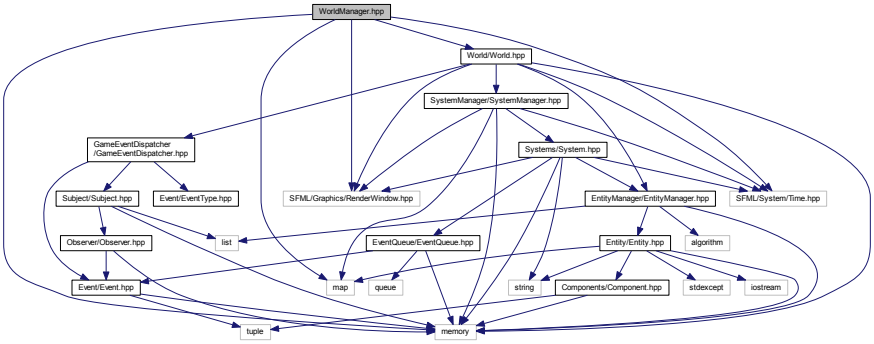
Namespaces

- ant

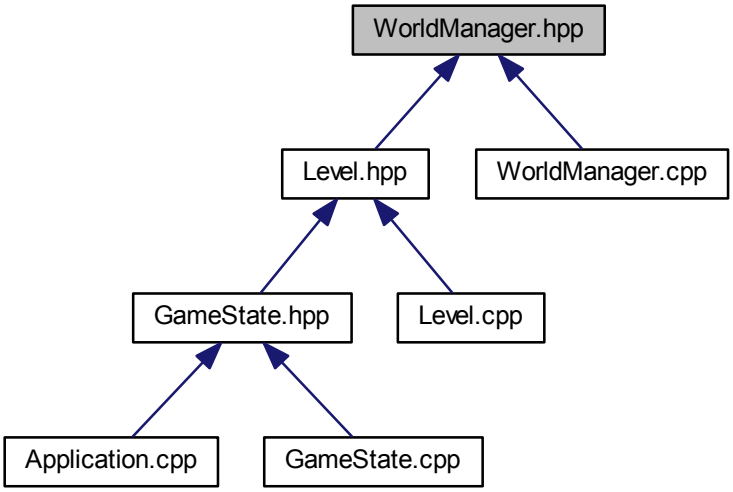
10.88 WorldManager.hpp File Reference

```
#include <World/World.hpp>
#include <map>
#include <memory>
#include <SFML/System/Time.hpp>
#include <SFML/Graphics/RenderWindow.hpp>
```

Include dependency graph for WorldManager.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ant::WorldManager

Clase Contenedor para los mundos.

Namespaces

- ant