Engine API documentation

Generated by Doxygen 1.8.10

Thu Dec 17 2015 08:15:24

Contents

1 JavaScript Public API **1**

2 GNU Rocket integration **3**

3 Deprecated List **5**

4 Module Index **7**

4.1 Modules 7

5 Data Structure Index **9**

5.1 Data Structures 9

6 Module Documentation **11**

6.1 Sound functionality. 11

6.1.1 Detailed Description 12

6.1.2 Function Documentation 12

6.1.2.1 getPlaylistMusic(void) 12

6.1.2.2 setPlaylistMusic(const char ∗file) 12

6.1.2.3 soundAddSongToPlaylist(const char ∗_filename, const char ∗title, int length) . . 12

6.1.2.4 soundCheckPlayer(void) 13

6.1.2.5 soundClearPlaylist(void) 13

6.1.2.6 soundGetCurrentSong(void) 13

6.1.2.7 soundGetPlaylistSize(void) 13

6.1.2.8 soundGetSongCurrentPlayTime(void) 14

6.1.2.9 soundGetSongFilename(int song_number) 14

6.1.2.10 soundGetSongLength(int song_number) 14

6.1.2.11 soundGetSongName(int song_number) 14

6.1.2.12 soundGetTrackNumber(int song_number) 15

6.1.2.13 soundIsMute(void) 15

6.1.2.14 soundIsPlaying(void) 15

6.1.2.15 soundLoadPlaylist(const char ∗_filename) 15

6.1.2.16 soundLoadSong(int song_number) 15

6.1.2.17 soundMute(int _mute_sound) 16

6.1.2.18 soundNextTrack(void) 16
6.1.2.19 soundPlaySong(int song_number) 16
6.1.2.20 soundPreviousTrack(void) 16
6.1.2.21 soundSetPosition(float position) 16
6.1.2.22 soundStop(void) 17
6.2 Camera functionality 18
6.2.1 Detailed Description 18
6.2.2 Function Documentation 18
6.2.2.1 getCamera() 18
6.2.2.2 setCameraLookAt(float x, float y, float z) 18
6.2.2.3 setCameraPerspective(double fovy, double aspect, double zNear, double zFar) . 19
6.2.2.4 setCameraPosition(float x, float y, float z) 19
6.2.2.5 setCameraPositionObject(object3d_t ∗object) 19
6.2.2.6 setCameraTargetObject(object3d_t ∗object) 20
6.2.2.7 setCameraUpVector(float x, float y, float z) 20
6.3 Frame Buffer Object (FBO) 21
6.3.1 Detailed Description 21
6.3.2 Function Documentation 21
6.3.2.1 fboBind(fbo_t ∗fbo) 21
6.3.2.2 fboBindTextures(fbo_t ∗fbo) 21
6.3.2.3 fboDeinit(fbo_t ∗fbo) 22
6.3.2.4 fboGenerateFramebuffer(fbo_t ∗fbo) 22
6.3.2.5 fboGetHeight(fbo_t ∗fbo) 22
6.3.2.6 fboGetWidth(fbo_t ∗fbo) 22
6.3.2.7 fboInit(const char ∗name) 23
6.3.2.8 fboSetDimensions(fbo_t ∗fbo, unsigned int width, unsigned int height) 23
6.3.2.9 fboSetRenderDimensions(fbo_t ∗fbo, double widthPercent, double heightPercent) 23
6.3.2.10 fboStoreDepth(fbo_t ∗fbo, int _storeDepth) 24
6.3.2.11 fboUpdateViewport(fbo_t ∗fbo) 25
6.4 Screen functionality 26
6.4.1 Detailed Description 26
6.4.2 Function Documentation 26
6.4.2.1 perspective2dBegin(int w, int h) 26
6.4.2.2 perspective2dEnd(void) 26
6.4.2.3 viewReset(void) 26
6.5 Particle handling 28
6.5.1 Detailed Description 28
6.5.2 Function Documentation 28
6.5.2.1 deinitParticleContainer(void ∗particleContainerPointer) 28
6.5.2.2 drawParticleContainer(particleContainer_t ∗particleContainer) 28

