Legend of the Great Unwashed

v0.1

Generated by Doxygen 1.8.9.1

Sun Nov 22 2015 11:36:57

Contents

1 Namespace Index **1**

1.1 Namespace List . . . 1

2 Hierarchical Index **3**

2.1 Class Hierarchy . . . 3

3 Class Index **5**

3.1 Class List . . . 5

4 File Index **7**

4.1 File List . . . 7

5 Namespace Documentation **9**

5.1 MainNS Namespace Reference . . . 9

5.1.1 Function Documentation . . . 9

5.1.1.1 logError . . . 9

5.2 mediawrap Namespace Reference . . . 9

5.3 teamusa Namespace Reference . . . 9

5.3.1 Typedef Documentation . . . 11

5.3.1.1 ActorList . . . 11

5.3.1.2 AudioID . . . 11

5.3.1.3 BaseActorPtr . . . 11

5.3.1.4 Region . . . 11

5.3.1.5 TextureID . . . 11

5.3.2 Enumeration Type Documentation . . . 11

5.3.2.1 ActorEventType . . . 11

5.3.2.2 CursorStyle . . . 11

6 Class Documentation **13**

6.1 teamusa::ActorEvent Class Reference . . . 13

6.1.1 Detailed Description . . . 13

6.1.2 Constructor & Destructor Documentation . . . 13

6.1.2.1 ActorEvent . . . 13

6.1.3 Member Data Documentation . . . 13
6.1.3.1 type . . . 13
6.1.3.2 value . . . 13
6.2 teamusa::ActorVideo Struct Reference . . . 13
6.2.1 Constructor & Destructor Documentation . . . 14
6.2.1.1 ActorVideo . . . 14
6.2.2 Member Data Documentation . . . 14
6.2.2.1 layer . . . 14
6.2.2.2 textureID . . . 14
6.3 teamusa::AudioEngine Class Reference . . . 14
6.3.1 Detailed Description . . . 15
6.3.2 Member Function Documentation . . . 15
6.3.2.1 deleteSound . . . 15
6.3.2.2 deleteSoundGroup . . . 15
6.3.2.3 loadSound . . . 15
6.3.2.4 playSound . . . 15
6.3.2.5 playStream . . . 15
6.3.3 Member Data Documentation . . . 15
6.3.3.1 audioPlayer . . . 15
6.3.3.2 coreResources . . . 15
6.3.3.3 levelResources . . . 15
6.3.3.4 MAX_RESERVED_ID . . . 16
6.4 mediawrap::AudioPlayer Class Reference . . . 16
6.4.1 Detailed Description . . . 17
6.4.2 Member Typedef Documentation . . . 17
6.4.2.1 AudioID . . . 17
6.4.3 Constructor & Destructor Documentation . . . 17
6.4.3.1 AudioPlayer . . . 17
6.4.3.2 ~AudioPlayer . . . 17
6.4.4 Member Function Documentation . . . 17
6.4.4.1 clear_samples . . . 17
6.4.4.2 delete_sample . . . 17
6.4.4.3 load_sample . . . 17
6.4.4.4 load_stream . . . 17
6.4.4.5 play_sample . . . 18
6.4.4.6 stream_audio . . . 18
6.4.5 Member Data Documentation . . . 18
6.4.5.1 audio_buffer . . . 18
6.4.5.2 audio_channels . . . 18
6.4.5.3 audio_format . . . 18

6.4.5.4 audio_rate . . . 18
6.4.5.5 audio_samples . . . 18
6.4.5.6 audio_stream . . . 18
6.5 teamusa::AudioStreamActor Class Reference . . . 18
6.5.1 Detailed Description . . . 19
6.5.2 Constructor & Destructor Documentation . . . 19
6.5.2.1 AudioStreamActor . . . 19
6.5.2.2 ~AudioStreamActor . . . 19
6.5.3 Member Function Documentation . . . 19
6.5.3.1 getPath . . . 19
6.5.3.2 step . . . 19
6.5.4 Member Data Documentation . . . 19
6.5.4.1 activated . . . 19
6.5.4.2 path . . . 19
6.6 teamusa::BaseActor Class Reference . . . 20
6.6.1 Detailed Description . . . 21
6.6.2 Constructor & Destructor Documentation . . . 21
6.6.2.1 BaseActor . . . 21
6.6.2.2 ~BaseActor . . . 21
6.6.3 Member Function Documentation . . . 21
6.6.3.1 getLayer . . . 21
6.6.3.2 getRegion . . . 21
6.6.3.3 getTextureID . . . 21
6.6.3.4 hasVideo . . . 21
6.6.3.5 isInBounds . . . 22
6.6.3.6 onClick . . . 23
6.6.3.7 onHover . . . 23
6.6.3.8 setRegion . . . 23
6.6.3.9 step . . . 23
6.6.4 Member Data Documentation . . . 24
6.6.4.1 mAudioID . . . 24
6.6.4.2 mRegion . . . 24
6.6.4.3 mVideo . . . 24
6.7 teamusa::DelayedAudioActor Class Reference . . . 24
6.7.1 Detailed Description . . . 25
6.7.2 Constructor & Destructor Documentation . . . 25
6.7.2.1 DelayedAudioActor . . . 25
6.7.2.2 ~DelayedAudioActor . . . 25
6.7.3 Member Function Documentation . . . 25
6.7.3.1 step . . . 25

6.7.4 Member Data Documentation . . . 25
6.7.4.1 audioId . . . 25
6.7.4.2 currentStep . . . 25
6.7.4.3 delaySteps . . . 25
6.8 teamusa::DelayedVideoActor Class Reference . . . 25
6.8.1 Detailed Description . . . 26
6.8.2 Constructor & Destructor Documentation . . . 26
6.8.2.1 DelayedVideoActor . . . 26
6.8.2.2 ~DelayedVideoActor . . . 26
6.8.3 Member Function Documentation . . . 26
6.8.3.1 step . . . 26
6.8.4 Member Data Documentation . . . 26
6.8.4.1 currentStep . . . 26
6.8.4.2 delaySteps . . . 26
6.8.4.3 disappear . . . 26
6.8.4.4 textureId . . . 26
6.9 teamusa::Engine Class Reference . . . 27
6.9.1 Detailed Description . . . 27
6.9.2 Member Typedef Documentation . . . 28
6.9.2.1 ActorEventHandler . . . 28
6.9.3 Constructor & Destructor Documentation . . . 28
6.9.3.1 Engine . . . 28
6.9.3.2 ~Engine . . . 28
6.9.4 Member Function Documentation . . . 28
6.9.4.1 freeAndLoadLevel . . . 28
6.9.4.2 getMouseClickState . . . 28
6.9.4.3 getMouseCoordinates . . . 28
6.9.4.4 handleEvent . . . 28
6.9.4.5 onChangeScene . . . 28
6.9.4.6 onDisplayText . . . 28
6.9.4.7 onExitGame . . . 28
6.9.4.8 onLoadGame . . . 28
6.9.4.9 onLoadLevel . . . 28
6.9.4.10 onNewGame . . . 28
6.9.4.11 onPlayAudio . . . 28
6.9.4.12 onStreamAudio . . . 28
6.9.4.13 render . . . 28
6.9.4.14 run . . . 29
6.9.5 Member Data Documentation . . . 29
6.9.5.1 mActorEventHandlers . . . 29

6.9.5.2 mAudioEngine . . . 29
6.9.5.3 mIsRunning . . . 29
6.9.5.4 mLevel . . . 29
6.9.5.5 mPlayer . . . 29
6.9.5.6 mSerializer . . . 29
6.9.5.7 mVideoEngine . . . 29
6.10 teamusa::ExampleActor Class Reference . . . 29
6.10.1 Detailed Description . . . 30
6.10.2 Constructor & Destructor Documentation . . . 30
6.10.2.1 ExampleActor . . . 30
6.10.2.2 ∼ExampleActor . . . 30
6.10.3 Member Function Documentation . . . 30
6.10.3.1 onClick . . . 30
6.10.3.2 onHover . . . 30
6.10.3.3 step . . . 30
6.11 teamusa::GameSaveSerializer Class Reference . . . 30
6.11.1 Detailed Description . . . 31
6.11.2 Constructor & Destructor Documentation . . . 31
6.11.2.1 GameSaveSerializer . . . 31
6.11.2.2 ∼GameSaveSerializer . . . 31
6.11.3 Member Function Documentation . . . 31
6.11.3.1 load . . . 31
6.11.3.2 save . . . 31
6.11.3.3 saveInThread . . . 31
6.11.3.4 setSlot . . . 31
6.11.4 Member Data Documentation . . . 31
6.11.4.1 fileLock . . . 31
6.11.4.2 slot . . . 31
6.12 teamusa::InventoryItemActor Class Reference . . . 32
6.12.1 Detailed Description . . . 32
6.12.2 Constructor & Destructor Documentation . . . 32
6.12.2.1 InventoryItemActor . . . 32
6.12.2.2 ∼InventoryItemActor . . . 32
6.12.3 Member Function Documentation . . . 32
6.12.3.1 onClick . . . 32
6.12.3.2 onHover . . . 32
6.12.3.3 step . . . 33
6.12.4 Member Data Documentation . . . 33
6.12.4.1 itemID . . . 33
6.12.4.2 pickedUp . . . 33

6.13 teamusa::Level Class Reference . . . 33
6.13.1 Constructor & Destructor Documentation . . . 34
6.13.1.1 Level . . . 34
6.13.1.2 Level . . . 34
6.13.2 Member Function Documentation . . . 34
6.13.2.1 changeScene . . . 34
6.13.2.2 clearAll . . . 34
6.13.2.3 getActors . . . 34
6.13.2.4 getBGImageID . . . 34
6.13.2.5 getScene . . . 34
6.13.2.6 loadLevel . . . 34
6.13.2.7 parseAudioStreamActor . . . 34
6.13.2.8 parseDelayedAudioActor . . . 34
6.13.2.9 parseDelayedVideoActor . . . 34
6.13.2.10 parseInventoryItemActor . . . 34
6.13.2.11 parseLevelLink . . . 34
6.13.2.12 parseMovingActor . . . 34
6.13.2.13 parseResponsiveAudioActor . . . 34
6.13.2.14 parseResponsiveVideoActor . . . 35
6.13.2.15 parseSceneLink . . . 35
6.13.2.16 parseTextboxSpawnActor . . . 35
6.13.2.17 parseVideoActor . . . 35
6.13.2.18 parseVideoEventActor . . . 35
6.13.3 Member Data Documentation . . . 35
6.13.3.1 activeScene . . . 35
6.13.3.2 scenes . . . 35
6.13.3.3 startScene . . . 35
6.14 teamusa::LevelLink Class Reference . . . 35
6.14.1 Constructor & Destructor Documentation . . . 36
6.14.1.1 LevelLink . . . 36
6.14.1.2 ~LevelLink . . . 36
6.14.2 Member Function Documentation . . . 36
6.14.2.1 getSceneID . . . 36
6.14.2.2 getText . . . 36
6.14.2.3 onClick . . . 36
6.14.2.4 onHover . . . 36
6.14.2.5 step . . . 36
6.14.3 Member Data Documentation . . . 36
6.14.3.1 itemRequiredText . . . 36
6.14.3.2 levelID . . . 37

6.14.3.3 requiredItemID . . . 37
6.14.3.4 sceneID . . . 37
6.15 teamusa::MovingActor Class Reference . . . 37
6.15.1 Detailed Description . . . 37
6.15.2 Constructor & Destructor Documentation . . . 38
6.15.2.1 MovingActor . . . 38
6.15.2.2 ~MovingActor . . . 38
6.15.3 Member Function Documentation . . . 38
6.15.3.1 onClick . . . 38
6.15.3.2 onHover . . . 38
6.15.3.3 step . . . 38
6.15.4 Member Data Documentation . . . 38
6.15.4.1 currentStep . . . 38
6.15.4.2 endRegion . . . 38
6.15.4.3 hGrowth . . . 38
6.15.4.4 isActive . . . 38
6.15.4.5 transitionSteps . . . 38
6.15.4.6 wGrowth . . . 38
6.15.4.7 xSpeed . . . 38
6.15.4.8 ySpeed . . . 38
6.16 teamusa::Player Class Reference . . . 38
6.16.1 Detailed Description . . . 39
6.16.2 Member Typedef Documentation . . . 40
6.16.2.1 Inventory . . . 40
6.16.3 Constructor & Destructor Documentation . . . 40
6.16.3.1 Player . . . 40
6.16.3.2 ~Player . . . 40
6.16.4 Member Function Documentation . . . 40
6.16.4.1 addItem . . . 40
6.16.4.2 getCursorTextureID . . . 40
6.16.4.3 getInventory . . . 40
6.16.4.4 getPosition . . . 40
6.16.4.5 hasItem . . . 40
6.16.4.6 setCursor . . . 40
6.16.4.7 setInventory . . . 41
6.16.4.8 setPosition . . . 41
6.16.4.9 setPosition . . . 41
6.16.5 Member Data Documentation . . . 41
6.16.5.1 CURSOR_DEFAULT_ID . . . 41
6.16.5.2 CURSOR_DOWN_ID . . . 41

6.16.5.3 CURSOR_LEFT_ID . . . 41
6.16.5.4 CURSOR_RIGHT_ID . . . 41
6.16.5.5 CURSOR_SELECT_ID . . . 41
6.16.5.6 CURSOR_UP_ID . . . 41
6.16.5.7 FLASHLIGHT_ID . . . 41
6.16.5.8 mCursorStyle . . . 41
6.16.5.9 mInventory . . . 41
6.16.5.10 mLayer . . . 41
6.16.5.11 MOUSE_CLICK_ID . . . 41
6.16.5.12 mPosition . . . 41
6.16.5.13 mRegion . . . 42
6.16.5.14 mTextureID . . . 42
6.17 teamusa::Point Struct Reference . . . 42
6.17.1 Constructor & Destructor Documentation . . . 42
6.17.1.1 Point . . . 42
6.17.1.2 Point . . . 42
6.17.2 Member Data Documentation . . . 42
6.17.2.1 x . . . 42
6.17.2.2 y . . . 42
6.18 teamusa::ResponsiveAudioActor Class Reference . . . 42
6.18.1 Detailed Description . . . 43
6.18.2 Constructor & Destructor Documentation . . . 43
6.18.2.1 ResponsiveAudioActor . . . 43
6.18.2.2 ∼ResponsiveAudioActor . . . 43
6.18.3 Member Function Documentation . . . 43
6.18.3.1 onClick . . . 43
6.18.3.2 onHover . . . 43
6.18.3.3 step . . . 43
6.18.4 Member Data Documentation . . . 44
6.18.4.1 clickAudioId . . . 44
6.18.4.2 hoverAudioId . . . 44
6.19 teamusa::ResponsiveVideoActor Class Reference . . . 44
6.19.1 Constructor & Destructor Documentation . . . 44
6.19.1.1 ResponsiveVideoActor . . . 44
6.19.1.2 ∼ResponsiveVideoActor . . . 44
6.19.2 Member Function Documentation . . . 45
6.19.2.1 onClick . . . 45
6.19.2.2 onHover . . . 45
6.19.2.3 setTextureId . . . 45
6.19.2.4 step . . . 45

6.19.3 Member Data Documentation . . . 45
6.19.3.1 clickTexture . . . 45
6.19.3.2 defaultTextureId . . . 45
6.19.3.3 hoverTexture . . . 45
6.20 teamusa::Level::Scene Struct Reference . . . 45
6.20.1 Member Data Documentation . . . 45
6.20.1.1 actors . . . 45
6.20.1.2 bgImageID . . . 45
6.21 teamusa::SceneLink Class Reference . . . 46
6.21.1 Constructor & Destructor Documentation . . . 46
6.21.1.1 SceneLink . . . 46
6.21.1.2 ~SceneLink . . . 46
6.21.2 Member Function Documentation . . . 46
6.21.2.1 getText . . . 46
6.21.2.2 onClick . . . 46
6.21.2.3 onHover . . . 46
6.21.2.4 step . . . 47
6.21.3 Member Data Documentation . . . 47
6.21.3.1 cursorStyle . . . 47
6.21.3.2 itemRequiredText . . . 47
6.21.3.3 requiredItemID . . . 47
6.21.3.4 sceneID . . . 47
6.22 teamusa::TextboxSpawnActor Class Reference . . . 47
6.22.1 Detailed Description . . . 48
6.22.2 Constructor & Destructor Documentation . . . 48
6.22.2.1 TextboxSpawnActor . . . 48
6.22.2.2 ~TextboxSpawnActor . . . 48
6.22.3 Member Function Documentation . . . 48
6.22.3.1 getText . . . 48
6.22.3.2 onClick . . . 48
6.22.3.3 step . . . 48
6.22.4 Member Data Documentation . . . 49
6.22.4.1 activated . . . 49
6.22.4.2 text . . . 49
6.23 teamusa::Timer Class Reference . . . 49
6.23.1 Detailed Description . . . 49
6.23.2 Constructor & Destructor Documentation . . . 49
6.23.2.1 Timer . . . 49
6.23.2.2 ~Timer . . . 49
6.23.3 Member Function Documentation . . . 49

6.23.3.1 getTicks . . . 50
6.23.3.2 pause . . . 50
6.23.3.3 start . . . 50
6.23.3.4 stop . . . 50
6.23.3.5 unpause . . . 50
6.23.4 Member Data Documentation . . . 50
6.23.4.1 mPaused . . . 50
6.23.4.2 mPauseTicks . . . 50
6.23.4.3 mStarted . . . 50
6.23.4.4 mStartTicks . . . 50
6.24 teamusa::VideoActor Class Reference . . . 50
6.24.1 Constructor & Destructor Documentation . . . 51
6.24.1.1 VideoActor . . . 51
6.24.1.2 ∼VideoActor . . . 51
6.24.2 Member Function Documentation . . . 51
6.24.2.1 step . . . 51
6.25 mediawrap::VideoContext Class Reference . . . 51
6.25.1 Detailed Description . . . 52
6.25.2 Member Typedef Documentation . . . 53
6.25.2.1 Region . . . 53
6.25.2.2 texture_iter . . . 53
6.25.2.3 TextureID . . . 53
6.25.3 Member Enumeration Documentation . . . 53
6.25.3.1 BlendMode . . . 53
6.25.3.2 DebugColor . . . 53
6.25.3.3 Flip . . . 53
6.25.4 Constructor & Destructor Documentation . . . 54
6.25.4.1 VideoContext . . . 54
6.25.4.2 ∼VideoContext . . . 55
6.25.5 Member Function Documentation . . . 55
6.25.5.1 create_texture . . . 55
6.25.5.2 delete_texture . . . 55
6.25.5.3 display . . . 55
6.25.5.4 fill_texture . . . 55
6.25.5.5 load_font . . . 56
6.25.5.6 load_texture . . . 56
6.25.5.7 render . . . 56
6.25.5.8 render_clear . . . 56
6.25.5.9 render_clear . . . 56
6.25.5.10 render_onto . . . 57

6.25.5.11 render_rotate . . . 57
6.25.5.12 render_text . . . 57
6.25.5.13 renderDebugBox . . . 57
6.25.6 Member Data Documentation . . . 57
6.25.6.1 font . . . 57
6.25.6.2 renderer . . . 57
6.25.6.3 textures . . . 58
6.25.6.4 video_display . . . 58
6.26 mediawrap::VideoDisplay Class Reference . . . 58
6.26.1 Detailed Description . . . 58
6.26.2 Constructor & Destructor Documentation . . . 58
6.26.2.1 VideoDisplay . . . 58
6.26.2.2 ∼VideoDisplay . . . 58
6.26.3 Member Function Documentation . . . 59
6.26.3.1 get_renderer . . . 59
6.26.4 Member Data Documentation . . . 59
6.26.4.1 window . . . 59
6.27 teamusa::VideoEngine Class Reference . . . 59
6.27.1 Detailed Description . . . 60
6.27.2 Constructor & Destructor Documentation . . . 60
6.27.2.1 VideoEngine . . . 60
6.27.2.2 ∼VideoEngine . . . 60
6.27.3 Member Function Documentation . . . 60
6.27.3.1 clearLayers . . . 60
6.27.3.2 deleteResourceGroup . . . 61
6.27.3.3 deleteTexture . . . 62
6.27.3.4 display . . . 62
6.27.3.5 hideTextbox . . . 62
6.27.3.6 isShowingTextbox . . . 62
6.27.3.7 loadTexture . . . 62
6.27.3.8 render . . . 62
6.27.3.9 renderDebugBox . . . 62
6.27.3.10 renderRotate . . . 63
6.27.3.11 showTextbox . . . 64
6.27.4 Member Data Documentation . . . 64
6.27.4.1 coreResources . . . 64
6.27.4.2 layers . . . 64
6.27.4.3 levelResources . . . 64
6.27.4.4 MAX_RESERVED_ID . . . 64
6.27.4.5 NUM_LAYERS . . . 64

6.27.4.6 SHADOW_LAYER . . . 64
6.27.4.7 TEXT_LAYER . . . 64
6.27.4.8 textboxActive . . . 64
6.27.4.9 textboxPadding . . . 64
6.27.4.10 textboxRegion . . . 64
6.27.4.11 videoContext . . . 64
6.28 teamusa::VideoEventActor Class Reference . . . 64
6.28.1 Detailed Description . . . 65
6.28.2 Constructor & Destructor Documentation . . . 65
6.28.2.1 VideoEventActor . . . 65
6.28.2.2 ~VideoEventActor . . . 65
6.28.3 Member Function Documentation . . . 65
6.28.3.1 onClick . . . 65
6.28.3.2 onHover . . . 65
6.28.3.3 step . . . 66
6.28.4 Member Data Documentation . . . 66
6.28.4.1 actorEvent . . . 66

7 File Documentation **67**
7.1 ActorEvent.h File Reference . . . 67
7.1.1 Detailed Description . . . 67
7.2 Assert.h File Reference . . . 67
7.2.1 Detailed Description . . . 68
7.2.2 Macro Definition Documentation . . . 68
7.2.2.1 Assert . . . 68
7.3 AudioEngine.cpp File Reference . . . 68
7.4 AudioEngine.hpp File Reference . . . 68
7.5 AudioPlayer.cpp File Reference . . . 68
7.6 AudioPlayer.hpp File Reference . . . 69
7.7 AudioStreamActor.cpp File Reference . . . 69
7.7.1 Detailed Description . . . 69
7.8 AudioStreamActor.h File Reference . . . 69
7.8.1 Detailed Description . . . 69
7.9 BaseActor.cpp File Reference . . . 70
7.10 BaseActor.h File Reference . . . 70
7.10.1 Detailed Description . . . 70
7.11 CursorStyle.h File Reference . . . 70
7.11.1 Detailed Description . . . 70
7.12 DelayedAudioActor.cpp File Reference . . . 71
7.12.1 Detailed Description . . . 71

7.13 DelayedAudioActor.h File Reference . . . 71
7.13.1 Detailed Description . . . 71
7.14 DelayedVideoActor.cpp File Reference . . . 71
7.14.1 Detailed Description . . . 71
7.15 DelayedVideoActor.h File Reference . . . 71
7.15.1 Detailed Description . . . 72
7.16 Engine.cpp File Reference . . . 72
7.16.1 Detailed Description . . . 72
7.16.2 Macro Definition Documentation . . . 72
7.16.2.1 BIND . . . 72
7.16.3 Variable Documentation . . . 72
7.16.3.1 FRAME_TIME . . . 72
7.17 Engine.h File Reference . . . 72
7.17.1 Detailed Description . . . 73
7.18 ExampleActor.cpp File Reference . . . 73
7.18.1 Detailed Description . . . 73
7.19 ExampleActor.h File Reference . . . 73
7.19.1 Detailed Description . . . 73
7.20 GameSaveSerializer.cpp File Reference . . . 74
7.21 GameSaveSerializer.h File Reference . . . 74
7.21.1 Detailed Description . . . 74
7.22 Headers.h File Reference . . . 74
7.22.1 Detailed Description . . . 75
7.23 InventoryItemActor.cpp File Reference . . . 75
7.23.1 Detailed Description . . . 75
7.24 InventoryItemActor.h File Reference . . . 75
7.24.1 Detailed Description . . . 75
7.25 Level.cpp File Reference . . . 75
7.25.1 Detailed Description . . . 76
7.25.2 Function Documentation . . . 76
7.25.2.1 loadError . . . 76
7.25.2.2 operator>> . . . 76
7.25.2.3 operator>> . . . 76
7.26 Level.h File Reference . . . 76
7.26.1 Detailed Description . . . 77
7.27 LevelLink.cpp File Reference . . . 77
7.27.1 Detailed Description . . . 77
7.28 LevelLink.h File Reference . . . 77
7.28.1 Detailed Description . . . 77
7.29 main.cpp File Reference . . . 77

7.29.1 Detailed Description . . . 78
7.29.2 Function Documentation . . . 78
7.29.2.1 main . . . 78
7.30 MovingActor.cpp File Reference . . . 78
7.31 MovingActor.h File Reference . . . 78
7.32 Player.cpp File Reference . . . 78
7.32.1 Detailed Description . . . 78
7.33 Player.h File Reference . . . 79
7.33.1 Detailed Description . . . 79
7.34 Point.h File Reference . . . 79
7.34.1 Detailed Description . . . 79
7.35 ResourceGroup.hpp File Reference . . . 79
7.35.1 Enumeration Type Documentation . . . 80
7.35.1.1 ResourceGroup . . . 80
7.36 ResponsiveAudioActor.cpp File Reference . . . 80
7.36.1 Detailed Description . . . 80
7.37 ResponsiveAudioActor.h File Reference . . . 80
7.37.1 Detailed Description . . . 80
7.38 ResponsiveVideoActor.cpp File Reference . . . 80
7.38.1 Detailed Description . . . 80
7.39 ResponsiveVideoActor.h File Reference . . . 81
7.39.1 Detailed Description . . . 81
7.40 SceneLink.cpp File Reference . . . 81
7.40.1 Detailed Description . . . 81
7.41 SceneLink.h File Reference . . . 81
7.41.1 Detailed Description . . . 81
7.42 TextboxSpawnActor.cpp File Reference . . . 82
7.42.1 Detailed Description . . . 82
7.43 TextboxSpawnActor.h File Reference . . . 82
7.43.1 Detailed Description . . . 82
7.44 Timer.cpp File Reference . . . 82
7.44.1 Detailed Description . . . 82
7.45 Timer.h File Reference . . . 82
7.45.1 Detailed Description . . . 83
7.46 VideoActor.cpp File Reference . . . 83
7.46.1 Detailed Description . . . 83
7.47 VideoActor.h File Reference . . . 83
7.47.1 Detailed Description . . . 83
7.48 VideoContext.cpp File Reference . . . 83
7.49 VideoContext.hpp File Reference . . . 84

7.50 VideoDisplay.cpp File Reference 84
7.51 VideoDisplay.hpp File Reference 84
7.52 VideoEngine.cpp File Reference 84
7.53 VideoEngine.hpp File Reference 85
7.54 VideoEventActor.cpp File Reference 85
7.54.1 Detailed Description 85
7.55 VideoEventActor.h File Reference 85
7.55.1 Detailed Description 86

Index **87**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

MainNS 9
mediawrap 9
teamusa 9

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

teamusa::ActorEvent . . . 13
teamusa::ActorVideo . . . 13
teamusa::AudioEngine . . . 14
mediawrap::AudioPlayer . . . 16
teamusa::BaseActor . . . 20
- teamusa::AudioStreamActor . . . 18
- teamusa::DelayedAudioActor . . . 24
- teamusa::DelayedVideoActor . . . 25
- teamusa::ExampleActor . . . 29
- teamusa::InventoryItemActor . . . 32
- teamusa::LevelLink . . . 35
- teamusa::MovingActor . . . 37
- teamusa::ResponsiveAudioActor . . . 42
- teamusa::ResponsiveVideoActor . . . 44
- teamusa::SceneLink . . . 46
- teamusa::TextboxSpawnActor . . . 47
- teamusa::VideoActor . . . 50
- teamusa::VideoEventActor . . . 64

teamusa::Engine . . . 27
teamusa::GameSaveSerializer . . . 30
teamusa::Level . . . 33
teamusa::Player . . . 38
teamusa::Point . . . 42
teamusa::Level::Scene . . . 45
teamusa::Timer . . . 49
mediawrap::VideoContext . . . 51
mediawrap::VideoDisplay . . . 58
teamusa::VideoEngine . . . 59

Chapter 3

Class Index

3.1 Class List

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

teamusa::ActorEvent
: Event data generated by Actors, handled by Engine . . . 13

teamusa::ActorVideo . . . 13

teamusa::AudioEngine
: Provides project-specific functionality for Legend of the Great Unwashed . . . 14

mediawrap::AudioPlayer
: Provides basic audio playing capabilities with WAV files . . . 16

teamusa::AudioStreamActor
: If this actor is not activated, it will emit a StreamAudio event and set its status to activated when the step method is called . . . 18

teamusa::BaseActor
: Abstract class which all actors must derive from . . . 20

teamusa::DelayedAudioActor
: :Will increment a counter every time the step method is called . . . 24

teamusa::DelayedVideoActor
: Will increment a counter every time the step method is called . . . 25

teamusa::Engine
: Processes all components of the game each frame . . . 27

teamusa::ExampleActor
: <Give brief="" description="" here>=""> . . . 29

teamusa::GameSaveSerializer
: Provides multithreaded save, single-thread load of save files . . . 30

teamusa::InventoryItemActor
: <Give brief="" description="" here>=""> . . . 32

teamusa::Level . . . 33

teamusa::LevelLink . . . 35

teamusa::MovingActor
: Will transition from one region to the next by calculating the distance to move each frame for a set number of frames . . . 37

teamusa::Player
: Handles all data relevant to the player engaging the game . . . 38

teamusa::Point . . . 42

teamusa::ResponsiveAudioActor
: \ Brief: Will increment the value of stepCount until it is equal to durationSteps for each call to the step method . . . 42

teamusa::ResponsiveVideoActor . . . 44

teamusa::Level::Scene . . . 45

teamusa::SceneLink . . . 46

teamusa::TextboxSpawnActor
<Give brief="" description="" here>=""> 47
teamusa::Timer
A timer that counts up from zero in milliseconds 49
teamusa::VideoActor 50
mediawrap::VideoContext
Provides basic 2D rendering capabilities 51
mediawrap::VideoDisplay
Creates a window and initializes SDL2 and SDL2_IMG 58
teamusa::VideoEngine
Provides video capabilities that are specific to Legend of the Great Unwashed 59
teamusa::VideoEventActor
Will display a texture and perform no action until clicked 64

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

ActorEvent.h
    Declares ActorEvent struct . . . 67
Assert.h
    Declares custom Assert macro . . . 67
AudioEngine.cpp . . . 68
AudioEngine.hpp . . . 68
AudioPlayer.cpp . . . 68
AudioPlayer.hpp . . . 69
AudioStreamActor.cpp
    Implements AudioStreamActor class . . . 69
AudioStreamActor.h
    Declares AudioStreamActor class . . . 69
BaseActor.cpp . . . 70
BaseActor.h
    Implements BaseActor class . . . 70
CursorStyle.h
    Declares CursorStyle enumerations . . . 70
DelayedAudioActor.cpp
    Implements DelayedAudioActor class . . . 71
DelayedAudioActor.h
    Declares DelayedAudioActor class . . . 71
DelayedVideoActor.cpp
    Declares DelayedVideoActor class . . . 71
DelayedVideoActor.h
    Declares DelayedVideoActor class . . . 71
Engine.cpp
    Implements Engine class . . . 72
Engine.h
    Declares Engine class . . . 72
ExampleActor.cpp
    Implements ExampleActor class . . . 73
ExampleActor.h
    Declares ExampleActor class . . . 73
GameSaveSerializer.cpp . . . 74
GameSaveSerializer.h
    Declares save file serializer class . . . 74
Headers.h
    Easy way to include all headers needed . . . 74

InventoryItemActor.cpp
Implements InventoryItemActor class . . . 75
InventoryItemActor.h
Declares InventoryItemActor class . . . 75
Level.cpp
Implements Level class . . . 75
Level.h
Declares Level class . . . 76
LevelLink.cpp
Implements LevelLink class . . . 77
LevelLink.h
Declares LevelLink class . . . 77
main.cpp
Entry point of program . . . 77
MovingActor.cpp . . . 78
MovingActor.h . . . 78
Player.cpp
Implements Player class . . . 78
Player.h
Declares Player class . . . 79
Point.h
Declares Point struct . . . 79
ResourceGroup.hpp . . . 79
ResponsiveAudioActor.cpp
Implements ResponsiveAudioActor class . . . 80
ResponsiveAudioActor.h
Declares ResponsiveAudioActor class . . . 80
ResponsiveVideoActor.cpp
Will display the default TextureID . . . 80
ResponsiveVideoActor.h
Declares ResponsivevideoActor class . . . 81
SceneLink.cpp
Implements SceneLink class . . . 81
SceneLink.h
Declares SceneLink class . . . 81
TextboxSpawnActor.cpp
Declares TextboxSpawnActor class . . . 82
TextboxSpawnActor.h
Declares TextboxSpawnActor class . . . 82
Timer.cpp
Implements Timer class . . . 82
Timer.h
Declares Timer class . . . 82
VideoActor.cpp
Implements VideoActor class . . . 83
VideoActor.h
This module makes sure An actor that will only display a texture at a given region . . . 83
VideoContext.cpp . . . 83
VideoContext.hpp . . . 84
VideoDisplay.cpp . . . 84
VideoDisplay.hpp . . . 84
VideoEngine.cpp . . . 84
VideoEngine.hpp . . . 85
VideoEventActor.cpp
Declares VideoEventActor class . . . 85
VideoEventActor.h
Declares VideoEventActor class . . . 85

Chapter 5

Namespace Documentation

5.1 MainNS Namespace Reference

Functions

- static void logError (const std::string &desc)

5.1.1 Function Documentation

5.1.1.1 static void MainNS::logError (const std::string & *desc*) [static]

5.2 mediawrap Namespace Reference

Classes

- class AudioPlayer

  Provides basic audio playing capabilities with WAV files.
- class VideoContext

  Provides basic 2D rendering capabilities.
- class VideoDisplay

  Creates a window and initializes SDL2 and SDL2_IMG.

5.3 teamusa Namespace Reference

Classes

- class ActorEvent

  Event data generated by Actors, handled by Engine.
- struct ActorVideo
- class AudioEngine

  Provides project-specific functionality for Legend of the Great Unwashed.
- class AudioStreamActor

  If this actor is not activated, it will emit a StreamAudio event and set its status to activated when the step method is called.
- class BaseActor

  Abstract class which all actors must derive from.

- class DelayedAudioActor

  :Will increment a counter every time the step method is called.
- class DelayedVideoActor

  Will increment a counter every time the step method is called.
- class Engine

  Processes all components of the game each frame.
- class ExampleActor

  <Give brief="" description="" here>=""
- class GameSaveSerializer

  Provides multithreaded save, single-thread load of save files.
- class InventoryItemActor

  <Give brief="" description="" here>=""
- class Level
- class LevelLink
- class MovingActor

  Will transition from one region to the next by calculating the distance to move each frame for a set number of frames.
- class Player

  Handles all data relevant to the player engaging the game.
- struct Point
- class ResponsiveAudioActor

  \ Brief: Will increment the value of stepCount until it is equal to durationSteps for each call to the step method.
- class ResponsiveVideoActor
- class SceneLink
- class TextboxSpawnActor

  <Give brief="" description="" here>=""
- class Timer

  A timer that counts up from zero in milliseconds.
- class VideoActor
- class VideoEngine

  Provides video capabilities that are specific to Legend of the Great Unwashed.
- class VideoEventActor

  Will display a texture and perform no action until clicked.

Typedefs

- typedef mediawrap::AudioPlayer::AudioID AudioID
- typedef std::shared_ptr< BaseActor > BaseActorPtr
- typedef std::vector< BaseActorPtr > ActorList
- typedef mediawrap::VideoContext::TextureID TextureID
- typedef mediawrap::VideoContext::Region Region

Enumerations

- enum ActorEventType {
  Nil = -1, ChangeScene, LoadLevel, PlayAudio,
  NewGame, LoadGame, DisplayText, ExitGame,
  StreamAudio }
- enum CursorStyle {
  CursorStyle::CURSOR_DEFAULT, CursorStyle::CURSOR_SELECT, CursorStyle::CURSOR_LEF↩
  T, CursorStyle::CURSOR_RIGHT,
  CursorStyle::CURSOR_UP, CursorStyle::CURSOR_DOWN }

  The possible styles for the mouse cursor.

5.3.1 Typedef Documentation

5.3.1.1 typedef std::vector<BaseActorPtr> teamusa::ActorList

5.3.1.2 typedef mediawrap::AudioPlayer::AudioID teamusa::AudioID

5.3.1.3 typedef std::shared_ptr<BaseActor> teamusa::BaseActorPtr

5.3.1.4 typedef mediawrap::VideoContext::Region teamusa::Region

5.3.1.5 typedef mediawrap::VideoContext::TextureID teamusa::TextureID

5.3.2 Enumeration Type Documentation

5.3.2.1 enum teamusa::ActorEventType

Enumerator

Nil

ChangeScene

LoadLevel

PlayAudio

NewGame

LoadGame

DisplayText

ExitGame

StreamAudio

5.3.2.2 enum teamusa::CursorStyle [strong]

The possible styles for the mouse cursor.

Enumerator

CURSOR_DEFAULT Default cursor.

CURSOR_SELECT Offers the ability to select an object.

CURSOR_LEFT Points left.

CURSOR_RIGHT Points right.

CURSOR_UP Points up.

CURSOR_DOWN Points down.

Chapter 6

Class Documentation

6.1 teamusa::ActorEvent Class Reference

Event data generated by Actors, handled by Engine.

`#include <ActorEvent.h>`

Public Member Functions

- ActorEvent (void)

Public Attributes

- int32_t value
- ActorEventType type

6.1.1 Detailed Description

Event data generated by Actors, handled by Engine.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 teamusa::ActorEvent::ActorEvent (void) [inline]

6.1.3 Member Data Documentation

6.1.3.1 ActorEventType teamusa::ActorEvent::type

6.1.3.2 int32_t teamusa::ActorEvent::value

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

- ActorEvent.h

6.2 teamusa::ActorVideo Struct Reference

`#include <BaseActor.h>`

Public Member Functions

- ActorVideo (void)

Public Attributes

- int32_t layer
- int32_t textureID

6.2.1 Constructor & Destructor Documentation

6.2.1.1 teamusa::ActorVideo::ActorVideo (void) [inline]

6.2.2 Member Data Documentation

6.2.2.1 int32_t teamusa::ActorVideo::layer

6.2.2.2 int32_t teamusa::ActorVideo::textureID

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

- BaseActor.h

6.3 teamusa::AudioEngine Class Reference

Provides project-specific functionality for Legend of the Great Unwashed.