6.5.2.3 initParticleContainer(particleContainer_t ∗particleContainer) 28
6.5.2.4 initParticleContainerParticles(particleContainer_t ∗particleContainer, unsigned int particleI, unsigned int count) 29
6.6 Texture functionality 30
6.6.1 Detailed Description 30
6.6.2 Function Documentation 30
6.6.2.1 drawTexture(texture_t ∗texture) 30
6.6.2.2 setCustomDimensionToTexture(texture_t ∗texture, int w, int h) 31
6.6.2.3 setTextureBlendFunc(texture_t ∗texture, unsigned int srcBlend, unsigned int dst←Blend) 31
6.6.2.4 setTextureCanvasDimensions(texture_t ∗texture, int w, int h) 31
6.6.2.5 setTextureCenterAlignment(texture_t ∗texture, int center) 31
6.6.2.6 setTextureDefaults(texture_t ∗texture) 32
6.6.2.7 setTexturePerspective3d(texture_t ∗texture, int perspective3d) 32
6.6.2.8 setTexturePivot(texture_t ∗texture, double x, double y, double z) 32
6.6.2.9 setTexturePosition(texture_t ∗texture, double x, double y, double z) 32
6.6.2.10 setTextureRotation(texture_t ∗texture, double degreesX, double degreesY, double degreesZ, double x, double y, double z) 33
6.6.2.11 setTextureScale(texture_t ∗texture, double scaleW, double scaleH) 33
6.6.2.12 setTextureSizeToScreenSize(texture_t ∗texture) 33
6.6.2.13 setTextureUnitTexture(texture_t ∗texture, unsigned int unitIndex, texture_←t ∗textureDst) 34
6.6.2.14 setTextureUvDimensions(texture_t ∗texture, double uMin, double vMin, double uMax, double vMax) 34
6.6.2.15 textureDeinit(texture_t ∗texture) 34
6.6.2.16 textureInit(texture_t ∗texture) 34
6.7 GNU Rocket Sync Editor 35
6.7.1 Detailed Description 35
6.7.2 Function Documentation 35
6.7.2.1 isSyncEditor(void) 35
6.7.2.2 syncEditorDeinit(void) 35
6.7.2.3 syncEditorGetRowsPerBeat(void) 35
6.7.2.4 syncEditorGetTrack(const char ∗trackName) 36
6.7.2.5 syncEditorGetTrackCurrentValue(void ∗trackPointer) 37
6.7.2.6 syncEditorInit(void) 37
6.7.2.7 syncEditorSetRowsPerBeat(int _rowsPerBeat) 37
6.8 Global time handling functionality 38
6.8.1 Detailed Description 38
6.8.2 Function Documentation 38
6.8.2.1 timerGetBeatsPerMinute(void) 38
6.8.2.2 timerGetCurrentBeat(void) 38

6.8.2.3 timerSetBeatsPerMinute(double _bpm) 38

7 Data Structure Documentation 39

7.1 _player_texture Struct Reference 39

7.1.1 Detailed Description 39

7.2 camera_t Struct Reference 39

7.2.1 Detailed Description 40

7.3 color_t Struct Reference 40

7.3.1 Detailed Description 40

7.4 cubic Struct Reference 40

7.4.1 Detailed Description 41

7.5 cubic3d Struct Reference 41

7.5.1 Detailed Description 41

7.6 cubicSpline Struct Reference 41

7.6.1 Detailed Description 41

7.7 object3d_t::data Union Reference 41

7.7.1 Detailed Description 42

7.8 dimension_t Struct Reference 42

7.8.1 Detailed Description 42

7.9 exprCalculation_t Struct Reference 42

7.9.1 Detailed Description 42

7.10 exprFunction_t Struct Reference 43

7.10.1 Detailed Description 43

7.11 exprOperator_t Struct Reference 43

7.11.1 Detailed Description 43

7.12 exprVariable_t Struct Reference 43

7.12.1 Detailed Description 44

7.13 fbo_t Struct Reference 44

7.13.1 Detailed Description 44

7.14 gui_component_t Struct Reference 44

7.14.1 Detailed Description 45

7.15 gui_mouse_t Struct Reference 45

7.15.1 Detailed Description 45

7.16 imageData_t Struct Reference 45

7.16.1 Detailed Description 46

7.17 light_t Struct Reference 46

7.17.1 Detailed Description 46

7.18 matrix33_t Struct Reference 46

7.18.1 Detailed Description 46

7.19 matrix44_t Struct Reference 47

7.19.1 Detailed Description 47
7.20 memory_t Struct Reference 47
7.20.1 Detailed Description 47
7.21 mesh_extension Struct Reference 47
7.21.1 Detailed Description 47
7.22 object3d_t Struct Reference 48
7.22.1 Detailed Description 48
7.23 object_shape_cylinder_t Struct Reference 48
7.23.1 Detailed Description 48
7.24 object_shape_disk_t Struct Reference 49
7.24.1 Detailed Description 49
7.25 object_shape_sphere_t Struct Reference 49
7.25.1 Detailed Description 49
7.26 particle_t Struct Reference 49
7.26.1 Detailed Description 50
7.27 particleContainer_t Struct Reference 50
7.27.1 Detailed Description 50
7.28 playerEffect Struct Reference 51
7.28.1 Detailed Description 51
7.29 playerScene Struct Reference 51
7.29.1 Detailed Description 51
7.30 point Struct Reference 51
7.30.1 Detailed Description 52
7.31 point2d_t Struct Reference 52
7.31.1 Detailed Description 52
7.32 point3d Struct Reference 52
7.32.1 Detailed Description 52
7.33 point3d_t Struct Reference 52
7.33.1 Detailed Description 53
7.34 sceneTime_t Struct Reference 53
7.34.1 Detailed Description 53
7.35 shader_t Struct Reference 53
7.35.1 Detailed Description 53
7.36 shaderProgram_t Struct Reference 54
7.36.1 Detailed Description 54
7.37 object3d_t::shape Union Reference 54
7.37.1 Detailed Description 54
7.38 song_t Struct Reference 54
7.38.1 Detailed Description 54
7.39 spline Struct Reference 55