`#include <AudioEngine.hpp>`

Public Member Functions

- void loadSound (const std::string &path, AudioID id, ResourceGroup group)

  Loads the given sound file and associates it with the given id.
- void playSound (AudioID id)

  Plays the sound associated with the given id.
- void playStream (const std::string &path)

  Plays the given stream in a loop continuously.
- void deleteSound (AudioID id)

  Deletes the given sound from memory.
- void deleteSoundGroup (ResourceGroup resourceGroup)

  Deletes the entire group of sounds.

Private Attributes

- std::vector< AudioID > coreResources
- std::vector< AudioID > levelResources
- AudioPlayer audioPlayer

Static Private Attributes

- static const AudioID MAX_RESERVED_ID = 1000

6.3.1 Detailed Description

Provides project-specific functionality for Legend of the Great Unwashed.

6.3.2 Member Function Documentation

6.3.2.1 void teamusa::AudioEngine::deleteSound (AudioID *id*)

Deletes the given sound from memory.

Parameters

id	The id of the audio to delete.

6.3.2.2 void teamusa::AudioEngine::deleteSoundGroup (ResourceGroup *resourceGroup*)

Deletes the entire group of sounds.

Parameters

6.3.2.3 void teamusa::AudioEngine::loadSound (const std::string & *path,* AudioID *id,* ResourceGroup *group*)

Loads the given sound file and associates it with the given id.

Parameters

path	The relative path of the sound file to load.
id	The id to associate with the given sound file.

6.3.2.4 void teamusa::AudioEngine::playSound (AudioID *id*)

Plays the sound associated with the given id.

Parameters

id	The id of the sound to play.

6.3.2.5 void teamusa::AudioEngine::playStream (const std::string & *path*)

Plays the given stream in a loop continuously.

Parameters

path	The path of the audio to stream.

6.3.3 Member Data Documentation

6.3.3.1 AudioPlayer teamusa::AudioEngine::audioPlayer [private]

6.3.3.2 std::vector<AudioID> teamusa::AudioEngine::coreResources [private]

6.3.3.3 std::vector<AudioID> teamusa::AudioEngine::levelResources [private]

6.3.3.4 const AudioID teamusa::AudioEngine::MAX_RESERVED_ID = 1000 [static], [private]

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

- AudioEngine.hpp
- AudioEngine.cpp

6.4 mediawrap::AudioPlayer Class Reference

Provides basic audio playing capabilities with WAV files.

`#include <AudioPlayer.hpp>`

Public Types

- typedef unsigned int AudioID

  Used to uniquely identify each audio sample.

Public Member Functions

- AudioPlayer ()

  Constructs a new audio player.
- ~AudioPlayer ()

  Deletes the audio player and all of its samples and streams.
- void load_stream (const std::string &file_path)

  Loads the given audio file and prepares it for streaming.
- void stream_audio (int loops=-1)

  Plays the loaded audio stream loop+1 times.
- void load_sample (AudioID id, const std::string &file_path)

  Loads the given audio sample into memory.
- void play_sample (AudioID id)

  Plays the given audio sample in the first available channel.
- void delete_sample (AudioID id)

  Deletes the sample created by a call to load_sample().
- void clear_samples ()

  Deletes all samples created by a call to load_sample().

Private Attributes

- std::unordered_map< AudioID, Mix_Chunk * > * audio_samples
- Mix_Music * audio_stream

Static Private Attributes

- static const int audio_rate = 44100
- static const int audio_channels = 1
- static const int audio_buffer = 4096
- static const Uint16 audio_format = AUDIO_S16

6.4.1 Detailed Description

Provides basic audio playing capabilities with WAV files.

Acts as an abstraction layer for SDL2.

6.4.2 Member Typedef Documentation

6.4.2.1 typedef unsigned int mediawrap::AudioPlayer::AudioID

Used to uniquely identify each audio sample.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 mediawrap::AudioPlayer::AudioPlayer ( )

Constructs a new audio player.

Enables SDL audio functionality.

6.4.3.2 mediawrap::AudioPlayer::~AudioPlayer ( )

Deletes the audio player and all of its samples and streams.

Disables SDL audio functionality.

6.4.4 Member Function Documentation

6.4.4.1 void mediawrap::AudioPlayer::clear_samples ( )

Deletes all samples created by a call to load_sample().

6.4.4.2 void mediawrap::AudioPlayer::delete_sample (AudioID *id*)

Deletes the sample created by a call to load_sample().

Parameters

id	The id of the sample to delete.

6.4.4.3 void mediawrap::AudioPlayer::load_sample (AudioID *id*, const std::string & *file_path*)

Loads the given audio sample into memory.

Loading a sample into an existing id will delete the sample associated with it before the new sample is loaded.

Parameters

id	The unique id to store the sample under.
file_path	The path of the audio file to load into memory.

6.4.4.4 void mediawrap::AudioPlayer::load_stream (const std::string & *file_path*)

Loads the given audio file and prepares it for streaming.

Only one audio stream can be loaded at a time. The previously loaded stream will be deleted if this method is called multiple times.

Parameters

file_path	The path of the file to load.

6.4.4.5 void mediawrap::AudioPlayer::play_sample (AudioID *id*)

Plays the given audio sample in the first available channel.

Parameters

id	The id of the audio sample to play.

6.4.4.6 void mediawrap::AudioPlayer::stream_audio (int *loops* = -1)

Plays the loaded audio stream loop+1 times.

If set to -1, the audio will loop indefinitely. Only one audio stream can be played at a time.

Parameters

loops	The number of times to play the audio. A value of -1 is infinite. Defaults to looping infinitely.

6.4.5 Member Data Documentation

6.4.5.1 const int mediawrap::AudioPlayer::audio_buffer = 4096 [static], [private]

6.4.5.2 const int mediawrap::AudioPlayer::audio_channels = 1 [static], [private]

6.4.5.3 const Uint16 mediawrap::AudioPlayer::audio_format = AUDIO_S16 [static], [private]

6.4.5.4 const int mediawrap::AudioPlayer::audio_rate = 44100 [static], [private]

6.4.5.5 std::unordered_map<AudioID, Mix_Chunk*>* mediawrap::AudioPlayer::audio_samples [private]

6.4.5.6 Mix_Music* mediawrap::AudioPlayer::audio_stream [private]

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

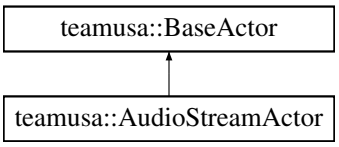
- AudioPlayer.hpp
- AudioPlayer.cpp

6.5 teamusa::AudioStreamActor Class Reference

If this actor is not activated, it will emit a StreamAudio event and set its status to activated when the step method is called.

`#include <AudioStreamActor.h>`

Inheritance diagram for teamusa::AudioStreamActor:



Public Member Functions

- AudioStreamActor (std::string path)
- virtual ~AudioStreamActor (void) override
- virtual const ActorEvent step (Player &player) override

  <Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>=""
- std::string getPath ()

Private Attributes

- std::string path
- bool activated

Additional Inherited Members

6.5.1 Detailed Description

If this actor is not activated, it will emit a StreamAudio event and set its status to activated when the step method is called.

The engine can then retrieve the path to the audio file by a call to this actor's getPath method.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 AudioStreamActor::AudioStreamActor (std::string *path*) [explicit]

6.5.2.2 AudioStreamActor::~AudioStreamActor (void) [override], [virtual]

6.5.3 Member Function Documentation

6.5.3.1 std::string AudioStreamActor::getPath ( )

6.5.3.2 const ActorEvent AudioStreamActor::step (Player & *player*) [override], [virtual]

<Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>=""

Implements teamusa::BaseActor.

6.5.4 Member Data Documentation

6.5.4.1 bool teamusa::AudioStreamActor::activated [private]

6.5.4.2 std::string teamusa::AudioStreamActor::path [private]

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

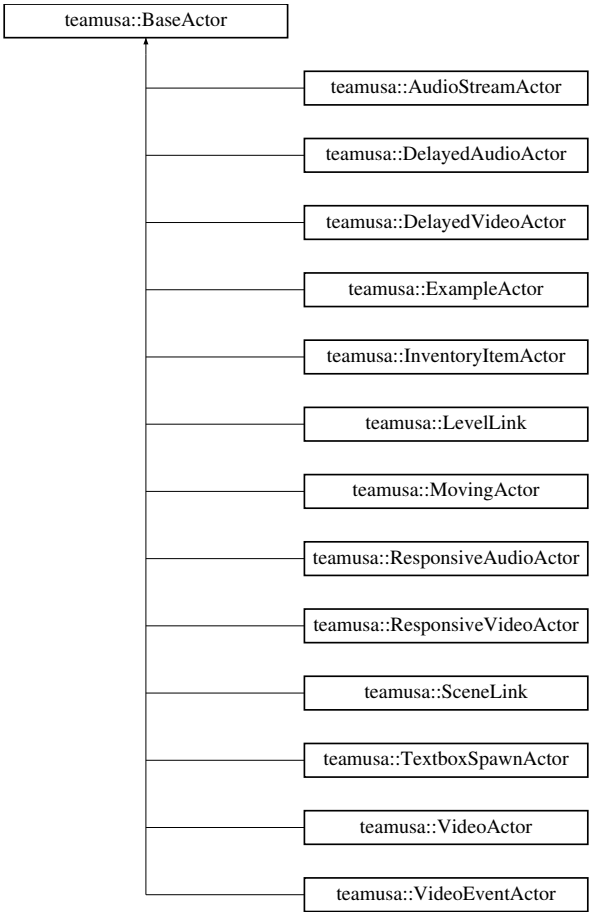
- AudioStreamActor.h
- AudioStreamActor.cpp

6.6 teamusa::BaseActor Class Reference

Abstract class which all actors must derive from.

`#include <BaseActor.h>`

Inheritance diagram for teamusa::BaseActor:



Public Member Functions

- BaseActor (const Region &region=Region())
- virtual ~BaseActor (void)=0
- virtual const ActorEvent onClick (Player &player)

  Called when the actor is clicked on.
- virtual const ActorEvent onHover (Player &player)

  Called when the actor is hovered over with the mouse.
- virtual const ActorEvent step (Player &player)=0

  Called each frame, each derived actor should handle this.
- virtual const bool isInBounds (const Point &point)

  Calculates if point is in bounds of actor's region.
- virtual void setRegion (const Region &region)

  Sets the actor's region (can be used by Level when loading).
- virtual const Region getRegion (void) const

  Gets the actor's Region.
- virtual const int32_t getLayer (void) const

  Gets the layer the actor should be rendered on.

- virtual const int32_t getTextureID (void) const

  Gets the texture ID of the actor.
- const bool hasVideo (void) const

  Returns true if the actor has a video component.

Protected Attributes

- Region mRegion
- AudioID mAudioID
- ActorVideo * mVideo

6.6.1 Detailed Description

Abstract class which all actors must derive from.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 BaseActor::BaseActor (const Region & *region* = Region ()) [explicit]

6.6.2.2 BaseActor::~BaseActor (void) [pure virtual]

6.6.3 Member Function Documentation

6.6.3.1 const int32_t BaseActor::getLayer (void) const [virtual]

Gets the layer the actor should be rendered on.

Returns

An integer containing the layer.

6.6.3.2 const Region BaseActor::getRegion (void) const [virtual]

Gets the actor's Region.

Returns

The actor's Region struct.

6.6.3.3 const int32_t BaseActor::getTextureID (void) const [virtual]

Gets the texture ID of the actor.

Returns

The integer containing the texture ID.

6.6.3.4 const bool BaseActor::hasVideo (void) const

Returns true if the actor has a video component.

6.6.3.5 const bool BaseActor::isInBounds (const Point & *point*) [virtual]

Calculates if point is in bounds of actor's region.

Parameters

point	The point to test.

Returns

True if point is within actor's region.

6.6.3.6 const ActorEvent BaseActor::onClick (Player & *player*) [virtual]

Called when the actor is clicked on.

Parameters

player	The player in the scene.

Returns

The ActorEvent to be handled by Engine when clicked on.

Reimplemented in teamusa::MovingActor, teamusa::TextboxSpawnActor, teamusa::ResponsiveAudioActor, teamusa::VideoEventActor, teamusa::ExampleActor, teamusa::InventoryItemActor, teamusa::LevelLink, teamusa::SceneLink, and teamusa::ResponsiveVideoActor.

6.6.3.7 const ActorEvent BaseActor::onHover (Player & *player*) [virtual]

Called when the actor is hovered over with the mouse.

Parameters

player	The player in the scene.

Returns

The ActorEvent to be handled by Engine when hovered over.

Reimplemented in teamusa::MovingActor, teamusa::ResponsiveAudioActor, teamusa::VideoEventActor, teamusa::ExampleActor, teamusa::LevelLink, teamusa::SceneLink, teamusa::ResponsiveVideoActor, and teamusa::InventoryItemActor.

6.6.3.8 void BaseActor::setRegion (const Region & *region*) [virtual]

Sets the actor's region (can be used by Level when loading).

Parameters

region	The Region to set.

6.6.3.9 virtual const ActorEvent teamusa::BaseActor::step (Player & *player*) [pure virtual]

Called each frame, each derived actor should handle this.

Parameters

player	The player in the scene.

Returns

Any ActorEvent that should be handled immediately by Engine.

Implemented in teamusa::MovingActor, teamusa::ResponsiveAudioActor, teamusa::VideoEventActor, teamusa::TextboxSpawnActor, teamusa::DelayedVideoActor, teamusa::ExampleActor, teamusa::DelayedAudioActor, teamusa::LevelLink, teamusa::ResponsiveVideoActor, teamusa::SceneLink, teamusa::InventoryItemActor, teamusa::AudioStreamActor, and teamusa::VideoActor.

6.6.4 Member Data Documentation

6.6.4.1 AudioID teamusa::BaseActor::mAudioID [protected]

6.6.4.2 Region teamusa::BaseActor::mRegion [protected]

6.6.4.3 ActorVideo* teamusa::BaseActor::mVideo [protected]

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

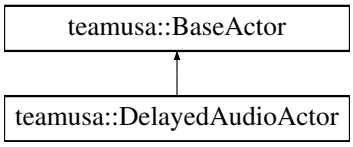
- BaseActor.h
- BaseActor.cpp

6.7 teamusa::DelayedAudioActor Class Reference

:Will increment a counter every time the step method is called.

`#include <DelayedAudioActor.h>`

Inheritance diagram for teamusa::DelayedAudioActor:



Public Member Functions

- DelayedAudioActor (int audioID, int delaySteps)
- virtual ~DelayedAudioActor (void) override
- virtual const ActorEvent step (Player &player) override

  <Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>=""

Private Attributes

- int audioId
- int delaySteps
- int currentStep

Additional Inherited Members

6.7.1 Detailed Description

:Will increment a counter every time the step method is called.

After a specified number of steps have occurred, this actor will change its TextureID to a valid value and will be displayed. When the number of steps is equal to the disappearing step, the TextureID will be set to an ignored value, causing the actor to disappear.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 DelayedAudioActor::DelayedAudioActor (int *audioID,* int *delaySteps* = 0) [explicit]

6.7.2.2 DelayedAudioActor::~DelayedAudioActor (void) [override], [virtual]

6.7.3 Member Function Documentation

6.7.3.1 const ActorEvent DelayedAudioActor::step (Player & *player*) [override], [virtual]

<Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>="">

Implements teamusa::BaseActor.

6.7.4 Member Data Documentation

6.7.4.1 int teamusa::DelayedAudioActor::audioId [private]

6.7.4.2 int teamusa::DelayedAudioActor::currentStep [private]

6.7.4.3 int teamusa::DelayedAudioActor::delaySteps [private]

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

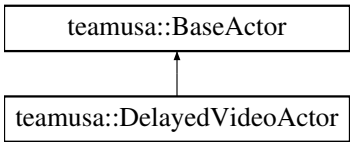
- DelayedAudioActor.h
- DelayedAudioActor.cpp

6.8 teamusa::DelayedVideoActor Class Reference

Will increment a counter every time the step method is called.

`#include <DelayedVideoActor.h>`

Inheritance diagram for teamusa::DelayedVideoActor:



Public Member Functions

- DelayedVideoActor (Region region, int textureID, int delaysteps, int disappearStep, int layer)

- virtual ~DelayedVideoActor (void) override
- virtual const ActorEvent step (Player &player) override

  increment a step counter

Private Attributes

- int textureId
- int delaySteps
- int currentStep
- int disappear

Additional Inherited Members

6.8.1 Detailed Description

Will increment a counter every time the step method is called.

After a specified number of steps have occurred, this actor will change its TextureID to a valid value and will be displayed. When the number of steps is equal to the disappearing step, the TextureID will be set to an ignored value, causing the actor to disappear

6.8.2 Constructor & Destructor Documentation

6.8.2.1 DelayedVideoActor::DelayedVideoActor (Region *region,* int *textureID,* int *delaysteps,* int *disappearStep,* int *layer*) [explicit]

6.8.2.2 DelayedVideoActor::~DelayedVideoActor (void) [override], [virtual]

6.8.3 Member Function Documentation

6.8.3.1 const ActorEvent DelayedVideoActor::step (Player & *player*) [override], [virtual]

increment a step counter

Implements teamusa::BaseActor.

6.8.4 Member Data Documentation

6.8.4.1 int teamusa::DelayedVideoActor::currentStep [private]

6.8.4.2 int teamusa::DelayedVideoActor::delaySteps [private]

6.8.4.3 int teamusa::DelayedVideoActor::disappear [private]

6.8.4.4 int teamusa::DelayedVideoActor::textureId [private]

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

- DelayedVideoActor.h
- DelayedVideoActor.cpp

6.9 teamusa::Engine Class Reference

Processes all components of the game each frame.

`#include <Engine.h>`

Public Member Functions

- Engine (void)
- ~Engine (void)
- void run (void)

  Starts the game, runs until the player quits or there is an exception.

Private Types

- typedef std::function< void(BaseActorPtr actor, const int32_t value) > ActorEventHandler

Private Member Functions

- const Point getMouseCoordinates (void) const

  Retrieves the window mouse coordinates.
- const int32_t getMouseClickState (void) const

  Retrives the current mouse button state.
- void handleEvent (BaseActorPtr actor, const ActorEvent &e)

  Handles actor event on actor who triggered it.
- void render (const ActorList &actors)

  Renders all actors in the scene.
- void onChangeScene (BaseActorPtr actor, const int32_t value)
- void onLoadLevel (BaseActorPtr actor, const int32_t value)
- void onPlayAudio (BaseActorPtr actor, const int32_t value)
- void onNewGame (BaseActorPtr actor, const int32_t value)
- void onLoadGame (BaseActorPtr actor, const int32_t value)
- void onDisplayText (BaseActorPtr actor, const int32_t value)
- void onExitGame (BaseActorPtr actor, const int32_t value)
- void onStreamAudio (BaseActorPtr actor, const int32_t value)
- void freeAndLoadLevel (const int32_t id)

Private Attributes

- std::shared_ptr< AudioEngine > mAudioEngine
- std::shared_ptr< VideoEngine > mVideoEngine
- Level mLevel
- Player mPlayer
- bool mIsRunning
- GameSaveSerializer mSerializer
- std::vector< ActorEventHandler > mActorEventHandlers

6.9.1 Detailed Description

Processes all components of the game each frame.

6.9.2 Member Typedef Documentation

6.9.2.1 typedef std::function< void(BaseActorPtr actor, const int32_t value) > teamusa::Engine::ActorEventHandler [private]

6.9.3 Constructor & Destructor Documentation

6.9.3.1 Engine::Engine (void) [explicit]

6.9.3.2 Engine::~Engine (void)

6.9.4 Member Function Documentation

6.9.4.1 void Engine::freeAndLoadLevel (const int32_t *id*) [private]

6.9.4.2 const int32_t Engine::getMouseClickState (void) const [private]

Retrives the current mouse button state.

Returns

Integer describing mouse state.

6.9.4.3 const Point Engine::getMouseCoordinates (void) const [private]

Retrieves the window mouse coordinates.

Returns

A Point struct containg the x and y values of the mouse.

6.9.4.4 void Engine::handleEvent (BaseActorPtr *actor,* const ActorEvent & *e*) [private]

Handles actor event on actor who triggered it.

Looks up function pointer in table, calls the corresponding function.

6.9.4.5 void Engine::onChangeScene (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.6 void Engine::onDisplayText (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.7 void Engine::onExitGame (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.8 void Engine::onLoadGame (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.9 void Engine::onLoadLevel (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.10 void Engine::onNewGame (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.11 void Engine::onPlayAudio (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.12 void Engine::onStreamAudio (BaseActorPtr *actor,* const int32_t *value*) [private]

6.9.4.13 void Engine::render (const ActorList & *actors*) [private]

Renders all actors in the scene.

6.9.4.14 void Engine::run (void)

Starts the game, runs until the player quits or there is an exception.

6.9.5 Member Data Documentation

6.9.5.1 std::vector<ActorEventHandler> teamusa::Engine::mActorEventHandlers [private]

6.9.5.2 std::shared_ptr<AudioEngine> teamusa::Engine::mAudioEngine [private]

6.9.5.3 bool teamusa::Engine::mIsRunning [private]

6.9.5.4 Level teamusa::Engine::mLevel [private]

6.9.5.5 Player teamusa::Engine::mPlayer [private]

6.9.5.6 GameSaveSerializer teamusa::Engine::mSerializer [private]

6.9.5.7 std::shared_ptr<VideoEngine> teamusa::Engine::mVideoEngine [private]

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

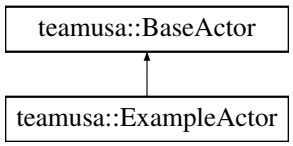
- Engine.h
- Engine.cpp

6.10 teamusa::ExampleActor Class Reference

<Give brief="" description="" here>="">

`#include <ExampleActor.h>`

Inheritance diagram for teamusa::ExampleActor:



Public Member Functions

- ExampleActor (void)
- virtual ~ExampleActor (void) override
- virtual const ActorEvent onClick (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" click>=""
- virtual const ActorEvent onHover (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" hover>=""
- virtual const ActorEvent step (Player &player) override

  <Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>=""

Additional Inherited Members

6.10.1 Detailed Description

<Give brief="" description="" here>=""></Give brief="" description="" here>=""></Give>

6.10.2 Constructor & Destructor Documentation

6.10.2.1 ExampleActor::ExampleActor (void) [explicit]

6.10.2.2 ExampleActor::~ExampleActor (void) [override], [virtual]

6.10.3 Member Function Documentation

6.10.3.1 const ActorEvent ExampleActor::onClick (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" click>=""></Give></Give>

Reimplemented from teamusa::BaseActor.

6.10.3.2 const ActorEvent ExampleActor::onHover (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" hover>=""></Give></Give>

Reimplemented from teamusa::BaseActor.

6.10.3.3 const ActorEvent ExampleActor::step (Player & *player*) [override], [virtual]

<Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>=""></Give></Give>

Implements teamusa::BaseActor.

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

- ExampleActor.h
- ExampleActor.cpp

6.11 teamusa::GameSaveSerializer Class Reference

Provides multithreaded save, single-thread load of save files.

`#include <GameSaveSerializer.h>`

Public Member Functions

- GameSaveSerializer (void)
- ~GameSaveSerializer (void)
- void setSlot (const int32_t slot)

  Sets the slot number to save/load in.
- bool load (int &levelID, int &sceneID, Player::Inventory &inventory)

  Loads a save file.
- void save (const int &levelID, const int &sceneID, const Player::Inventory &inventory)

  Saves a file.
- void saveInThread (const int levelID, const int sceneID, const Player::Inventory inventory)

  Saves a file in a separate thread.

Private Attributes

- std::mutex fileLock
- int32_t slot

6.11.1 Detailed Description

Provides multithreaded save, single-thread load of save files.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 teamusa::GameSaveSerializer::GameSaveSerializer (void)

6.11.2.2 teamusa::GameSaveSerializer::~GameSaveSerializer (void)

6.11.3 Member Function Documentation

6.11.3.1 bool teamusa::GameSaveSerializer::load (int & *levelID,* int & *sceneID,* Player::Inventory & *inventory*)

Loads a save file.

Returns

True if save file was loaded successfully, false if it doesn't exist.

6.11.3.2 void teamusa::GameSaveSerializer::save (const int & *levelID,* const int & *sceneID,* const Player::Inventory & *inventory*)

Saves a file.

6.11.3.3 void teamusa::GameSaveSerializer::saveInThread (const int *levelID,* const int *sceneID,* const Player::Inventory *inventory*)

Saves a file in a separate thread.

6.11.3.4 void teamusa::GameSaveSerializer::setSlot (const int32_t *slot*)

Sets the slot number to save/load in.

6.11.4 Member Data Documentation

6.11.4.1 std::mutex teamusa::GameSaveSerializer::fileLock [private]

6.11.4.2 int32_t teamusa::GameSaveSerializer::slot [private]

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

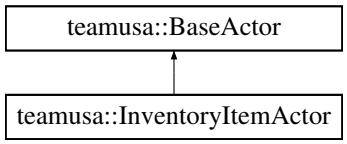
- GameSaveSerializer.h
- GameSaveSerializer.cpp

6.12 teamusa::InventoryItemActor Class Reference

<Give brief="" description="" here>="">

`#include <InventoryItemActor.h>`

Inheritance diagram for teamusa::InventoryItemActor:



Public Member Functions

- InventoryItemActor (Region region, const int itemID=-1, const int textureID=-1, const int layer=-1)
- virtual ~InventoryItemActor (void) override
- virtual const ActorEvent onHover (Player &player) override

  Called when the actor is hovered over with the mouse.
- virtual const ActorEvent onClick (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" click>=""
- virtual const ActorEvent step (Player &player) override

  Called each frame, each derived actor should handle this.

Private Attributes

- int itemID
- bool pickedUp = false

Additional Inherited Members

6.12.1 Detailed Description

<Give brief="" description="" here>=""

6.12.2 Constructor & Destructor Documentation

6.12.2.1 InventoryItemActor::InventoryItemActor (Region *region,* const int *itemID* = −1, const int *textureID* = −1, const int *layer* = −1) [explicit]

6.12.2.2 InventoryItemActor::~InventoryItemActor (void) [override], [virtual]

6.12.3 Member Function Documentation

6.12.3.1 const ActorEvent InventoryItemActor::onClick (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" click>=""

Reimplemented from teamusa::BaseActor.

6.12.3.2 const ActorEvent InventoryItemActor::onHover (Player & *player*) [override], [virtual]

Called when the actor is hovered over with the mouse.

Parameters

player	The player in the scene.

Returns

The ActorEvent to be handled by Engine when hovered over.

Reimplemented from teamusa::BaseActor.

6.12.3.3 const ActorEvent InventoryItemActor::step (Player & *player*) [override], [virtual]

Called each frame, each derived actor should handle this.

Parameters

player	The player in the scene.

Returns

Any ActorEvent that should be handled immediately by Engine.

Implements teamusa::BaseActor.

6.12.4 Member Data Documentation

6.12.4.1 int teamusa::InventoryItemActor::itemID [private]

6.12.4.2 bool teamusa::InventoryItemActor::pickedUp = false [private]

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

- InventoryItemActor.h
- InventoryItemActor.cpp

6.13 teamusa::Level Class Reference

`#include <Level.h>`

Classes

- struct Scene

Public Member Functions

- Level (void)
- Level (int levelID, AudioEngine &audioEngine, VideoEngine &videoEngine)
- const ActorList & getActors (void) const
- const int getBGImageID (void) const
- const int loadLevel (const std::string &path, AudioEngine &audioEngine, VideoEngine &videoEngine)
- void changeScene (const int sceneID)
- const int getScene ()
- void clearAll (void)

Private Member Functions

- BaseActorPtr parseAudioStreamActor (std::fstream &fs)
- BaseActorPtr parseDelayedAudioActor (std::fstream &fs)
- BaseActorPtr parseDelayedVideoActor (std::fstream &fs)
- BaseActorPtr parseInventoryItemActor (std::fstream &fs)
- BaseActorPtr parseLevelLink (std::fstream &fs)
- BaseActorPtr parseMovingActor (std::fstream &fs)
- BaseActorPtr parseResponsiveAudioActor (std::fstream &fs)
- BaseActorPtr parseResponsiveVideoActor (std::fstream &fs)
- BaseActorPtr parseSceneLink (std::fstream &fs)
- BaseActorPtr parseTextboxSpawnActor (std::fstream &fs)
- BaseActorPtr parseVideoActor (std::fstream &fs)
- BaseActorPtr parseVideoEventActor (std::fstream &fs)

Private Attributes

- std::unordered_map< int, Scene > scenes
- int startScene
- int activeScene

6.13.1 Constructor & Destructor Documentation

6.13.1.1 Level::Level (void)

6.13.1.2 Level::Level (int *levelID,* AudioEngine & *audioEngine,* VideoEngine & *videoEngine*)

6.13.2 Member Function Documentation

6.13.2.1 void Level::changeScene (const int *sceneID*)

6.13.2.2 void Level::clearAll (void)

6.13.2.3 const ActorList & Level::getActors (void) const

6.13.2.4 const int Level::getBGImageID (void) const

6.13.2.5 const int Level::getScene ( )

6.13.2.6 const int Level::loadLevel (const std::string & *path,* AudioEngine & *audioEngine,* VideoEngine & *videoEngine*)

6.13.2.7 BaseActorPtr Level::parseAudioStreamActor (std::fstream & *fs*) [private]

6.13.2.8 BaseActorPtr Level::parseDelayedAudioActor (std::fstream & *fs*) [private]

6.13.2.9 BaseActorPtr Level::parseDelayedVideoActor (std::fstream & *fs*) [private]

6.13.2.10 BaseActorPtr Level::parseInventoryItemActor (std::fstream & *fs*) [private]

6.13.2.11 BaseActorPtr Level::parseLevelLink (std::fstream & *fs*) [private]

6.13.2.12 BaseActorPtr Level::parseMovingActor (std::fstream & *fs*) [private]

6.13.2.13 BaseActorPtr Level::parseResponsiveAudioActor (std::fstream & *fs*) [private]

6.13.2.14 BaseActorPtr Level::parseResponsiveVideoActor (std::fstream & *fs*) [private]

6.13.2.15 BaseActorPtr Level::parseSceneLink (std::fstream & *fs*) [private]

6.13.2.16 BaseActorPtr Level::parseTextboxSpawnActor (std::fstream & *fs*) [private]

6.13.2.17 BaseActorPtr Level::parseVideoActor (std::fstream & *fs*) [private]

6.13.2.18 BaseActorPtr Level::parseVideoEventActor (std::fstream & *fs*) [private]

6.13.3 Member Data Documentation

6.13.3.1 int teamusa::Level::activeScene [private]

6.13.3.2 std::unordered_map<int, Scene> teamusa::Level::scenes [private]

6.13.3.3 int teamusa::Level::startScene [private]

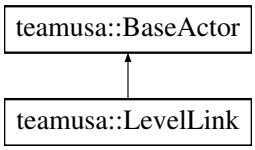
The documentation for this class was generated from the following files:

- Level.h
- Level.cpp

6.14 teamusa::LevelLink Class Reference

`#include <LevelLink.h>`

Inheritance diagram for teamusa::LevelLink:



Public Member Functions

- LevelLink (Region region, const int Level_ID, const int sceneID, const std::string itemRequired_Text, const int item_ID=-1)
- virtual ~LevelLink (void) override
- virtual const ActorEvent onClick (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" click>=""
- virtual const ActorEvent onHover (Player &player) override

  Called when the actor is hovered over with the mouse.
- virtual const ActorEvent step (Player &player) override

  Called each frame, each derived actor should handle this.
- const int getSceneID (void) const
- virtual const std::string getText ()

Private Attributes

- int sceneID
- int levelID

- std::string itemRequiredText
- int requiredItemID = -1

Additional Inherited Members

6.14.1 Constructor & Destructor Documentation

6.14.1.1 LevelLink::LevelLink (Region *region,* const int *Level_ID,* const int *sceneID,* const std::string *itemRequired_Text,* const int *item_ID* = −1) [explicit]

6.14.1.2 LevelLink::∼LevelLink (void) [override], [virtual]

6.14.2 Member Function Documentation

6.14.2.1 const int LevelLink::getSceneID (void) const

6.14.2.2 const std::string LevelLink::getText ( ) [virtual]

6.14.2.3 const ActorEvent LevelLink::onClick (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" click>="">

Reimplemented from teamusa::BaseActor.

6.14.2.4 const ActorEvent LevelLink::onHover (Player & *player*) [override], [virtual]

Called when the actor is hovered over with the mouse.

Parameters

player	The player in the scene.

Returns

The ActorEvent to be handled by Engine when hovered over.

Reimplemented from teamusa::BaseActor.

6.14.2.5 const ActorEvent LevelLink::step (Player & *player*) [override], [virtual]

Called each frame, each derived actor should handle this.

Parameters

player	The player in the scene.

Returns

Any ActorEvent that should be handled immediately by Engine.

Implements teamusa::BaseActor.

6.14.3 Member Data Documentation

6.14.3.1 std::string teamusa::LevelLink::itemRequiredText [private]

6.14.3.2 int teamusa::LevelLink::levelID [private]

6.14.3.3 int teamusa::LevelLink::requiredItemID = -1 [private]

6.14.3.4 int teamusa::LevelLink::sceneID [private]

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

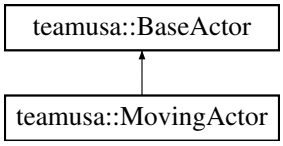
- LevelLink.h
- LevelLink.cpp

6.15 teamusa::MovingActor Class Reference

Will transition from one region to the next by calculating the distance to move each frame for a set number of frames.

`#include <MovingActor.h>`

Inheritance diagram for teamusa::MovingActor:



Public Member Functions

- MovingActor (Region startRegion, Region endregion, int textureId, int layer, int transitionsteps, bool moveOnSpawn)
- virtual ~MovingActor (void) override
- virtual const ActorEvent onClick (Player &player) override

  If move on spwan is not set then step will ove the actor.
- virtual const ActorEvent onHover (Player &player) override

  If inactive, display a select cursor.
- virtual const ActorEvent step (Player &player)

  If moveOnSpawn is set, then the step method will advance the actor immediately.

Private Attributes

- Region endRegion
- int transitionSteps = 1
- int currentStep = 0
- int xSpeed = 0
- int ySpeed = 0
- int hGrowth = 0
- int wGrowth = 0
- bool isActive = false

Additional Inherited Members

6.15.1 Detailed Description

Will transition from one region to the next by calculating the distance to move each frame for a set number of frames.

This allows for movement across the X and Y axis as well as scaling of the size of textures.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 MovingActor::MovingActor (Region *startRegion,* Region *endregion,* int *textureId,* int *layer,* int *transitionsteps,* bool *moveOnSpawn*) [explicit]

6.15.2.2 MovingActor::~MovingActor (void) [override], [virtual]

6.15.3 Member Function Documentation

6.15.3.1 const ActorEvent MovingActor::onClick (Player & *player*) [override], [virtual]

If move on spwan is not set then step will ove the actor.

Reimplemented from teamusa::BaseActor.

6.15.3.2 const ActorEvent MovingActor::onHover (Player & *player*) [override], [virtual]

If inactive, display a select cursor.

Reimplemented from teamusa::BaseActor.

6.15.3.3 const ActorEvent MovingActor::step (Player & *player*) [virtual]

If moveOnSpawn is set, then the step method will advance the actor immediately.

Implements teamusa::BaseActor.

6.15.4 Member Data Documentation

6.15.4.1 int teamusa::MovingActor::currentStep = 0 [private]

6.15.4.2 Region teamusa::MovingActor::endRegion [private]

6.15.4.3 int teamusa::MovingActor::hGrowth = 0 [private]

6.15.4.4 bool teamusa::MovingActor::isActive = false [private]

6.15.4.5 int teamusa::MovingActor::transitionSteps = 1 [private]

6.15.4.6 int teamusa::MovingActor::wGrowth = 0 [private]

6.15.4.7 int teamusa::MovingActor::xSpeed = 0 [private]

6.15.4.8 int teamusa::MovingActor::ySpeed = 0 [private]

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

- MovingActor.h
- MovingActor.cpp

6.16 teamusa::Player Class Reference

Handles all data relevant to the player engaging the game.

`#include <Player.h>`

Public Types

- typedef std::vector< int32_t > Inventory

Public Member Functions

- Player (void)
- ~Player (void)
- const bool hasItem (const int32_t itemType) const

  Tests if the player has an item in their inventory.
- void addItem (const int32_t itemType)

  Inserts an item into the player's inventory.
- void setCursor (const CursorStyle style)

  Sets the visual style of the player's mouse cursor.
- const int getCursorTextureID (void) const

  Returns the current cursor texture ID associated with the cursor style.
- void setPosition (const int32_t x, const int32_t y)

  Sets the position of the player's cursor.
- void setPosition (const Point &position)

  Sets the position of the player's cursor.
- const Point getPosition (void) const

  Gets the player's cursor position.
- const Inventory & getInventory () const

  Returns the player's inventory.
- void setInventory (const Inventory &inventory)

  Clears the player's current inventory and assigns the new one.

Static Public Attributes

- static const int FLASHLIGHT_ID = 1666
- static const int CURSOR_DEFAULT_ID = 1667
- static const int CURSOR_SELECT_ID = 1668
- static const int CURSOR_UP_ID = 1669
- static const int CURSOR_DOWN_ID = 1670
- static const int CURSOR_LEFT_ID = 1671
- static const int CURSOR_RIGHT_ID = 1672
- static const int MOUSE_CLICK_ID = 1700

Private Attributes

- Region mRegion
- int32_t mLayer
- int32_t mTextureID
- Point mPosition
- Inventory mInventory
- CursorStyle mCursorStyle

6.16.1 Detailed Description

Handles all data relevant to the player engaging the game.

6.16.2 Member Typedef Documentation

6.16.2.1 typedef std::vector<int32_t> teamusa::Player::Inventory

6.16.3 Constructor & Destructor Documentation

6.16.3.1 Player::Player (void) [explicit]

6.16.3.2 Player::~Player (void)

6.16.4 Member Function Documentation

6.16.4.1 void Player::addItem (const int32_t *itemType*)

Inserts an item into the player's inventory.

Parameters

itemType	The item identifier to insert.