7.39.1 Detailed Description 55
7.40 splineContainer Struct Reference 55
7.40.1 Detailed Description 55
7.41 splineLayer Struct Reference 55
7.41.1 Detailed Description 55
7.42 splinePoint Struct Reference 56
7.42.1 Detailed Description 56
7.43 sync_cb Struct Reference 56
7.43.1 Detailed Description 56
7.44 sync_device Struct Reference 56
7.44.1 Detailed Description 56
7.45 sync_io_cb Struct Reference 57
7.45.1 Detailed Description 57
7.46 sync_track Struct Reference 57
7.46.1 Detailed Description 57
7.47 texture_t Struct Reference 57
7.47.1 Detailed Description 58
7.48 track_key Struct Reference 58
7.48.1 Detailed Description 58
7.49 vbo_t Struct Reference 58
7.49.1 Detailed Description 59

8 Example Documentation 61
8.1 player.c 61

Index 63

Chapter 1

JavaScript Public API

JavaScript Public API refers that a function can be called from the JavaScript as well.

Please refer to the JavaScript engine documentation for more information and examples.

Chapter 2

GNU Rocket integration

- This directory contains GNU Rocket library: `https://github.com/kusma/rocket`
- The GNU Rocket library is customized
  - synceditor.c/h contains API wrappers that the engine is using. Everything else is considered "GNU Rocket internal" functionality
  - Sync editor and player (-DSYNC_PLAYER) functionality is bundled together. Demo engine will determine based on –tool command line switch and GNU Rocket server connectivity which mode should be used.

Chapter 3

Deprecated List

globalScope> Global setCustomDimensionToTexture (texture_t *texture, int w, int h)

This might be removed in the future

Chapter 4

Module Index

4.1 Modules

Here is a list of all modules:

Sound functionality. 11
Camera functionality 18
Frame Buffer Object (FBO) 21
Screen functionality 26
Particle handling 28
Texture functionality 30
GNU Rocket Sync Editor 35
Global time handling functionality 38

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

_player_texture 39
camera_t
Camera singleton data 39
color_t
RGBA color information 40
cubic 40
cubic3d 41
cubicSpline 41
object3d_t::data 41
dimension_t
Dimension information 42
exprCalculation_t 42
exprFunction_t 43
exprOperator_t 43
exprVariable_t 43
fbo_t
Frame Buffer Object (FBO) information 44
gui_component_t 44
gui_mouse_t 45
imageData_t 45
light_t 46
matrix33_t
3x3 matrix 46
matrix44_t
4x4 matrix 47
memory_t 47
mesh_extension 47
object3d_t 48
object_shape_cylinder_t 48
object_shape_disk_t 49
object_shape_sphere_t 49
particle_t 49
particleContainer_t 50
playerEffect 51
playerScene 51
point 51
point2d_t
2D coordinate 52

point3d 52
point3d_t
3D coordinate 52
sceneTime_t 53
shader_t 53
shaderProgram_t 54
object3d_t::shape 54
song_t 54
spline 55
splineContainer 55
splineLayer 55
splinePoint 56
sync_cb 56
sync_device 56
sync_io_cb 57
sync_track 57
texture_t 57
track_key 58
vbo_t 58

Chapter 6

Module Documentation

6.1 Sound functionality.

Functions

- void setPlaylistMusic (const char ∗file)

  Set the main music file.
- const char ∗ getPlaylistMusic (void)

  Get the main music file.
- void soundSetPosition (float position)

  Rewind sound to position in seconds.
- void soundInit (void)

  Initialize audio.
- void soundLoadPlaylist (const char ∗_filename)

  Load playlist in PLS format.
- int soundAddSongToPlaylist (const char ∗_filename, const char ∗title, int length)

  Add audio file to current playlist.
- void soundLoadSong (int song_number)

  Load song in the playlist.
- void soundPause (void)

  Pause or unpause the audio
  JavaScript Public API.
- void soundMute (int _mute_sound)

  Mute or unmute the audio.
- int soundIsMute (void)

  Check if audio is muted.
- void soundStop (void)

  Stop audio.
- void soundFree (void)

  Free sound and cleanup memory.
- void soundPreviousTrack (void)

  Play previous track in the playlist.
- void soundNextTrack (void)

  Play next track in the playlist.
- int soundGetPlaylistSize (void)

  Get size of the playlist.
- int soundGetCurrentSong (void)

Get the index of the current track.
- int soundGetSongLength (int song_number)

  Get length of the song.
- const char ∗ soundGetSongFilename (int song_number)

  Get file name of the track.
- const char ∗ soundGetSongName (int song_number)

  Get song name of the track.
- int soundGetTrackNumber (int song_number)

  Get the track number.
- int soundIsPlaying (void)

  Check if audio is playing.
- void soundCheckPlayer (void)

  Check player status and rewind to next track in the playlist if the song has ended.
- float soundGetSongCurrentPlayTime (void)

  Get the time how long the current track has played.
- void soundPlaySong (int song_number)

  Play sound.
- void soundClearPlaylist (void)

  Clear the current playlist.
- void soundDeinit (void)

  Deinitialize the audio.