6.16.4.2 const int Player::getCursorTextureID (void) const

Returns the current cursor texture ID associated with the cursor style.

6.16.4.3 const Player::Inventory & Player::getInventory ( ) const

Returns the player's inventory.

6.16.4.4 const Point Player::getPosition (void) const

Gets the player's cursor position.

Returns

A Point struct containing the cursor position.

6.16.4.5 const bool Player::hasItem (const int32_t *itemType*) const

Tests if the player has an item in their inventory.

Parameters

itemType	The item type identifier.

Returns

True if the player has the item.

6.16.4.6 void Player::setCursor (const CursorStyle *style*)

Sets the visual style of the player's mouse cursor.

Parameters

style	The style type for the cursor.

6.16.4.7 void Player::setInventory (const Inventory & *inventory*)

Clears the player's current inventory and assigns the new one.

Parameters

inventory	The inventory to assign to the player.

6.16.4.8 void Player::setPosition (const int32_t *x*, const int32_t *y*)

Sets the position of the player's cursor.

Parameters

x	The x-coordinate of the cursor.
y	The y-coordinate of the cursor.

6.16.4.9 void Player::setPosition (const Point & *position*)

Sets the position of the player's cursor.

Parameters

position	A Point struct containing the cursor position.

6.16.5 Member Data Documentation

6.16.5.1 const int Player::CURSOR_DEFAULT_ID = 1667 [static]

6.16.5.2 const int Player::CURSOR_DOWN_ID = 1670 [static]

6.16.5.3 const int Player::CURSOR_LEFT_ID = 1671 [static]

6.16.5.4 const int Player::CURSOR_RIGHT_ID = 1672 [static]

6.16.5.5 const int Player::CURSOR_SELECT_ID = 1668 [static]

6.16.5.6 const int Player::CURSOR_UP_ID = 1669 [static]

6.16.5.7 const int Player::FLASHLIGHT_ID = 1666 [static]

6.16.5.8 CursorStyle teamusa::Player::mCursorStyle [private]

6.16.5.9 Inventory teamusa::Player::mInventory [private]

6.16.5.10 int32_t teamusa::Player::mLayer [private]

6.16.5.11 const int Player::MOUSE_CLICK_ID = 1700 [static]

6.16.5.12 Point teamusa::Player::mPosition [private]

6.16.5.13 Region teamusa::Player::mRegion [private]

6.16.5.14 int32_t teamusa::Player::mTextureID [private]

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

- Player.h
- Player.cpp

6.17 teamusa::Point Struct Reference

`#include <Point.h>`

Public Member Functions

- Point (void)
- Point (int32_t x, const int32_t y)

Public Attributes

- int32_t x
- int32_t y

6.17.1 Constructor & Destructor Documentation

6.17.1.1 teamusa::Point::Point (void) [inline]

6.17.1.2 teamusa::Point::Point (int32_t *x*, const int32_t *y*) [inline]

6.17.2 Member Data Documentation

6.17.2.1 int32_t teamusa::Point::x

6.17.2.2 int32_t teamusa::Point::y

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

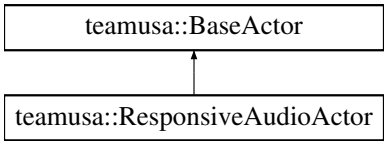
- Point.h

6.18 teamusa::ResponsiveAudioActor Class Reference

\ Brief: Will increment the value of stepCount until it is equal to durationSteps for each call to the step method.

`#include <ResponsiveAudioActor.h>`

Inheritance diagram for teamusa::ResponsiveAudioActor:



Public Member Functions

- ResponsiveAudioActor (Region region, int hoverAudioId, int clickAudioId)
- virtual ~ResponsiveAudioActor (void) override
- virtual const ActorEvent onClick (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" click>="">
- virtual const ActorEvent onHover (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" hover>="">
- virtual const ActorEvent step (Player &player) override

  <Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>="">

Private Attributes

- int hoverAudioId
- int clickAudioId

Additional Inherited Members

6.18.1 Detailed Description

\ Brief: Will increment the value of stepCount until it is equal to durationSteps for each call to the step method.

A call to onClick or onHover will set the value of stepCount to zero and emit an AudioID and value if stepCount is equal to durationSteps. The hoverAudioID or clickAudioID can be set to an invalid AudioID value to prevent sound from being played.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 ResponsiveAudioActor::ResponsiveAudioActor (Region *region,* int *hoverAudioId* = −1, int *clickAudioId* = −1) [explicit]

6.18.2.2 ResponsiveAudioActor::~ResponsiveAudioActor (void) [override], [virtual]

6.18.3 Member Function Documentation

6.18.3.1 const ActorEvent ResponsiveAudioActor::onClick (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" click>="">

Reimplemented from teamusa::BaseActor.

6.18.3.2 const ActorEvent ResponsiveAudioActor::onHover (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" hover>="">

Reimplemented from teamusa::BaseActor.

6.18.3.3 const ActorEvent ResponsiveAudioActor::step (Player & *player*) [override], [virtual]

<Give description="" of="" what="" happens="" each="" frame="" to="" this="" actor>="">

Implements teamusa::BaseActor.

6.18.4 Member Data Documentation

6.18.4.1 int teamusa::ResponsiveAudioActor::clickAudioId [private]

6.18.4.2 int teamusa::ResponsiveAudioActor::hoverAudioId [private]

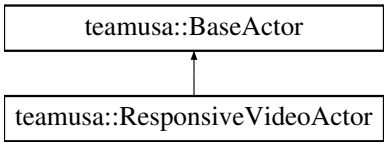
The documentation for this class was generated from the following files:

- ResponsiveAudioActor.h
- ResponsiveAudioActor.cpp

6.19 teamusa::ResponsiveVideoActor Class Reference

`#include <ResponsiveVideoActor.h>`

Inheritance diagram for teamusa::ResponsiveVideoActor:



Public Member Functions

- ResponsiveVideoActor (Region region, int hoverTextureId, int clickTextureID, int defaulTextureID, int layer)
- virtual ~ResponsiveVideoActor (void) override
- virtual const ActorEvent onClick (Player &player) override

  When onClick is called, the onClick texture will be set permanently. Can disappear or appear by utilizing an invalid TextureID.
- virtual const ActorEvent onHover (Player &player) override

  When onHover is called, it will display the hover TextureID.
- virtual const ActorEvent step (Player &player) override

  does nothing.
- void setTextureId (int TextureId)

Private Attributes

- int hoverTexture
- int clickTexture
- int defaultTextureId

Additional Inherited Members

6.19.1 Constructor & Destructor Documentation

6.19.1.1 ResponsiveVideoActor::ResponsiveVideoActor (Region *region,* int *hoverTextureId,* int *clickTextureID,* int *defaulTextureID,* int *layer*) [explicit]

6.19.1.2 ResponsiveVideoActor::~ResponsiveVideoActor (void) [override], [virtual]

6.19.2 Member Function Documentation

6.19.2.1 const ActorEvent ResponsiveVideoActor::onClick (Player & *player*) [override], [virtual]

When onClick is called, the onClick texture will be set permanently. Can disappear or appear by utilizing an invalid TextureID.

Reimplemented from teamusa::BaseActor.

6.19.2.2 const ActorEvent ResponsiveVideoActor::onHover (Player & *player*) [override], [virtual]

When onHover is called, it will display the hover TextureID.

Reimplemented from teamusa::BaseActor.

6.19.2.3 void ResponsiveVideoActor::setTextureId (int *TextureId*)

6.19.2.4 const ActorEvent ResponsiveVideoActor::step (Player & *player*) [override], [virtual]

does nothing.

Implements teamusa::BaseActor.

6.19.3 Member Data Documentation

6.19.3.1 int teamusa::ResponsiveVideoActor::clickTexture [private]

6.19.3.2 int teamusa::ResponsiveVideoActor::defaultTextureId [private]

6.19.3.3 int teamusa::ResponsiveVideoActor::hoverTexture [private]

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

- ResponsiveVideoActor.h
- ResponsiveVideoActor.cpp

6.20 teamusa::Level::Scene Struct Reference

Public Attributes

- ActorList actors
- int bgImageID

6.20.1 Member Data Documentation

6.20.1.1 ActorList teamusa::Level::Scene::actors

6.20.1.2 int teamusa::Level::Scene::bgImageID

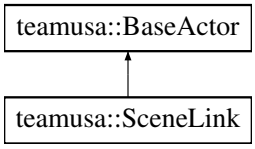
The documentation for this struct was generated from the following file:

- Level.h

6.21 teamusa::SceneLink Class Reference

`#include <SceneLink.h>`

Inheritance diagram for teamusa::SceneLink:



Public Member Functions

- SceneLink (Region region, const int scene_ID, const std::string &itemRequired_Text, const int item_ID=-1)
- virtual ~SceneLink (void) override
- virtual const ActorEvent onClick (Player &player) override

  <Give description="" of="" what="" actor="" does="" on="" click>=""
- virtual const ActorEvent onHover (Player &player) override

  Called when the actor is hovered over with the mouse.
- virtual const ActorEvent step (Player &player) override

  Called each frame, each derived actor should handle this.
- virtual const std::string getText ()

Private Attributes

- int sceneID
- std::string itemRequiredText
- int requiredItemID
- CursorStyle cursorStyle

Additional Inherited Members

6.21.1 Constructor & Destructor Documentation

6.21.1.1 SceneLink::SceneLink (Region *region*, const int *scene_ID*, const std::string & *itemRequired_Text*, const int *item_ID* = -1) [explicit]

6.21.1.2 SceneLink::~SceneLink (void) [override], [virtual]

6.21.2 Member Function Documentation

6.21.2.1 const std::string SceneLink::getText ( ) [virtual]

6.21.2.2 const ActorEvent SceneLink::onClick (Player & *player*) [override], [virtual]

<Give description="" of="" what="" actor="" does="" on="" click>=""

Reimplemented from teamusa::BaseActor.

6.21.2.3 const ActorEvent SceneLink::onHover (Player & *player*) [override], [virtual]

Called when the actor is hovered over with the mouse.

Parameters

player	The player in the scene.

Returns

The ActorEvent to be handled by Engine when hovered over.

Reimplemented from teamusa::BaseActor.

6.21.2.4 const ActorEvent SceneLink::step (Player & *player*) [override], [virtual]

Called each frame, each derived actor should handle this.

Parameters

player	The player in the scene.

Returns

Any ActorEvent that should be handled immediately by Engine.

Implements teamusa::BaseActor.

6.21.3 Member Data Documentation

6.21.3.1 CursorStyle teamusa::SceneLink::cursorStyle [private]

6.21.3.2 std::string teamusa::SceneLink::itemRequiredText [private]

6.21.3.3 int teamusa::SceneLink::requiredItemID [private]

6.21.3.4 int teamusa::SceneLink::sceneID [private]

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

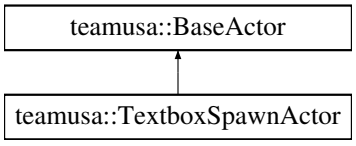
- SceneLink.h
- SceneLink.cpp

6.22 teamusa::TextboxSpawnActor Class Reference

<Give brief="" description="" here>="">

`#include <TextboxSpawnActor.h>`

Inheritance diagram for teamusa::TextboxSpawnActor:



Public Member Functions

- TextboxSpawnActor (Region region, std::string text)
- virtual ~TextboxSpawnActor (void)
- virtual const ActorEvent onClick (Player &player)

  return a display text actor event
- virtual const ActorEvent step (Player &player)

  Override.
- std::string getText (void)

  Retrurn the text when called.

Private Attributes

- std::string text
- bool activated

Additional Inherited Members

6.22.1 Detailed Description

<Give brief="" description="" here>="">

Will emit a DisplayText event when the onClick method is called.

The actor can then have its text accessed by the engine for display through a call to the getText method.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 TextboxSpawnActor::TextboxSpawnActor (Region *region,* std::string *text*) [explicit]

6.22.2.2 TextboxSpawnActor::~TextboxSpawnActor (void) [virtual]

6.22.3 Member Function Documentation

6.22.3.1 std::string TextboxSpawnActor::getText (void)

Retrurn the text when called.

6.22.3.2 const ActorEvent TextboxSpawnActor::onClick (Player & *player*) [virtual]

return a display text actor event

Reimplemented from teamusa::BaseActor.

6.22.3.3 const ActorEvent TextboxSpawnActor::step (Player & *player*) [virtual]

Override.

Implements teamusa::BaseActor.

6.22.4 Member Data Documentation

6.22.4.1 bool teamusa::TextboxSpawnActor::activated [private]

6.22.4.2 std::string teamusa::TextboxSpawnActor::text [private]

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

- TextboxSpawnActor.h
- TextboxSpawnActor.cpp

6.23 teamusa::Timer Class Reference

A timer that counts up from zero in milliseconds.

`#include <Timer.h>`

Public Member Functions

- Timer (void)
- ~Timer (void)
- const uint32_t start (void)

  Starts the timer.
- void stop (void)

  Stops the timer.
- void pause (void)

  Pauses the timer.
- void unpause (void)

  Unpauses the timer.
- const uint32_t getTicks (void) const

  Gets the time in milliseconds since the timer was started.

Private Attributes

- uint32_t mStartTicks
- uint32_t mPauseTicks
- bool mPaused
- bool mStarted

6.23.1 Detailed Description

A timer that counts up from zero in milliseconds.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Timer::Timer (void) [explicit]

6.23.2.2 Timer::~Timer (void)

6.23.3 Member Function Documentation

6.23.3.1 const uint32_t Timer::getTicks (void) const

Gets the time in milliseconds since the timer was started.

Returns

The elapsed time.

6.23.3.2 void Timer::pause (void)

Pauses the timer.

6.23.3.3 const uint32_t Timer::start (void)

Starts the timer.

6.23.3.4 void Timer::stop (void)

Stops the timer.

6.23.3.5 void Timer::unpause (void)

Unpauses the timer.

6.23.4 Member Data Documentation

6.23.4.1 bool teamusa::Timer::mPaused [private]

6.23.4.2 uint32_t teamusa::Timer::mPauseTicks [private]

6.23.4.3 bool teamusa::Timer::mStarted [private]

6.23.4.4 uint32_t teamusa::Timer::mStartTicks [private]

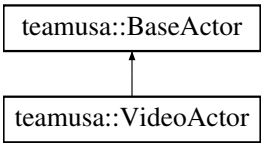
The documentation for this class was generated from the following files:

- Timer.h
- Timer.cpp

6.24 teamusa::VideoActor Class Reference

`#include <VideoActor.h>`

Inheritance diagram for teamusa::VideoActor:



Public Member Functions

- VideoActor (Region region, int textureId, int layer)
- virtual ~VideoActor (void) override
- virtual const ActorEvent step (Player &player) override

  Called each frame, each derived actor should handle this.

Additional Inherited Members

6.24.1 Constructor & Destructor Documentation

6.24.1.1 VideoActor::VideoActor (Region *region*, int *textureId* = −1, int *layer* = 1) [explicit]

6.24.1.2 VideoActor::~VideoActor (void) [override], [virtual]

6.24.2 Member Function Documentation

6.24.2.1 const ActorEvent VideoActor::step (Player & *player*) [override], [virtual]

Called each frame, each derived actor should handle this.

Parameters

player	The player in the scene.

Returns

Any ActorEvent that should be handled immediately by Engine.

Implements teamusa::BaseActor.

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

- VideoActor.h
- VideoActor.cpp

6.25 mediawrap::VideoContext Class Reference

Provides basic 2D rendering capabilities.

`#include <VideoContext.hpp>`

Public Types

- enum Flip { FLIP_NONE = SDL_FLIP_NONE, FLIP_HORIZONTAL = SDL_FLIP_HORIZONTAL, FLIP_V↩ ERTICAL = SDL_FLIP_VERTICAL }

  Used to designate how an image should be flipped across an axis.
- enum BlendMode { BLENDMODE_NONE = SDL_BLENDMODE_NONE, BLENDMODE_BLEND = SDL_↩ BLENDMODE_BLEND, BLENDMODE_ADD = SDL_BLENDMODE_ADD, BLENDMODE_MOD = SDL_BL↩ ENDMODE_MOD }

  Used to specify how a texture should behave when objects are rendered onto it.
- enum DebugColor { RED = 0, GREEN, BLUE }
- typedef SDL_Rect Region

  Used to specify x, y, width, height of an texture source or destination region.

- typedef unsigned int TextureID

  Used to identify each texture uniquely.
- typedef std::unordered_map< TextureID, SDL_Texture * >::iterator texture_iter

  Used to access elements in the texture map.

Public Member Functions

- VideoContext (const std::string &title, unsigned int width, unsigned int height)

  Constructs a new rendering context that includes a window and the renderer associated with it.
- ~VideoContext (void)

  Deletes the renderer and window associated with this context.
- void display (void)

  Displays the rendered textures on screen.
- Region load_texture (TextureID id, const std::string &image_path, BlendMode blend=BLENDMODE_BLEND)

  Loads a texture from the filename into the specified texture id.
- Region create_texture (TextureID id, int width, int height, BlendMode blend=BLENDMODE_BLEND)

  Creates a blank texture, which should be filled completely or cleared before rendering to prevent old fragments from appearing.
- void delete_texture (TextureID id)

  The deletes the given texture from this context.
- void render (TextureID id, Region *dest, Region *src)

  Draws the given texture onto the canvas.
- void renderDebugBox (const Region &region, const DebugColor color, const TextureID layer)
- void render_onto (TextureID dest_id, TextureID src_id, const Region *dest_region, Region *src_region)

  Draws the given source texture onto the destination texture.
- void render_rotate (TextureID dest_id, TextureID src_id, Region *dest_region, Region *src_region, double angle=0.0, Flip flip=FLIP_NONE)

  Draws the given source texture onto the destination texture after applying a rotate and flip operation.
- void render_clear ()

  Clears the canvas with the default clear color.
- void render_clear (TextureID id)

  Clears the given texture with the default clear color.
- void fill_texture (TextureID id, int r, int g, int b, int a)

  Fills the given texture with the given rgba value.
- void load_font (const std::string &font_path, int font_size)

  Loads the given font from the path specified.
- void render_text (TextureID dest_id, Region *dest_region, const std::string &text, Uint8 r, Uint8 g, Uint8 b, Uint8 a)

  Renders the given text onto the the destination texture.

Private Attributes

- std::unordered_map< TextureID, SDL_Texture * > * textures
- VideoDisplay * video_display
- SDL_Renderer * renderer
- TTF_Font * font

6.25.1 Detailed Description

Provides basic 2D rendering capabilities.

Acts as an abstraction layer to the SDL2 video library.

6.25.2 Member Typedef Documentation

6.25.2.1 typedef SDL_Rect mediawrap::VideoContext::Region

Used to specify x, y, width, height of an texture source or destination region.

6.25.2.2 typedef std::unordered_map<TextureID, SDL_Texture*>::iterator mediawrap::VideoContext::texture_iter

Used to access elements in the texture map.

6.25.2.3 typedef unsigned int mediawrap::VideoContext::TextureID

Used to identify each texture uniquely.

Each texture loaded is to be assigned a key of this type.

6.25.3 Member Enumeration Documentation

6.25.3.1 enum mediawrap::VideoContext::BlendMode

Used to specify how a texture should behave when objects are rendered onto it.

Enumerator

BLENDMODE_NONE

BLENDMODE_BLEND

BLENDMODE_ADD

BLENDMODE_MOD

6.25.3.2 enum mediawrap::VideoContext::DebugColor

Enumerator

RED

GREEN

BLUE

6.25.3.3 enum mediawrap::VideoContext::Flip

Used to designate how an image should be flipped across an axis.

These two values can be ORed together to achive both effects.

Enumerator

FLIP_NONE

FLIP_HORIZONTAL

FLIP_VERTICAL

6.25.4 Constructor & Destructor Documentation

6.25.4.1 VideoContext::VideoContext (const std::string & *title,* unsigned int *width,* unsigned int *height*)

Constructs a new rendering context that includes a window and the renderer associated with it.

Provides utilities for loading textures and storing them in an internal mapping.

Parameters

title	The title to display at the top of the window.
width	The width of the window created.
height	The height of the window created.

6.25.4.2 VideoContext::~VideoContext (void)

Deletes the renderer and window associated with this context.

Also deletes all textures currently loaded by this context.

6.25.5 Member Function Documentation

6.25.5.1 VideoContext::Region VideoContext::create_texture (TextureID *id,* int *width,* int *height,* BlendMode *blend =* BLENDMODE_BLEND)

Creates a blank texture, which should be filled completely or cleared before rendering to prevent old fragments from appearing.

Must be deleted using delete_texture.

Parameters

id	The id to assign to this texture. If this id is already in use, it deletes the existing texture first before loading this new one.
width	The width of the new texture
height	The height of the next texture
blend	The blending mode which decides how to react with other textures. Defaults to BLENDMO↩ DE_BLEND.

Returns

The source region of the new texture created.

6.25.5.2 void VideoContext::delete_texture (TextureID *id*)

The deletes the given texture from this context.

Parameters

id	The id of the texture to delete.

6.25.5.3 void VideoContext::display (void)

Displays the rendered textures on screen.

6.25.5.4 void VideoContext::fill_texture (TextureID *id,* int *r,* int *g,* int *b,* int *a*)

Fills the given texture with the given rgba value.

Parameters

id	The id of the texture to fill with the specified color.
r	The red value 0-255
g	The green value 0-255
b	The blue value 0-255
a	The alpha value 0-255

6.25.5.5 void VideoContext::load_font (const std::string & *font_path*, int *font_size*)

Loads the given font from the path specified.

Only one font may be loaded at any given time. Repeated calls to this function will delete the previous font before creating a new one.

Parameters

font_path	The path to the ttf file to load as a font.
font_size	The size of the font to load.

6.25.5.6 VideoContext::Region VideoContext::load_texture (TextureID *id*, const std::string & *image_path*, BlendMode *blend* = BLENDMODE_BLEND)

Loads a texture from the filename into the specified texture id.

Must be deleted using delete_texture.

Parameters

id	The id to assign to this texture. If this id is already in use, it deletes the existing texture first before loading this new one.
image_path	The path of the file to load as a texture.
blend	The blending mode which decides how to react with other textures. Defaults to BLENDMO↩ DE_BLEND.

Returns

The auto-detected source rectangle for this image.

6.25.5.7 void VideoContext::render (TextureID *id*, Region * *dest*, Region * *src*)

Draws the given texture onto the canvas.

Parameters

id	The id of the texture to draw onto the canvas.
dest	The destination region to draw onto the canvas.
src	The source region to copy from when drawing.

6.25.5.8 void VideoContext::render_clear ( )

Clears the canvas with the default clear color.

6.25.5.9 void VideoContext::render_clear (TextureID *id*)

Clears the given texture with the default clear color.

Parameters

id	The id of the texture to clear.

6.25.5.10 void VideoContext::render_onto (TextureID *dest_id*, TextureID *src_id*, const Region * *dest_region*, Region * *src_region*)

Draws the given source texture onto the destination texture.

Parameters

dest_id	The id of the texture that will act as a canvas and be drawn on.
src_id	The id of the texture to draw over the destination Texture.
dest_region	The region to draw the source texture into.
src_region	The region to copy the source texture from.

6.25.5.11 void VideoContext::render_rotate (TextureID *dest_id*, TextureID *src_id*, Region * *dest_region*, Region * *src_region*, double *angle* = 0.0, Flip *flip* = FLIP_NONE)

Draws the given source texture onto the destination texture after applying a rotate and flip operation.

Parameters

dest_id	The id of the texture that will act as a canvas and be drawn on.
src_id	The id of the texture to draw over the destination Texture.
dest_region	The region to draw the source texture into.
src_region	The region to copy the source texture from.
angle	The angle in degrees to rotate the source image. Defaults to zero.
flip	The direction to flip the source texture in. Defaults to none.

6.25.5.12 void VideoContext::render_text (TextureID *dest_id*, Region * *dest_region*, const std::string & *text*, Uint8 *r*, Uint8 *g*, Uint8 *b*, Uint8 *a*)

Renders the given text onto the the destination texture.

A successful call to load_font must be performed before this method should be called.

Parameters

dest_id	The destination texture to render onto.
dest_region	The region on the destination texture to render the font into.
text	The string to render.
r	The red value 0-255
g	The green value 0-255
b	The blue value 0-255
a	The alpha value 0-255

6.25.5.13 void VideoContext::renderDebugBox (const Region & *region*, const DebugColor *color*, const TextureID *layer*)

6.25.6 Member Data Documentation

6.25.6.1 TTF_Font* mediawrap::VideoContext::font [private]

6.25.6.2 SDL_Renderer* mediawrap::VideoContext::renderer [private]

6.25.6.3 std::unordered_map<TextureID, SDL_Texture*>* mediawrap::VideoContext::textures [private]

6.25.6.4 VideoDisplay* mediawrap::VideoContext::video_display [private]

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

- VideoContext.hpp
- VideoContext.cpp

6.26 mediawrap::VideoDisplay Class Reference

Creates a window and initializes SDL2 and SDL2_IMG.

`#include <VideoDisplay.hpp>`

Public Member Functions

- VideoDisplay (const std::string &title, unsigned int width, unsigned int height)

  Attempts to init SDL2 and SDL2_IMG and create a window.
- ~VideoDisplay (void)

  Destroys the window and renderer.
- SDL_Renderer * get_renderer (void)

  Creates a renderer attached to this window.

Private Attributes

- SDL_Window * window

6.26.1 Detailed Description

Creates a window and initializes SDL2 and SDL2_IMG.

Must be destroyed after use.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 mediawrap::VideoDisplay::VideoDisplay (const std::string & *title*, unsigned int *width*, unsigned int *height*)

Attempts to init SDL2 and SDL2_IMG and create a window.

Throws runtime_error if unable to set up any of these.

Parameters

title	The title to display at the top of the window.
width	The width of the window created.
height	The height of the window created.

6.26.2.2 mediawrap::VideoDisplay::~VideoDisplay (void)

Destroys the window and renderer.

Uninitializes SDL and SDL_Image.

6.26.3 Member Function Documentation

6.26.3.1 SDL_Renderer * mediawrap::VideoDisplay::get_renderer (void)

Creates a renderer attached to this window.

Must be deleted after use.

Returns

An SDL2 renderer for this window.

6.26.4 Member Data Documentation

6.26.4.1 SDL_Window* mediawrap::VideoDisplay::window [private]

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

- VideoDisplay.hpp
- VideoDisplay.cpp

6.27 teamusa::VideoEngine Class Reference

Provides video capabilities that are specific to Legend of the Great Unwashed.

`#include <VideoEngine.hpp>`

Public Member Functions

- VideoEngine (const std::string &title, unsigned int width, unsigned int height)

  Creates a new window that provides basic 2D drawing capabilities.
- ~VideoEngine ()

  Destroys the video engine after freeing all associated textures.
- void loadTexture (const std::string &path, TextureID id, ResourceGroup group)

  Loads the image file from the given path, transforms it into a surface, and pushes it onto the graphics card as a texture.
- void render (const Region &region, const unsigned int layer, const TextureID id)

  Renders the texture onto the given layer in the given region.
- void renderDebugBox (const Region &region, const VideoContext::DebugColor=VideoContext::Debug↩ Color::BLUE)
- void renderRotate (Region &region, unsigned int layer, TextureID id, float angle=0.0)

  Renders the texture onto the given layer in the given region with the given rotation angle.
- bool isShowingTextbox ()

  States whether a textbox is currently being displayed or not.
- void showTextbox (const std::string &text)

  Displays the given text in a textbox.
- void hideTextbox ()

  Clears the current textbox so it does not appear.
- void deleteTexture (TextureID id)

  Removes the current texture from graphics memory.
- void deleteResourceGroup (ResourceGroup resourceGroup)

  Deletes all textures associated with the given resource group.
- void display ()

  Displays all rendered textures on screen.

Private Member Functions

- void clearLayers ()

  Clears all layers with the default clear color.

Private Attributes

- bool textboxActive
- TextureID layers [NUM_LAYERS]
- std::vector< TextureID > coreResources
- std::vector< TextureID > levelResources
- VideoContext * videoContext
- Region textboxPadding
- Region textboxRegion

Static Private Attributes

- static const unsigned int NUM_LAYERS = 7
- static const unsigned int SHADOW_LAYER = 4
- static const TextureID TEXT_LAYER = 8
- static const TextureID MAX_RESERVED_ID = 1000

6.27.1 Detailed Description

Provides video capabilities that are specific to Legend of the Great Unwashed.

Utilizes VideoContext to perform rendering.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 VideoEngine::VideoEngine (const std::string & *title*, unsigned int *width*, unsigned int *height*)

Creates a new window that provides basic 2D drawing capabilities.

Parameters

title	The title to be displayed at the top of the window.
width	The width of the window in pixels.
height	The height of the window in pixels.

6.27.2.2 VideoEngine::~VideoEngine ( )

Destroys the video engine after freeing all associated textures.

6.27.3 Member Function Documentation

6.27.3.1 void VideoEngine::clearLayers ( ) [private]

Clears all layers with the default clear color.

Does not modify the textbox layer.

6.27.3.2 void VideoEngine::deleteResourceGroup (ResourceGroup *resourceGroup*)

Deletes all textures associated with the given resource group.

Parameters

resourceGroup	The group of textures to delete from video memory.

6.27.3.3 void VideoEngine::deleteTexture (TextureID *id*)

Removes the current texture from graphics memory.

Parameters

id	The id of the texture to delete.

6.27.3.4 void VideoEngine::display (void)

Displays all rendered textures on screen.

6.27.3.5 void VideoEngine::hideTextbox ( )

Clears the current textbox so it does not appear.

6.27.3.6 bool VideoEngine::isShowingTextbox ( )

States whether a textbox is currently being displayed or not.

Returns

The status of the textbox.

6.27.3.7 void VideoEngine::loadTexture (const std::string & *path,* TextureID *id,* ResourceGroup *group*)

Loads the image file from the given path, transforms it into a surface, and pushes it onto the graphics card as a texture.

Parameters

path	The relative location of the image to load.
id	The id to assign to the loaded texture.
resGroup	The group to load the resource into.

6.27.3.8 void VideoEngine::render (const Region & *region,* const unsigned int *layer,* const TextureID *id*)

Renders the texture onto the given layer in the given region.

Parameters

region	The region to draw the texture into.
layer	The layer to render the image onto (0-6) are valid.
id	The id of the texture to draw.

6.27.3.9 void VideoEngine::renderDebugBox (const Region & *region,* const VideoContext::DebugColor *color =* `VideoContext::DebugColor::BLUE`)

6.27.3.10 void VideoEngine::renderRotate (Region & *region,* unsigned int *layer,* TextureID *id,* float *angle =* 0.0)

Renders the texture onto the given layer in the given region with the given rotation angle.

Parameters

region	The region to draw the texture into.
layer	The layer to render the image onto (0-6) are valid.
id	The id of the texture to draw.
angle	The angle in degrees to rotate the image. Defaults to 0.

6.27.3.11 void VideoEngine::showTextbox (const std::string & *text*)

Displays the given text in a textbox.

Parameters

text	The text to display on screen.

6.27.4 Member Data Documentation

6.27.4.1 std::vector<TextureID> teamusa::VideoEngine::coreResources [private]

6.27.4.2 TextureID teamusa::VideoEngine::layers[NUM_LAYERS] [private]

6.27.4.3 std::vector<TextureID> teamusa::VideoEngine::levelResources [private]

6.27.4.4 const TextureID teamusa::VideoEngine::MAX_RESERVED_ID = 1000 [static], [private]

6.27.4.5 const unsigned int teamusa::VideoEngine::NUM_LAYERS = 7 [static], [private]

6.27.4.6 const unsigned int teamusa::VideoEngine::SHADOW_LAYER = 4 [static], [private]

6.27.4.7 const TextureID teamusa::VideoEngine::TEXT_LAYER = 8 [static], [private]

6.27.4.8 bool teamusa::VideoEngine::textboxActive [private]

6.27.4.9 Region teamusa::VideoEngine::textboxPadding [private]

6.27.4.10 Region teamusa::VideoEngine::textboxRegion [private]

6.27.4.11 VideoContext* teamusa::VideoEngine::videoContext [private]

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

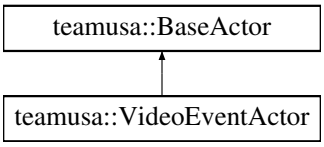
- VideoEngine.hpp
- VideoEngine.cpp

6.28 teamusa::VideoEventActor Class Reference

Will display a texture and perform no action until clicked.

`#include <VideoEventActor.h>`

Inheritance diagram for teamusa::VideoEventActor:



Public Member Functions

- VideoEventActor (Region region, int textureID, ActorEventType eventType, int eventValue, int layer)
- virtual ~VideoEventActor (void) override
- virtual const ActorEvent onClick (Player &player) override

  The onClick method will return the ActorEventType and value given to this actor during construction time.
- virtual const ActorEvent onHover (Player &player) override

  The onHover method will set the player's cursor style to CURSOR_SELECT.
- virtual const ActorEvent step (Player &player)

  Override.

Private Attributes

- ActorEvent actorEvent

Additional Inherited Members

6.28.1 Detailed Description

Will display a texture and perform no action until clicked.

The TextureID can be set to an invalid value during construction if no image needs to be displayed.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 VideoEventActor::VideoEventActor (Region *region,* int *textureID,* ActorEventType *eventType,* int *eventValue,* int *layer*) [explicit]

6.28.2.2 VideoEventActor::~VideoEventActor (void) [override], [virtual]

6.28.3 Member Function Documentation

6.28.3.1 const ActorEvent VideoEventActor::onClick (Player & *player*) [override], [virtual]

The onClick method will return the ActorEventType and value given to this actor during construction time.

Can be used to cause LoadGame and SaveGame events.

Reimplemented from teamusa::BaseActor.

6.28.3.2 const ActorEvent VideoEventActor::onHover (Player & *player*) [override], [virtual]

The onHover method will set the player's cursor style to CURSOR_SELECT.

Reimplemented from teamusa::BaseActor.

6.28.3.3 const ActorEvent VideoEventActor::step (Player & *player*) [virtual]

Override.

Implements teamusa::BaseActor.

6.28.4 Member Data Documentation

6.28.4.1 ActorEvent teamusa::VideoEventActor::actorEvent [private]

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

- VideoEventActor.h
- VideoEventActor.cpp

Chapter 7

File Documentation

7.1 ActorEvent.h File Reference

Declares ActorEvent struct.

```
#include "Headers.h"
```

Classes

- class teamusa::ActorEvent

  Event data generated by Actors, handled by Engine.

Namespaces

- teamusa

Enumerations

- enum teamusa::ActorEventType {
  teamusa::Nil = -1, teamusa::ChangeScene, teamusa::LoadLevel, teamusa::PlayAudio,
  teamusa::NewGame, teamusa::LoadGame, teamusa::DisplayText, teamusa::ExitGame,
  teamusa::StreamAudio }

7.1.1 Detailed Description

Declares ActorEvent struct.

7.2 Assert.h File Reference

Declares custom Assert macro.

Namespaces

- teamusa

Macros

- #define Assert(exp) ;

7.2.1 Detailed Description

Declares custom Assert macro.

7.2.2 Macro Definition Documentation

7.2.2.1 #define Assert(*exp*) ;

7.3 AudioEngine.cpp File Reference

```
#include "AudioEngine.hpp"
```

7.4 AudioEngine.hpp File Reference

```
#include <string>
#include <vector>
#include "AudioPlayer.hpp"
#include "Engine/ResourceGroup.hpp"
```

Classes

- class teamusa::AudioEngine

  Provides project-specific functionality for Legend of the Great Unwashed.

Namespaces

- teamusa

Typedefs

- typedef mediawrap::AudioPlayer::AudioID teamusa::AudioID

7.5 AudioPlayer.cpp File Reference

```
#include "AudioPlayer.hpp"
```

7.6 AudioPlayer.hpp File Reference

```
#include <stdexcept>
#include <string>
#include <unordered_map>
#include "SDL2/SDL.h"
#include "SDL2/SDL_mixer.h"
```

Classes

- class mediawrap::AudioPlayer

  Provides basic audio playing capabilities with WAV files.

Namespaces

- mediawrap

7.7 AudioStreamActor.cpp File Reference

Implements AudioStreamActor class.

```
#include "AudioStreamActor.h"
```

7.7.1 Detailed Description

Implements AudioStreamActor class.

7.8 AudioStreamActor.h File Reference

Declares AudioStreamActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::AudioStreamActor

  If this actor is not activated, it will emit a StreamAudio event and set its status to activated when the step method is called.

Namespaces

- teamusa

7.8.1 Detailed Description

Declares AudioStreamActor class.

7.9 BaseActor.cpp File Reference

```
#include "BaseActor.h"
#include "Engine/Assert.h"
#include "Engine/Point.h"
```

7.10 BaseActor.h File Reference

Implements BaseActor class.

```
#include "ActorEvent.h"
#include "Audio/AudioEngine.hpp"
#include "Video/VideoEngine.hpp"
```

Classes

- struct teamusa::ActorVideo
- class teamusa::BaseActor

  Abstract class which all actors must derive from.

Namespaces

- teamusa

7.10.1 Detailed Description

Implements BaseActor class.

Declares BaseActor class.

7.11 CursorStyle.h File Reference

Declares CursorStyle enumerations.

Namespaces

- teamusa

Enumerations

- enum teamusa::CursorStyle { teamusa::CursorStyle::CURSOR_DEFAULT, teamusa::CursorStyle::CURSOR_SELECT, teamusa::CursorStyle::CURSOR_LEFT, teamusa::CursorStyle::CURSOR_RIGHT, teamusa::CursorStyle::CURSOR_UP, teamusa::CursorStyle::CURSOR_DOWN }

  The possible styles for the mouse cursor.

7.11.1 Detailed Description

Declares CursorStyle enumerations.

7.12 DelayedAudioActor.cpp File Reference

Implements DelayedAudioActor class.

```
#include "DelayedAudioActor.h"
```

7.12.1 Detailed Description

Implements DelayedAudioActor class.

7.13 DelayedAudioActor.h File Reference

Declares DelayedAudioActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::DelayedAudioActor

  :Will increment a counter every time the step method is called.

Namespaces

- teamusa

7.13.1 Detailed Description

Declares DelayedAudioActor class.

7.14 DelayedVideoActor.cpp File Reference

Declares DelayedVideoActor class.

```
#include "DelayedVideoActor.h"
#include <iostream>
```

7.14.1 Detailed Description

Declares DelayedVideoActor class.