6.1.1 Detailed Description

6.1.2 Function Documentation

6.1.2.1 const char∗ getPlaylistMusic (void)

Get the main music file.

JavaScript Public API

Definition at line 43 of file playlist.c.

6.1.2.2 void setPlaylistMusic (const char ∗ *file*)

Set the main music file.

JavaScript Public API

Definition at line 32 of file playlist.c.

6.1.2.3 int soundAddSongToPlaylist (const char ∗ *_filename,* const char ∗ *title,* int *length*)

Add audio file to current playlist.

Parameters

_filename	Audio file.
title	Title of the audio file.
length	Length of the audio file.

Returns

Song position in the playlist.

JavaScript Public API

Definition at line 517 of file sound.c.

Referenced by soundLoadPlaylist().

6.1.2.4 void soundCheckPlayer (void)

Check player status and rewind to next track in the playlist if the song has ended.

Returns

Track index + 1

Definition at line 812 of file sound.c.

References soundGetCurrentSong(), soundGetSongLength(), soundIsPlaying(), and soundNextTrack().

6.1.2.5 void soundClearPlaylist (void)

Clear the current playlist.

JavaScript Public API

Definition at line 909 of file sound.c.

Referenced by soundDeinit(), and soundLoadPlaylist().

6.1.2.6 int soundGetCurrentSong (void)

Get the index of the current track.

Returns

Index of the current track in the playlist.

JavaScript Public API

Definition at line 744 of file sound.c.

Referenced by soundCheckPlayer(), soundNextTrack(), soundPlaySong(), and soundPreviousTrack().

6.1.2.7 int soundGetPlaylistSize (void)

Get size of the playlist.

Returns

size of the playlist

JavaScript Public API

Definition at line 733 of file sound.c.

6.1.2.8 float soundGetSongCurrentPlayTime (void)

Get the time how long the current track has played.

Returns

Play time in seconds.

JavaScript Public API

Definition at line 831 of file sound.c.

6.1.2.9 const char∗ soundGetSongFilename (int *song_number*)

Get file name of the track.

Parameters

song_number	Index of the track in the playlist.

Returns

file name of the track.

JavaScript Public API

Definition at line 768 of file sound.c.

6.1.2.10 int soundGetSongLength (int *song_number*)

Get length of the song.

Parameters

song_number	Index of the track in the playlist.

Returns

length of the song in seconds.

JavaScript Public API

Definition at line 756 of file sound.c.

Referenced by soundCheckPlayer(), and soundPlaySong().

6.1.2.11 const char∗ soundGetSongName (int *song_number*)

Get song name of the track.

Parameters

song_number	Index of the track in the playlist.

Returns

song name of the track.

JavaScript Public API

Definition at line 780 of file sound.c.

6.1.2.12 int soundGetTrackNumber (int *song_number*)

Get the track number.

Returns

Track index + 1

JavaScript Public API

Definition at line 791 of file sound.c.

6.1.2.13 int soundIsMute (void)

Check if audio is muted.

Returns

1 is mute, 0 is unmute

JavaScript Public API

Definition at line 644 of file sound.c.

6.1.2.14 int soundIsPlaying (void)

Check if audio is playing.

Returns

1 if audio is playing, 0 if audio is not playing.

JavaScript Public API

Definition at line 802 of file sound.c.

Referenced by soundCheckPlayer().

6.1.2.15 void soundLoadPlaylist (const char ∗ *_filename*)

Load playlist in PLS format.

Parameters

_filename	PLS file

JavaScript Public API

Definition at line 354 of file sound.c.

References soundAddSongToPlaylist(), and soundClearPlaylist().

6.1.2.16 void soundLoadSong (int *song_number*)

Load song in the playlist.

Parameters

song_number	Song index position in the playlist.

JavaScript Public API

Definition at line 546 of file sound.c.

Referenced by soundPlaySong().

6.1.2.17 void soundMute (int *_mute_sound*)

Mute or unmute the audio.

Parameters

_mute_sound	1 is mute, 0 is unmute

JavaScript Public API

Definition at line 634 of file sound.c.

6.1.2.18 void soundNextTrack (void)

Play next track in the playlist.

When rewinding last track the player will jump to the first track in the playlist.

JavaScript Public API

Definition at line 716 of file sound.c.

References soundGetCurrentSong(), and soundPlaySong().

Referenced by soundCheckPlayer().