7.15 DelayedVideoActor.h File Reference

Declares DelayedVideoActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::DelayedVideoActor

  Will increment a counter every time the step method is called.

Namespaces

- teamusa

7.15.1 Detailed Description

Declares DelayedVideoActor class.

7.16 Engine.cpp File Reference

Implements Engine class.

```
#include "Engine.h"
#include "Actor/AudioStreamActor.h"
#include "Actor/SceneLink.h"
#include "Actor/TextboxSpawnActor.h"
#include "Audio/AudioEngine.hpp"
#include "Engine/Assert.h"
#include "Engine/ResourceGroup.hpp"
#include "Engine/Timer.h"
#include "Video/VideoEngine.hpp"
```

Macros

- #define BIND(function) (std::bind(function, this, std::placeholders::_1, std::placeholders::_2))

Variables

- static const double FRAME_TIME = 16.67

7.16.1 Detailed Description

Implements Engine class.

7.16.2 Macro Definition Documentation

7.16.2.1 #define BIND(*function*) (std::bind(function, this, std::placeholders::_1, std::placeholders::_2))

7.16.3 Variable Documentation

7.16.3.1 const double FRAME_TIME = 16.67 [static]

7.17 Engine.h File Reference

Declares Engine class.

```
#include "Headers.h"
#include "Engine/Level.h"
#include "GameSaveSerializer/GameSaveSerializer.h"
#include "Player/Player.h"
```

Classes

- class teamusa::Engine

  Processes all components of the game each frame.

Namespaces

- teamusa

7.17.1 Detailed Description

Declares Engine class.

7.18 ExampleActor.cpp File Reference

Implements ExampleActor class.

```
#include "ExampleActor.h"
```

7.18.1 Detailed Description

Implements ExampleActor class.

7.19 ExampleActor.h File Reference

Declares ExampleActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::ExampleActor

  <Give brief="" description="" here>=""></

Namespaces

- teamusa

7.19.1 Detailed Description

Declares ExampleActor class.

7.20 GameSaveSerializer.cpp File Reference

```
#include "GameSaveSerializer.h"
#include "Engine/Assert.h"
```

Namespaces

- teamusa

7.21 GameSaveSerializer.h File Reference

Declares save file serializer class.

```
#include <vector>
#include <fstream>
#include <mutex>
#include <string>
#include <thread>
#include "Player/Player.h"
```

Classes

- class teamusa::GameSaveSerializer

  Provides multithreaded save, single-thread load of save files.

Namespaces

- teamusa

7.21.1 Detailed Description

Declares save file serializer class.

7.22 Headers.h File Reference

Easy way to include all headers needed.

```
#include <exception>
#include <fstream>
#include <functional>
#include <iostream>
#include <map>
#include <memory>
#include <stack>
#include <string>
#include <vector>
#include <stdint.h>
```

7.22.1 Detailed Description

Easy way to include all headers needed.

7.23 InventoryItemActor.cpp File Reference

Implements InventoryItemActor class.

```
#include "InventoryItemActor.h"
#include "Player/Player.h"
```

7.23.1 Detailed Description

Implements InventoryItemActor class.

7.24 InventoryItemActor.h File Reference

Declares InventoryItemActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::InventoryItemActor

  <Give brief="" description="" here>=""

Namespaces

- teamusa

7.24.1 Detailed Description

Declares InventoryItemActor class.

7.25 Level.cpp File Reference

Implements Level class.

```
#include "Assert.h"
#include "Level.h"
#include "Actor/ActorEvent.h"
#include "Actor/AudioStreamActor.h"
#include "Actor/DelayedAudioActor.h"
#include "Actor/DelayedVideoActor.h"
#include "Actor/InventoryItemActor.h"
#include "Actor/LevelLink.h"
#include "Actor/MovingActor.h"
#include "Actor/ResponsiveAudioActor.h"
#include "Actor/ResponsiveVideoActor.h"
#include "Actor/SceneLink.h"
#include "Actor/TextboxSpawnActor.h"
#include "Actor/VideoActor.h"
#include "Actor/VideoEventActor.h"
#include "Audio/AudioEngine.hpp"
#include "Video/VideoEngine.hpp"
```

Functions

- static std::istream & operator>> (std::istream &fs, Region &dst)
- static std::istream & operator>> (std::istream &fs, ActorEventType &dst)
- static void loadError (const std::string &msg)

7.25.1 Detailed Description

Implements Level class.

7.25.2 Function Documentation

7.25.2.1 static void loadError (const std::string & *msg*) [static]

7.25.2.2 static std::istream& operator>> (std::istream & *fs,* Region & *dst*) [inline], [static]

7.25.2.3 static std::istream& operator>> (std::istream & *fs,* ActorEventType & *dst*) [inline], [static]

7.26 Level.h File Reference

Declares Level class.

```
#include <unordered_map>
#include "Headers.h"
```

Classes

- class teamusa::Level
- struct teamusa::Level::Scene

Namespaces

- teamusa

Typedefs

- typedef std::shared_ptr< BaseActor > teamusa::BaseActorPtr
- typedef std::vector< BaseActorPtr > teamusa::ActorList

7.26.1 Detailed Description

Declares Level class.

7.27 LevelLink.cpp File Reference

Implements LevelLink class.

```
#include "LevelLink.h"
#include "Player/Player.h"
```

7.27.1 Detailed Description

Implements LevelLink class.

7.28 LevelLink.h File Reference

Declares LevelLink class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::LevelLink

Namespaces

- teamusa

7.28.1 Detailed Description

Declares LevelLink class.

7.29 main.cpp File Reference

Entry point of program.

```
#include "Headers.h"
#include "Engine/Engine.h"
```

Namespaces

- MainNS

Functions

- static void MainNS::logError (const std::string &desc)
- int main (int argc, char **argv)

7.29.1 Detailed Description

Entry point of program.

7.29.2 Function Documentation

7.29.2.1 int main (int *argc*, char ** *argv*)

7.30 MovingActor.cpp File Reference

```
#include "MovingActor.h"
#include "Player/Player.h"
```

7.31 MovingActor.h File Reference

```
#include "BaseActor.h"
```

Classes

- class teamusa::MovingActor

  Will transition from one region to the next by calculating the distance to move each frame for a set number of frames.

Namespaces

- teamusa

7.32 Player.cpp File Reference

Implements Player class.

```
#include "Player.h"
#include "Engine/Assert.h"
```

7.32.1 Detailed Description

Implements Player class.

7.33 Player.h File Reference

Declares Player class.

```
#include "Headers.h"
#include "CursorStyle.h"
#include "Engine/Point.h"
#include "Video/VideoEngine.hpp"
```

Classes

- class teamusa::Player

  Handles all data relevant to the player engaging the game.

Namespaces

- teamusa

7.33.1 Detailed Description

Declares Player class.

7.34 Point.h File Reference

Declares Point struct.

```
#include <stdint.h>
```

Classes

- struct teamusa::Point

Namespaces

- teamusa

7.34.1 Detailed Description

Declares Point struct.

7.35 ResourceGroup.hpp File Reference

Enumerations

- enum ResourceGroup { CORE_RESOURCE, LEVEL_RESOURCE }

7.35.1 Enumeration Type Documentation

7.35.1.1 enum ResourceGroup

Enumerator

CORE_RESOURCE

LEVEL_RESOURCE

7.36 ResponsiveAudioActor.cpp File Reference

Implements ResponsiveAudioActor class.

```
#include "ResponsiveAudioActor.h"
```

7.36.1 Detailed Description

Implements ResponsiveAudioActor class.

7.37 ResponsiveAudioActor.h File Reference

Declares ResponsiveAudioActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::ResponsiveAudioActor

  \ *Brief: Will increment the value of stepCount until it is equal to durationSteps for each call to the step method.*

Namespaces

- teamusa

7.37.1 Detailed Description

Declares ResponsiveAudioActor class.

7.38 ResponsiveVideoActor.cpp File Reference

Will display the default TextureID.

```
#include "ResponsiveVideoActor.h"
```

7.38.1 Detailed Description

Will display the default TextureID.

7.39 ResponsiveVideoActor.h File Reference

Declares ResponsivevideoActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::ResponsiveVideoActor

Namespaces

- teamusa

7.39.1 Detailed Description

Declares ResponsivevideoActor class.

7.40 SceneLink.cpp File Reference

Implements SceneLink class.

```
#include "SceneLink.h"
#include "Player/Player.h"
```

7.40.1 Detailed Description

Implements SceneLink class.

7.41 SceneLink.h File Reference

Declares SceneLink class.

```
#include "BaseActor.h"
#include "Player/CursorStyle.h"
```

Classes

- class teamusa::SceneLink

Namespaces

- teamusa

7.41.1 Detailed Description

Declares SceneLink class.

7.42 TextboxSpawnActor.cpp File Reference

Declares TextboxSpawnActor class.

```
#include "TextboxSpawnActor.h"
```

7.42.1 Detailed Description

Declares TextboxSpawnActor class.

7.43 TextboxSpawnActor.h File Reference

Declares TextboxSpawnActor class.

```
#include "BaseActor.h"
#include <string>
```

Classes

- class teamusa::TextboxSpawnActor

  <Give brief="" description="" here>=""

Namespaces

- teamusa

7.43.1 Detailed Description

Declares TextboxSpawnActor class.

7.44 Timer.cpp File Reference

Implements Timer class.

```
#include "Engine/Timer.h"
#include <SDL2/SDL.h>
```

7.44.1 Detailed Description

Implements Timer class.

7.45 Timer.h File Reference

Declares Timer class.

```
#include "Headers.h"
```

Classes

- class teamusa::Timer

  A timer that counts up from zero in milliseconds.

Namespaces

- teamusa

7.45.1 Detailed Description

Declares Timer class.

7.46 VideoActor.cpp File Reference

Implements VideoActor class.

```
#include "VideoActor.h"
```

7.46.1 Detailed Description

Implements VideoActor class.

7.47 VideoActor.h File Reference

This module makes sure An actor that will only display a texture at a given region.

```
#include "BaseActor.h"
```

Classes

- class teamusa::VideoActor

Namespaces

- teamusa

7.47.1 Detailed Description

This module makes sure An actor that will only display a texture at a given region.

This actor will have no interation with the player.

7.48 VideoContext.cpp File Reference

```
#include "VideoContext.hpp"
```

7.49 VideoContext.hpp File Reference

```
#include <unordered_map>
#include <string>
#include "SDL2/SDL.h"
#include "SDL2/SDL_image.h"
#include "SDL2/SDL_ttf.h"
#include "VideoDisplay.hpp"
```

Classes

- class mediawrap::VideoContext

  Provides basic 2D rendering capabilities.

Namespaces

- mediawrap

7.50 VideoDisplay.cpp File Reference

```
#include "VideoDisplay.hpp"
```

7.51 VideoDisplay.hpp File Reference

```
#include <stdexcept>
#include "SDL2/SDL.h"
#include "SDL2/SDL_image.h"
#include "SDL2/SDL_ttf.h"
```

Classes

- class mediawrap::VideoDisplay

  Creates a window and initializes SDL2 and SDL2_IMG.

Namespaces

- mediawrap

7.52 VideoEngine.cpp File Reference

```
#include "VideoEngine.hpp"
```

7.53 VideoEngine.hpp File Reference

```
#include <stdexcept>
#include <string>
#include <vector>
#include "VideoContext.hpp"
#include "Engine/ResourceGroup.hpp"
```

Classes

- class teamusa::VideoEngine

  Provides video capabilities that are specific to Legend of the Great Unwashed.

Namespaces

- teamusa

Typedefs

- typedef mediawrap::VideoContext::TextureID teamusa::TextureID
- typedef mediawrap::VideoContext::Region teamusa::Region

7.54 VideoEventActor.cpp File Reference

declares VideoEventActor class

```
#include "VideoEventActor.h"
#include "Player/Player.h"
```

7.54.1 Detailed Description

declares VideoEventActor class

7.55 VideoEventActor.h File Reference

Declares VideoEventActor class.

```
#include "BaseActor.h"
```

Classes

- class teamusa::VideoEventActor

  Will display a texture and perform no action until clicked.

Namespaces

- teamusa

7.55.1 Detailed Description

Declares VideoEventActor class.

Index

~AudioPlayer
    mediawrap::AudioPlayer, 17
~AudioStreamActor
    teamusa::AudioStreamActor, 19
~BaseActor
    teamusa::BaseActor, 21
~DelayedAudioActor
    teamusa::DelayedAudioActor, 25
~DelayedVideoActor
    teamusa::DelayedVideoActor, 26
~Engine
    teamusa::Engine, 28
~ExampleActor
    teamusa::ExampleActor, 30
~GameSaveSerializer
    teamusa::GameSaveSerializer, 31
~InventoryItemActor
    teamusa::InventoryItemActor, 32
~LevelLink
    teamusa::LevelLink, 36
~MovingActor
    teamusa::MovingActor, 38
~Player
    teamusa::Player, 40
~ResponsiveAudioActor
    teamusa::ResponsiveAudioActor, 43
~ResponsiveVideoActor
    teamusa::ResponsiveVideoActor, 44
~SceneLink
    teamusa::SceneLink, 46
~TextboxSpawnActor
    teamusa::TextboxSpawnActor, 48
~Timer
    teamusa::Timer, 49
~VideoActor
    teamusa::VideoActor, 51
~VideoContext
    mediawrap::VideoContext, 55
~VideoDisplay
    mediawrap::VideoDisplay, 58
~VideoEngine
    teamusa::VideoEngine, 60
~VideoEventActor
    teamusa::VideoEventActor, 65

activated
    teamusa::AudioStreamActor, 19
    teamusa::TextboxSpawnActor, 49
activeScene
    teamusa::Level, 35
ActorEvent
    teamusa::ActorEvent, 13
actorEvent
    teamusa::VideoEventActor, 66
ActorEvent.h, 67
ActorEventHandler
    teamusa::Engine, 28
ActorEventType
    teamusa, 11
ActorList
    teamusa, 11
ActorVideo
    teamusa::ActorVideo, 14
actors
    teamusa::Level::Scene, 45
addItem
    teamusa::Player, 40
Assert
    Assert.h, 68
Assert.h, 67
    Assert, 68
audio_buffer
    mediawrap::AudioPlayer, 18
audio_channels
    mediawrap::AudioPlayer, 18
audio_format
    mediawrap::AudioPlayer, 18
audio_rate
    mediawrap::AudioPlayer, 18
audio_samples
    mediawrap::AudioPlayer, 18
audio_stream
    mediawrap::AudioPlayer, 18
AudioEngine.cpp, 68
AudioEngine.hpp, 68
AudioID
    mediawrap::AudioPlayer, 17
    teamusa, 11
audioId
    teamusa::DelayedAudioActor, 25
AudioPlayer
    mediawrap::AudioPlayer, 17
audioPlayer
    teamusa::AudioEngine, 15
AudioPlayer.cpp, 68
AudioPlayer.hpp, 69
AudioStreamActor
    teamusa::AudioStreamActor, 19
AudioStreamActor.cpp, 69

AudioStreamActor.h, 69

BIND
    Engine.cpp, 72
BLENDMODE_ADD
    mediawrap::VideoContext, 53
BLENDMODE_BLEND
    mediawrap::VideoContext, 53
BLENDMODE_MOD
    mediawrap::VideoContext, 53
BLENDMODE_NONE
    mediawrap::VideoContext, 53
BLUE
    mediawrap::VideoContext, 53
BaseActor
    teamusa::BaseActor, 21
BaseActor.cpp, 70
BaseActor.h, 70
BaseActorPtr
    teamusa, 11
bgImageID
    teamusa::Level::Scene, 45
BlendMode
    mediawrap::VideoContext, 53

CORE_RESOURCE
    ResourceGroup.hpp, 80
CURSOR_DEFAULT
    teamusa, 11
CURSOR_DEFAULT_ID
    teamusa::Player, 41
CURSOR_DOWN
    teamusa, 11
CURSOR_DOWN_ID
    teamusa::Player, 41
CURSOR_LEFT
    teamusa, 11
CURSOR_LEFT_ID
    teamusa::Player, 41
CURSOR_RIGHT
    teamusa, 11
CURSOR_RIGHT_ID
    teamusa::Player, 41
CURSOR_SELECT
    teamusa, 11
CURSOR_SELECT_ID
    teamusa::Player, 41
CURSOR_UP
    teamusa, 11
CURSOR_UP_ID
    teamusa::Player, 41
ChangeScene
    teamusa, 11
changeScene
    teamusa::Level, 34
clear_samples
    mediawrap::AudioPlayer, 17
clearAll
    teamusa::Level, 34
clearLayers
    teamusa::VideoEngine, 60
clickAudioId
    teamusa::ResponsiveAudioActor, 44
clickTexture
    teamusa::ResponsiveVideoActor, 45
coreResources
    teamusa::AudioEngine, 15
    teamusa::VideoEngine, 64
create_texture
    mediawrap::VideoContext, 55
currentStep
    teamusa::DelayedAudioActor, 25
    teamusa::DelayedVideoActor, 26
    teamusa::MovingActor, 38
CursorStyle
    teamusa, 11
cursorStyle
    teamusa::SceneLink, 47
CursorStyle.h, 70

DebugColor
    mediawrap::VideoContext, 53
defaultTextureId
    teamusa::ResponsiveVideoActor, 45
delaySteps
    teamusa::DelayedAudioActor, 25
    teamusa::DelayedVideoActor, 26
DelayedAudioActor
    teamusa::DelayedAudioActor, 25
DelayedAudioActor.cpp, 71
DelayedAudioActor.h, 71
DelayedVideoActor
    teamusa::DelayedVideoActor, 26
DelayedVideoActor.cpp, 71
DelayedVideoActor.h, 71
delete_sample
    mediawrap::AudioPlayer, 17
delete_texture
    mediawrap::VideoContext, 55
deleteResourceGroup
    teamusa::VideoEngine, 60
deleteSound
    teamusa::AudioEngine, 15
deleteSoundGroup
    teamusa::AudioEngine, 15
deleteTexture
    teamusa::VideoEngine, 62
disappear
    teamusa::DelayedVideoActor, 26
display
    mediawrap::VideoContext, 55
    teamusa::VideoEngine, 62
DisplayText
    teamusa, 11

endRegion
    teamusa::MovingActor, 38
Engine

teamusa::Engine, 28
Engine.cpp, 72
BIND, 72
FRAME_TIME, 72
Engine.h, 72
ExampleActor
teamusa::ExampleActor, 30
ExampleActor.cpp, 73
ExampleActor.h, 73
ExitGame
teamusa, 11

FLASHLIGHT_ID
teamusa::Player, 41
FLIP_HORIZONTAL
mediawrap::VideoContext, 53
FLIP_NONE
mediawrap::VideoContext, 53
FLIP_VERTICAL
mediawrap::VideoContext, 53
FRAME_TIME
Engine.cpp, 72
fileLock
teamusa::GameSaveSerializer, 31
fill_texture
mediawrap::VideoContext, 55
Flip
mediawrap::VideoContext, 53
font
mediawrap::VideoContext, 57
freeAndLoadLevel
teamusa::Engine, 28