6.1.2.19 void soundPlaySong (int *song_number*)

Play sound.

Parameters

song_number	Index of the song to-be-played in the playlist.

JavaScript Public API

Definition at line 842 of file sound.c.

References soundFree(), soundGetCurrentSong(), soundGetSongLength(), soundInit(), soundLoadSong(), and soundStop().

Referenced by soundNextTrack(), and soundPreviousTrack().

6.1.2.20 void soundPreviousTrack (void)

Play previous track in the playlist.

When rewinding first track the player will jump to the last track in the playlist.

JavaScript Public API

Definition at line 697 of file sound.c.

References soundGetCurrentSong(), and soundPlaySong().

6.1.2.21 void soundSetPosition (float *position*)

Rewind sound to position in seconds.

JavaScript Public API

Definition at line 258 of file sound.c.

Referenced by soundStop().

6.1.2.22 void soundStop (void)

Stop audio.

JavaScript Public API

Definition at line 654 of file sound.c.

References soundPause(), and soundSetPosition().

Referenced by soundPlaySong().

6.2 Camera functionality

Functions

- void setCameraPositionObject (object3d_t ∗object)

  Set camera's position to track a specific 3d object.
- void setCameraTargetObject (object3d_t ∗object)

  Set camera's target to track a specific 3d object.
- void setCameraPerspective (double fovy, double aspect, double zNear, double zFar)

  Set camera's perspective.
- void setCameraPosition (float x, float y, float z)

  Set camera's position.
- void setCameraLookAt (float x, float y, float z)

  Set camera's target (look at).
- void setCameraUpVector (float x, float y, float z)

  Set camera's up vector.
- camera_t ∗ getCamera ()

  Get camera's data.
- void cameraInit ()

  Initialize camera's default settings.

6.2.1 Detailed Description

6.2.2 Function Documentation

6.2.2.1 camera_t∗ getCamera ( )

Get camera's data.

Returns

pointer to camera data

See also

camera_t

JavaScript Public API

Definition at line 99 of file camera.c.

Referenced by viewReset().

6.2.2.2 void setCameraLookAt (float *x*, float *y*, float *z*)

Set camera's target (look at).

Default is (0,0,0).

Parameters

x	Camera's X target
y	Camera's Y target
z	Camera's Z target

JavaScript Public API

Definition at line 74 of file camera.c.

Referenced by cameraInit().

6.2.2.3 void setCameraPerspective (double *fovy,* double *aspect,* double *zNear,* double *zFar*)

Set camera's perspective.

Parameters

fovy	Specifies the field of view angle, in degrees, in the y direction. Default is 45.
aspect	Specifies the aspect ratio that determines the field of view in the x direction. The aspect ratio is the ratio of x (width) to y (height). Default is relative screen width / screen height.
zNear	Specifies the distance from the viewer to the near clipping plane (always positive). Default is 1.0.
zFar	Specifies the distance from the viewer to the far clipping plane (always positive). Default is 1000.0.

JavaScript Public API

Definition at line 45 of file camera.c.

Referenced by cameraInit().

6.2.2.4 void setCameraPosition (float *x,* float *y,* float *z*)

Set camera's position.

Default is (0,0,2).

Parameters

x	Camera's X position
y	Camera's Y position
z	Camera's Z position

JavaScript Public API

Definition at line 61 of file camera.c.

Referenced by cameraInit().

6.2.2.5 void setCameraPositionObject (object3d_t ∗ *object*)

Set camera's position to track a specific 3d object.

Camera's position value will be relative to the 3d object. Default is NULL.

Parameters

object	[in] pointer to the object that camera's position will track

JavaScript Public API

Definition at line 20 of file camera.c.

Referenced by cameraInit().

6.2.2.6 void setCameraTargetObject (object3d_t ∗ *object*)

Set camera's target to track a specific 3d object.

Camera's target value will be relative to the 3d object. Default is NULL.

Parameters

object	[in] pointer to the object that camera's target will track

JavaScript Public API

Definition at line 31 of file camera.c.

Referenced by cameraInit().

6.2.2.7 void setCameraUpVector (float *x,* float *y,* float *z*)

Set camera's up vector.

Default is (0,1,0).

Parameters

x	Camera's X up vector
y	Camera's Y up vector
z	Camera's Z up vector

JavaScript Public API

Definition at line 87 of file camera.c.

Referenced by cameraInit().