GREEN
mediawrap::VideoContext, 53
GameSaveSerializer
teamusa::GameSaveSerializer, 31
GameSaveSerializer.cpp, 74
GameSaveSerializer.h, 74
get_renderer
mediawrap::VideoDisplay, 59
getActors
teamusa::Level, 34
getBGImageID
teamusa::Level, 34
getCursorTextureID
teamusa::Player, 40
getInventory
teamusa::Player, 40
getLayer
teamusa::BaseActor, 21
getMouseClickState
teamusa::Engine, 28
getMouseCoordinates
teamusa::Engine, 28
getPath
teamusa::AudioStreamActor, 19
getPosition
teamusa::Player, 40
getRegion
teamusa::BaseActor, 21
getScene
teamusa::Level, 34
getSceneID
teamusa::LevelLink, 36
getText
teamusa::LevelLink, 36
teamusa::SceneLink, 46
teamusa::TextboxSpawnActor, 48
getTextureID
teamusa::BaseActor, 21
getTicks
teamusa::Timer, 49

hGrowth
teamusa::MovingActor, 38
handleEvent
teamusa::Engine, 28
hasItem
teamusa::Player, 40
hasVideo
teamusa::BaseActor, 21
Headers.h, 74
hideTextbox
teamusa::VideoEngine, 62
hoverAudioId
teamusa::ResponsiveAudioActor, 44
hoverTexture
teamusa::ResponsiveVideoActor, 45

Inventory
teamusa::Player, 40
InventoryItemActor
teamusa::InventoryItemActor, 32
InventoryItemActor.cpp, 75
InventoryItemActor.h, 75
isActive
teamusa::MovingActor, 38
isInBounds
teamusa::BaseActor, 21
isShowingTextbox
teamusa::VideoEngine, 62
itemID
teamusa::InventoryItemActor, 33
itemRequiredText
teamusa::LevelLink, 36
teamusa::SceneLink, 47

LEVEL_RESOURCE
ResourceGroup.hpp, 80
layer
teamusa::ActorVideo, 14
layers
teamusa::VideoEngine, 64
Level
teamusa::Level, 34
Level.cpp, 75
loadError, 76

operator>>, 76
Level.h, 76
levelID
teamusa::LevelLink, 36
LevelLink
teamusa::LevelLink, 36
LevelLink.cpp, 77
LevelLink.h, 77
levelResources
teamusa::AudioEngine, 15
teamusa::VideoEngine, 64
load
teamusa::GameSaveSerializer, 31
load_font
mediawrap::VideoContext, 56
load_sample
mediawrap::AudioPlayer, 17
load_stream
mediawrap::AudioPlayer, 17
load_texture
mediawrap::VideoContext, 56
loadError
Level.cpp, 76
LoadGame
teamusa, 11
LoadLevel
teamusa, 11
loadLevel
teamusa::Level, 34
loadSound
teamusa::AudioEngine, 15
loadTexture
teamusa::VideoEngine, 62
logError
MainNS, 9

MAX_RESERVED_ID
teamusa::AudioEngine, 15
teamusa::VideoEngine, 64
mActorEventHandlers
teamusa::Engine, 29
mAudioEngine
teamusa::Engine, 29
mAudioID
teamusa::BaseActor, 24
mCursorStyle
teamusa::Player, 41
mInventory
teamusa::Player, 41
mIsRunning
teamusa::Engine, 29
mLayer
teamusa::Player, 41
mLevel
teamusa::Engine, 29
MOUSE_CLICK_ID
teamusa::Player, 41
mPauseTicks
teamusa::Timer, 50
mPaused
teamusa::Timer, 50
mPlayer
teamusa::Engine, 29
mPosition
teamusa::Player, 41
mRegion
teamusa::BaseActor, 24
teamusa::Player, 41
mSerializer
teamusa::Engine, 29
mStartTicks
teamusa::Timer, 50
mStarted
teamusa::Timer, 50
mTextureID
teamusa::Player, 42
mVideo
teamusa::BaseActor, 24
mVideoEngine
teamusa::Engine, 29
main
main.cpp, 78
main.cpp, 77
main, 78
MainNS, 9
logError, 9
mediawrap, 9
mediawrap::AudioPlayer, 16
~AudioPlayer, 17
audio_buffer, 18
audio_channels, 18
audio_format, 18
audio_rate, 18
audio_samples, 18
audio_stream, 18
AudioID, 17
AudioPlayer, 17
clear_samples, 17
delete_sample, 17
load_sample, 17
load_stream, 17
play_sample, 18
stream_audio, 18
mediawrap::VideoContext, 51
~VideoContext, 55
BLENDMODE_ADD, 53
BLENDMODE_BLEND, 53
BLENDMODE_MOD, 53
BLENDMODE_NONE, 53
BLUE, 53
BlendMode, 53
create_texture, 55
DebugColor, 53
delete_texture, 55
display, 55
FLIP_HORIZONTAL, 53
FLIP_NONE, 53

FLIP_VERTICAL, 53
fill_texture, 55
Flip, 53
font, 57
GREEN, 53
load_font, 56
load_texture, 56
RED, 53
Region, 53
render, 56
render_clear, 56
render_onto, 57
render_rotate, 57
render_text, 57
renderDebugBox, 57
renderer, 57
texture_iter, 53
TextureID, 53
textures, 57
video_display, 58
VideoContext, 54
mediawrap::VideoDisplay, 58
~VideoDisplay, 58
get_renderer, 59
VideoDisplay, 58
window, 59
MovingActor
teamusa::MovingActor, 38
MovingActor.cpp, 78
MovingActor.h, 78

NUM_LAYERS
teamusa::VideoEngine, 64
NewGame
teamusa, 11
Nil
teamusa, 11

onChangeScene
teamusa::Engine, 28
onClick
teamusa::BaseActor, 23
teamusa::ExampleActor, 30
teamusa::InventoryItemActor, 32
teamusa::LevelLink, 36
teamusa::MovingActor, 38
teamusa::ResponsiveAudioActor, 43
teamusa::ResponsiveVideoActor, 45
teamusa::SceneLink, 46
teamusa::TextboxSpawnActor, 48
teamusa::VideoEventActor, 65
onDisplayText
teamusa::Engine, 28
onExitGame
teamusa::Engine, 28
onHover
teamusa::BaseActor, 23
teamusa::ExampleActor, 30
teamusa::InventoryItemActor, 32
teamusa::LevelLink, 36
teamusa::MovingActor, 38
teamusa::ResponsiveAudioActor, 43
teamusa::ResponsiveVideoActor, 45
teamusa::SceneLink, 46
teamusa::VideoEventActor, 65
onLoadGame
teamusa::Engine, 28
onLoadLevel
teamusa::Engine, 28
onNewGame
teamusa::Engine, 28
onPlayAudio
teamusa::Engine, 28
onStreamAudio
teamusa::Engine, 28
operator>>
Level.cpp, 76

parseAudioStreamActor
teamusa::Level, 34
parseDelayedAudioActor
teamusa::Level, 34
parseDelayedVideoActor
teamusa::Level, 34
parseInventoryItemActor
teamusa::Level, 34
parseLevelLink
teamusa::Level, 34
parseMovingActor
teamusa::Level, 34
parseResponsiveAudioActor
teamusa::Level, 34
parseResponsiveVideoActor
teamusa::Level, 34
parseSceneLink
teamusa::Level, 35
parseTextboxSpawnActor
teamusa::Level, 35
parseVideoActor
teamusa::Level, 35
parseVideoEventActor
teamusa::Level, 35
path
teamusa::AudioStreamActor, 19
pause
teamusa::Timer, 50
pickedUp
teamusa::InventoryItemActor, 33
play_sample
mediawrap::AudioPlayer, 18
PlayAudio
teamusa, 11
playSound
teamusa::AudioEngine, 15
playStream
teamusa::AudioEngine, 15
Player
teamusa::Player, 40

Player.cpp, 78
Player.h, 79
Point
    teamusa::Point, 42
Point.h, 79

RED
    mediawrap::VideoContext, 53
Region
    mediawrap::VideoContext, 53
    teamusa, 11
render
    mediawrap::VideoContext, 56
    teamusa::Engine, 28
    teamusa::VideoEngine, 62
render_clear
    mediawrap::VideoContext, 56
render_onto
    mediawrap::VideoContext, 57
render_rotate
    mediawrap::VideoContext, 57
render_text
    mediawrap::VideoContext, 57
renderDebugBox
    mediawrap::VideoContext, 57
    teamusa::VideoEngine, 62
renderRotate
    teamusa::VideoEngine, 62
renderer
    mediawrap::VideoContext, 57
requiredItemID
    teamusa::LevelLink, 37
    teamusa::SceneLink, 47
ResourceGroup
    ResourceGroup.hpp, 80
ResourceGroup.hpp, 79
    CORE_RESOURCE, 80
    LEVEL_RESOURCE, 80
    ResourceGroup, 80
ResponsiveAudioActor
    teamusa::ResponsiveAudioActor, 43
ResponsiveAudioActor.cpp, 80
ResponsiveAudioActor.h, 80
ResponsiveVideoActor
    teamusa::ResponsiveVideoActor, 44
ResponsiveVideoActor.cpp, 80
ResponsiveVideoActor.h, 81
run
    teamusa::Engine, 28

SHADOW_LAYER
    teamusa::VideoEngine, 64
save
    teamusa::GameSaveSerializer, 31
saveInThread
    teamusa::GameSaveSerializer, 31
sceneID
    teamusa::LevelLink, 37
    teamusa::SceneLink, 47
SceneLink
    teamusa::SceneLink, 46
SceneLink.cpp, 81
SceneLink.h, 81
scenes
    teamusa::Level, 35
setCursor
    teamusa::Player, 40
setInventory
    teamusa::Player, 41
setPosition
    teamusa::Player, 41
setRegion
    teamusa::BaseActor, 23
setSlot
    teamusa::GameSaveSerializer, 31
setTextureId
    teamusa::ResponsiveVideoActor, 45
showTextbox
    teamusa::VideoEngine, 64
slot
    teamusa::GameSaveSerializer, 31
start
    teamusa::Timer, 50
startScene
    teamusa::Level, 35
step
    teamusa::AudioStreamActor, 19
    teamusa::BaseActor, 23
    teamusa::DelayedAudioActor, 25
    teamusa::DelayedVideoActor, 26
    teamusa::ExampleActor, 30
    teamusa::InventoryItemActor, 33
    teamusa::LevelLink, 36
    teamusa::MovingActor, 38
    teamusa::ResponsiveAudioActor, 43
    teamusa::ResponsiveVideoActor, 45
    teamusa::SceneLink, 47
    teamusa::TextboxSpawnActor, 48
    teamusa::VideoActor, 51
    teamusa::VideoEventActor, 65
stop
    teamusa::Timer, 50
stream_audio
    mediawrap::AudioPlayer, 18
StreamAudio
    teamusa, 11

TEXT_LAYER
    teamusa::VideoEngine, 64
teamusa, 9
    ActorEventType, 11
    ActorList, 11
    AudioID, 11
    BaseActorPtr, 11
    CURSOR_DEFAULT, 11
    CURSOR_DOWN, 11
    CURSOR_LEFT, 11
    CURSOR_RIGHT, 11

CURSOR_SELECT, 11
CURSOR_UP, 11
ChangeScene, 11
CursorStyle, 11
DisplayText, 11
ExitGame, 11
LoadGame, 11
LoadLevel, 11
NewGame, 11
Nil, 11
PlayAudio, 11
Region, 11
StreamAudio, 11
TextureID, 11
teamusa::ActorEvent, 13
ActorEvent, 13
type, 13
value, 13
teamusa::ActorVideo, 13
ActorVideo, 14
layer, 14
textureID, 14
teamusa::AudioEngine, 14
audioPlayer, 15
coreResources, 15
deleteSound, 15
deleteSoundGroup, 15
levelResources, 15
loadSound, 15
MAX_RESERVED_ID, 15
playSound, 15
playStream, 15
teamusa::AudioStreamActor, 18
∼AudioStreamActor, 19
activated, 19
AudioStreamActor, 19
getPath, 19
path, 19
step, 19
teamusa::BaseActor, 20
∼BaseActor, 21
BaseActor, 21
getLayer, 21
getRegion, 21
getTextureID, 21
hasVideo, 21
isInBounds, 21
mAudioID, 24
mRegion, 24
mVideo, 24
onClick, 23
onHover, 23
setRegion, 23
step, 23
teamusa::DelayedAudioActor, 24
∼DelayedAudioActor, 25
audioId, 25
currentStep, 25
delaySteps, 25
DelayedAudioActor, 25
step, 25
teamusa::DelayedVideoActor, 25
∼DelayedVideoActor, 26
currentStep, 26
delaySteps, 26
DelayedVideoActor, 26
disappear, 26
step, 26
textureId, 26
teamusa::Engine, 27
∼Engine, 28
ActorEventHandler, 28
Engine, 28
freeAndLoadLevel, 28
getMouseClickState, 28
getMouseCoordinates, 28
handleEvent, 28
mActorEventHandlers, 29
mAudioEngine, 29
mIsRunning, 29
mLevel, 29
mPlayer, 29
mSerializer, 29
mVideoEngine, 29
onChangeScene, 28
onDisplayText, 28
onExitGame, 28
onLoadGame, 28
onLoadLevel, 28
onNewGame, 28
onPlayAudio, 28
onStreamAudio, 28
render, 28
run, 28
teamusa::ExampleActor, 29
∼ExampleActor, 30
ExampleActor, 30
onClick, 30
onHover, 30
step, 30
teamusa::GameSaveSerializer, 30
∼GameSaveSerializer, 31
fileLock, 31
GameSaveSerializer, 31
load, 31
save, 31
saveInThread, 31
setSlot, 31
slot, 31
teamusa::InventoryItemActor, 32
∼InventoryItemActor, 32
InventoryItemActor, 32
itemID, 33
onClick, 32
onHover, 32
pickedUp, 33

step, 33
teamusa::Level, 33
activeScene, 35
changeScene, 34
clearAll, 34
getActors, 34
getBGImageID, 34
getScene, 34
Level, 34
loadLevel, 34
parseAudioStreamActor, 34
parseDelayedAudioActor, 34
parseDelayedVideoActor, 34
parseInventoryItemActor, 34
parseLevelLink, 34
parseMovingActor, 34
parseResponsiveAudioActor, 34
parseResponsiveVideoActor, 34
parseSceneLink, 35
parseTextboxSpawnActor, 35
parseVideoActor, 35
parseVideoEventActor, 35
scenes, 35
startScene, 35
teamusa::Level::Scene, 45
actors, 45
bgImageID, 45
teamusa::LevelLink, 35
∼LevelLink, 36
getSceneID, 36
getText, 36
itemRequiredText, 36
levelID, 36
LevelLink, 36
onClick, 36
onHover, 36
requiredItemID, 37
sceneID, 37
step, 36
teamusa::MovingActor, 37
∼MovingActor, 38
currentStep, 38
endRegion, 38
hGrowth, 38
isActive, 38
MovingActor, 38
onClick, 38
onHover, 38
step, 38
transitionSteps, 38
wGrowth, 38
xSpeed, 38
ySpeed, 38
teamusa::Player, 38
∼Player, 40
addItem, 40
CURSOR_DEFAULT_ID, 41
CURSOR_DOWN_ID, 41
CURSOR_LEFT_ID, 41
CURSOR_RIGHT_ID, 41
CURSOR_SELECT_ID, 41
CURSOR_UP_ID, 41
FLASHLIGHT_ID, 41
getCursorTextureID, 40
getInventory, 40
getPosition, 40
hasItem, 40
Inventory, 40
mCursorStyle, 41
mInventory, 41
mLayer, 41
MOUSE_CLICK_ID, 41
mPosition, 41
mRegion, 41
mTextureID, 42
Player, 40
setCursor, 40
setInventory, 41
setPosition, 41
teamusa::Point, 42
Point, 42
x, 42
y, 42
teamusa::ResponsiveAudioActor, 42
∼ResponsiveAudioActor, 43
clickAudioId, 44
hoverAudioId, 44
onClick, 43
onHover, 43
ResponsiveAudioActor, 43
step, 43
teamusa::ResponsiveVideoActor, 44
∼ResponsiveVideoActor, 44
clickTexture, 45
defaultTextureId, 45
hoverTexture, 45
onClick, 45
onHover, 45
ResponsiveVideoActor, 44
setTextureId, 45
step, 45
teamusa::SceneLink, 46
∼SceneLink, 46
cursorStyle, 47
getText, 46
itemRequiredText, 47
onClick, 46
onHover, 46
requiredItemID, 47
sceneID, 47
SceneLink, 46
step, 47
teamusa::TextboxSpawnActor, 47
∼TextboxSpawnActor, 48
activated, 49
getText, 48

onClick, 48
step, 48
text, 49
TextboxSpawnActor, 48
teamusa::Timer, 49
~Timer, 49
getTicks, 49
mPauseTicks, 50
mPaused, 50
mStartTicks, 50
mStarted, 50
pause, 50
start, 50
stop, 50
Timer, 49
unpause, 50
teamusa::VideoActor, 50
~VideoActor, 51
step, 51
VideoActor, 51
teamusa::VideoEngine, 59
~VideoEngine, 60
clearLayers, 60
coreResources, 64
deleteResourceGroup, 60
deleteTexture, 62
display, 62
hideTextbox, 62
isShowingTextbox, 62
layers, 64
levelResources, 64
loadTexture, 62
MAX_RESERVED_ID, 64
NUM_LAYERS, 64
render, 62
renderDebugBox, 62
renderRotate, 62
SHADOW_LAYER, 64
showTextbox, 64
TEXT_LAYER, 64
textboxActive, 64
textboxPadding, 64
textboxRegion, 64
videoContext, 64
VideoEngine, 60
teamusa::VideoEventActor, 64
~VideoEventActor, 65
actorEvent, 66
onClick, 65
onHover, 65
step, 65
VideoEventActor, 65
text
teamusa::TextboxSpawnActor, 49
textboxActive
teamusa::VideoEngine, 64
textboxPadding
teamusa::VideoEngine, 64
textboxRegion
teamusa::VideoEngine, 64
TextboxSpawnActor
teamusa::TextboxSpawnActor, 48
TextboxSpawnActor.cpp, 82
TextboxSpawnActor.h, 82
texture_iter
mediawrap::VideoContext, 53
TextureID
mediawrap::VideoContext, 53
teamusa, 11
textureID
teamusa::ActorVideo, 14
textureId
teamusa::DelayedVideoActor, 26
textures
mediawrap::VideoContext, 57
Timer
teamusa::Timer, 49
Timer.cpp, 82
Timer.h, 82
transitionSteps
teamusa::MovingActor, 38
type
teamusa::ActorEvent, 13

unpause
teamusa::Timer, 50

value
teamusa::ActorEvent, 13
video_display
mediawrap::VideoContext, 58
VideoActor
teamusa::VideoActor, 51
VideoActor.cpp, 83
VideoActor.h, 83
VideoContext
mediawrap::VideoContext, 54
videoContext
teamusa::VideoEngine, 64
VideoContext.cpp, 83
VideoContext.hpp, 84
VideoDisplay
mediawrap::VideoDisplay, 58
VideoDisplay.cpp, 84
VideoDisplay.hpp, 84
VideoEngine
teamusa::VideoEngine, 60
VideoEngine.cpp, 84
VideoEngine.hpp, 85
VideoEventActor
teamusa::VideoEventActor, 65
VideoEventActor.cpp, 85
VideoEventActor.h, 85

wGrowth
teamusa::MovingActor, 38
window

mediawrap::VideoDisplay, 59

x
teamusa::Point, 42
xSpeed
teamusa::MovingActor, 38

y
teamusa::Point, 42
ySpeed
teamusa::MovingActor, 38