6.3 Frame Buffer Object (FBO)

Functions

- void fboBind (fbo_t ∗fbo)

  Bind framebuffer to FBO.
- fbo_t ∗ fboInit (const char ∗name)

  Initialize new FBO or retrieve existing FBO.
- int fboGenerateFramebuffer (fbo_t ∗fbo)

  Create FBO textures and the framebuffer object.
- void fboSetDimensions (fbo_t ∗fbo, unsigned int width, unsigned int height)

  Set render dimension percent relative to whole FBO dimensions.
- void fboSetRenderDimensions (fbo_t ∗fbo, double widthPercent, double heightPercent)

  Set render dimension percent relative to whole FBO dimensions.
- int fboGetWidth (fbo_t ∗fbo)

  Get FBO render width.
- int fboGetHeight (fbo_t ∗fbo)

  Get FBO render width.
- void fboUpdateViewport (fbo_t ∗fbo)

  Updates viewport by setting projection and model matrices according to the FBO details.
- void fboDeinit (fbo_t ∗fbo)

  Deinitialize and clean FBO.
- void fboStoreDepth (fbo_t ∗fbo, int _storeDepth)

  Store depth data in FBO.
- void fboBindTextures (fbo_t ∗fbo)

  Binds FBO textures to texture units.

6.3.1 Detailed Description

6.3.2 Function Documentation

6.3.2.1 void fboBind (fbo_t ∗ *fbo*)

Bind framebuffer to FBO.

Parameters

in	*fbo*	Pointer to fbo. NULL binds to default framebuffer.

JavaScript Public API

Definition at line 18 of file fbo.c.

References fbo_t::id.

Referenced by fboDeinit(), and fboGenerateFramebuffer().

6.3.2.2 void fboBindTextures (fbo_t ∗ *fbo*)

Binds FBO textures to texture units.

Texture unit 0 is the FBO color data. Texture unit 1 is the FBO depth data.

Parameters

in	*fbo*	Pointer to fbo. NULL resets texture unit bindings.

JavaScript Public API

Definition at line 256 of file fbo.c.

References fbo_t::color, and fbo_t::depth.

6.3.2.3 void fboDeinit (fbo_t * *fbo*)

Deinitialize and clean FBO.

Parameters

fbo	[in] Pointer to fbo.

JavaScript Public API

Definition at line 223 of file fbo.c.

References fboBind(), fbo_t::id, and fbo_t::name.

6.3.2.4 int fboGenerateFramebuffer (fbo_t * *fbo*)

Create FBO textures and the framebuffer object.

Parameters

fbo	[in] Pointer to fbo.

Returns

1 if OK, 0 if not OK

JavaScript Public API

Definition at line 99 of file fbo.c.

References fbo_t::color, fbo_t::depth, fboBind(), fbo_t::height, fbo_t::id, fbo_t::name, fbo_t::storeDepth, and fbo_t↩ ::width.

6.3.2.5 int fboGetHeight (fbo_t * *fbo*)

Get FBO render width.

Parameters

fbo	[in] Pointer to fbo.

JavaScript Public API

Definition at line 197 of file fbo.c.

References fbo_t::height, and fbo_t::renderHeightPercent.

6.3.2.6 int fboGetWidth (fbo_t * *fbo*)

Get FBO render width.

Parameters

fbo	[in] Pointer to fbo.

JavaScript Public API

Definition at line 185 of file fbo.c.

References fbo_t::renderWidthPercent, and fbo_t::width.

6.3.2.7 fbo_t∗ fboInit (const char ∗ *name*)

Initialize new FBO or retrieve existing FBO.

Parameters

name	[in] Name of the FBO. If FBO is not found with the given name then new FBO is created.

Returns

Pointer to fbo.

JavaScript Public API

Definition at line 50 of file fbo.c.

References fbo_t::color, fbo_t::depth, fboSetDimensions(), fboSetRenderDimensions(), fbo_t::id, fbo_t::name, and fbo_t::storeDepth.

6.3.2.8 void fboSetDimensions (fbo_t ∗ *fbo,* unsigned int *width,* unsigned int *height*)

Set render dimension percent relative to whole FBO dimensions.

Parameters

fbo	[in] Pointer to fbo.
width	FBO texture width
height	FBO texture height

JavaScript Public API

Definition at line 152 of file fbo.c.

References fbo_t::height, and fbo_t::width.

Referenced by fboInit().

6.3.2.9 void fboSetRenderDimensions (fbo_t ∗ *fbo,* double *widthPercent,* double *heightPercent*)

Set render dimension percent relative to whole FBO dimensions.

Parameters

fbo	[in] Pointer to fbo.
widthPercent	Percentual width
heightPercent	Percentual height

JavaScript Public API

Definition at line 167 of file fbo.c.

References fbo_t::renderHeightPercent, and fbo_t::renderWidthPercent.

Referenced by fboInit().

6.3.2.10 void fboStoreDepth (fbo_t * *fbo,* int *_storeDepth*)

Store depth data in FBO.

Parameters

fbo	[in] Pointer to fbo.
_storeDepth	1 if depth is stored, 0 if depth is not stored. Default is 0 in FBOs.

JavaScript Public API

Definition at line 242 of file fbo.c.

References fbo_t::storeDepth.

6.3.2.11 void fboUpdateViewport (fbo_t ∗ *fbo*)

Updates viewport by setting projection and model matrices according to the FBO details.

Parameters

fbo	[in] Pointer to fbo. NULL binds to default framebuffer.

JavaScript Public API

Definition at line 209 of file fbo.c.

References viewReset().

6.4 Screen functionality

Functions

- void viewReset (void)

  Setup projection and model matrices according to camera data.
- void perspective2dBegin (int w, int h)

  Change perspective to 2D.
- void perspective2dEnd (void)

  Change perspective from 2D to 3D.

6.4.1 Detailed Description

6.4.2 Function Documentation

6.4.2.1 void perspective2dBegin (int *w*, int *h*)

Change perspective to 2D.

Parameters

w	width of the 2D screen
h	height of the 2D screen

See also

perspective2dEnd

JavaScript Public API

Definition at line 100 of file graphics.c.

Referenced by drawParticleContainer(), and drawTexture().

6.4.2.2 void perspective2dEnd (void)

Change perspective from 2D to 3D.

See also

perspective2dBegin

JavaScript Public API

Definition at line 126 of file graphics.c.

References viewReset().

Referenced by drawParticleContainer(), and drawTexture().

6.4.2.3 void viewReset (void)

Setup projection and model matrices according to camera data.

See also

camera_t

JavaScript Public API

Definition at line 45 of file graphics.c.

References getCamera().

Referenced by fboUpdateViewport(), and perspective2dEnd().

6.5 Particle handling

Functions

- void deinitParticleContainer (void ∗particleContainerPointer)

  Deinitialize particle container and free memory.
- particleContainer_t ∗ initParticleContainer (particleContainer_t ∗particleContainer)

  Initialize particle container.
- void initParticleContainerParticles (particleContainer_t ∗particleContainer, unsigned int particleI, unsigned int count)

  Initialize particle container.
- void drawParticleContainer (particleContainer_t ∗particleContainer)

  Update and draw particle container.

6.5.1 Detailed Description

6.5.2 Function Documentation

6.5.2.1 void deinitParticleContainer (void ∗ *particleContainerPointer*)

Deinitialize particle container and free memory.

Parameters

particle↩ Container	[in] Pointer to particle container.

Definition at line 24 of file particle.c.

Referenced by initParticleContainer().

6.5.2.2 void drawParticleContainer (particleContainer_t ∗ *particleContainer*)

Update and draw particle container.

Parameters

particle↩ Container	[in] Pointer to particle container. NULL creates a new particle container.

JavaScript Public API

Definition at line 373 of file particle.c.

References perspective2dBegin(), and perspective2dEnd().

6.5.2.3 particleContainer_t∗ initParticleContainer (particleContainer_t ∗ *particleContainer*)

Initialize particle container.

Parameters

particle↩ Container	[in] Pointer to particle container. NULL creates a new particle container.

Returns

pointer to particle container

JavaScript Public API

Definition at line 40 of file particle.c.

References deinitParticleContainer().

6.5.2.4 void initParticleContainerParticles (particleContainer_t ∗ *particleContainer,* unsigned int *particleI,* unsigned int *count*)

Initialize particle container.

Parameters

particle←↩ Container	[in] Pointer to particle container. NULL creates a new particle container.
particleI	particle initialize index
count	number of particles

Returns

pointer to particle container

JavaScript Public API

Definition at line 320 of file particle.c.

6.6 Texture functionality

Functions

- texture_t ∗ textureInit (texture_t ∗texture)

  Initialize texture and set to default values.
- void textureDeinit (texture_t ∗texture)

  Deinitialize texture and free memory.
- void setTexturePerspective3d (texture_t ∗texture, int perspective3d)

  Set texture perspective to 2D or 3D.
- void setTextureBlendFunc (texture_t ∗texture, unsigned int srcBlend, unsigned int dstBlend)

  Set default blending mode for the texture.
- void setCustomDimensionToTexture (texture_t ∗texture, int w, int h)

  Set custom dimensions to the texture.
- void setTextureCanvasDimensions (texture_t ∗texture, int w, int h)

  Set texture's canvas dimensions.
- void setTextureUvDimensions (texture_t ∗texture, double uMin, double vMin, double uMax, double vMax)

  Set texture UV dimensions.
- void setTextureSizeToScreenSize (texture_t ∗texture)

  Set texture dimensions to screen dimensions.
- void setTextureCenterAlignment (texture_t ∗texture, int center)

  Align texture by centering, horizontally or vertically.
- void setTexturePosition (texture_t ∗texture, double x, double y, double z)

  Set texture position.
- void setTexturePivot (texture_t ∗texture, double x, double y, double z)

  Set texture pivot in pixels.
- void setTextureScale (texture_t ∗texture, double scaleW, double scaleH)

  Scale texture size.
- void setTextureRotation (texture_t ∗texture, double degreesX, double degreesY, double degreesZ, double x, double y, double z)

  Set texture rotation angle.
- void setTextureUnitTexture (texture_t ∗texture, unsigned int unitIndex, texture_t ∗textureDst)

  Sets texture units to enable multitexturing.
- void setTextureDefaults (texture_t ∗texture)

  Set texture to default values.
- void drawTexture (texture_t ∗texture)

  Display image in 2D perspective.

6.6.1 Detailed Description

6.6.2 Function Documentation

6.6.2.1 void drawTexture (texture_t ∗ *texture*)

Display image in 2D perspective.

Parameters

texture	[in] pointer to texture

JavaScript Public API

Warning

Name refactoring pending

Definition at line 348 of file texture.c.

References perspective2dBegin(), and perspective2dEnd().

6.6.2.2 void setCustomDimensionToTexture (texture_t ∗ *texture,* int *w,* int *h*)

Set custom dimensions to the texture.

Parameters

`in`	*texture*	pointer to texture
	w	new width
	w	new height

Deprecated This might be removed in the future

Definition at line 117 of file texture.c.

Referenced by setTextureDefaults(), and setTextureSizeToScreenSize().

6.6.2.3 void setTextureBlendFunc (texture_t ∗ *texture,* unsigned int *srcBlend,* unsigned int *dstBlend*)

Set default blending mode for the texture.

Parameters

`in`	*texture*	pointer to texture
	srcBlend	source blending factor, default is GL_SRC_ALPHA
	dstBlend	destination blending factor, default is GL_ONE_MINUS_SRC_ALPHA

See also

`glBlendFunc OpenGL documentation` JavaScript Public API

Definition at line 101 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.4 void setTextureCanvasDimensions (texture_t ∗ *texture,* int *w,* int *h*)

Set texture's canvas dimensions.

Parameters

texture	[in] pointer to texture
w	width, default is getScreenWidth()
w	height, default is getScreenHeight()

Definition at line 153 of file texture.c.

6.6.2.5 void setTextureCenterAlignment (texture_t ∗ *texture,* int *center*)

Align texture by centering, horizontally or vertically.

Parameters

in	*texture*	pointer to texture
	center	alignment mode: 1 = centered, 2 = horizontal, 3 = vertical

JavaScript Public API

Definition at line 202 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.6 void setTextureDefaults (texture_t ∗ *texture*)

Set texture to default values.

Parameters

in	*texture*	pointer to texture

JavaScript Public API

Definition at line 328 of file texture.c.

References setCustomDimensionToTexture(), setTextureBlendFunc(), setTextureCenterAlignment(), setTexturePivot(), setTexturePosition(), setTextureRotation(), and setTextureScale().

Referenced by textureInit().

6.6.2.7 void setTexturePerspective3d (texture_t ∗ *texture,* int *perspective3d*)

Set texture perspective to 2D or 3D.

Parameters

texture	[in] pointer to texture
perspective3d	1 if image is displayed in 3D space, 0 if 2D

JavaScript Public API

Definition at line 85 of file texture.c.

6.6.2.8 void setTexturePivot (texture_t ∗ *texture,* double *x,* double *y,* double *z*)

Set texture pivot in pixels.

Parameters

in	*texture*	pointer to texture
	x	pivot position X
	y	pivot position Y
	z	pivot position Z

JavaScript Public API

Definition at line 241 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.9 void setTexturePosition (texture_t ∗ *texture,* double *x,* double *y,* double *z*)

Set texture position.

Parameters

texture	[in] pointer to texture
x	position X
y	position Y
z	position Z

JavaScript Public API

Definition at line 218 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.10 void setTextureRotation (texture_t * *texture*, double *degreesX*, double *degreesY*, double *degreesZ*, double *x*, double *y*, double *z*)

Set texture rotation angle.

Parameters

texture	[in] pointer to texture
degreesX	X in degrees
degreesY	Y in degrees
degreesZ	Z in degrees
x	X coordinate
y	Y coordinate
z	Z coordinate

JavaScript Public API

Definition at line 284 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.11 void setTextureScale (texture_t * *texture*, double *scaleW*, double *scaleH*)

Scale texture size.

Parameters

`in`	*texture*	pointer to texture
	scaleW	width, 1.0 is default
	scaleH	height, 1.0 is default

JavaScript Public API

Warning

Name refactoring pending

Definition at line 264 of file texture.c.

Referenced by setTextureDefaults().

6.6.2.12 void setTextureSizeToScreenSize (texture_t * *texture*)

Set texture dimensions to screen dimensions.

Parameters

in	*texture*	pointer to texture

JavaScript Public API

Definition at line 186 of file texture.c.

References setCustomDimensionToTexture().

6.6.2.13 void setTextureUnitTexture (texture_t * *texture,* unsigned int *unitIndex,* texture_t * *textureDst*)

Sets texture units to enable multitexturing.

Parameters

texture	[in] pointer to source texture
unitIndex	Texture unit index number, value should be in range 0 ... MAX_TEXTURE_UNITS-1
textureDst	[in] pointer to destination texture which will be assigned to the texture unit

JavaScript Public API

Definition at line 313 of file texture.c.

6.6.2.14 void setTextureUvDimensions (texture_t * *texture,* double *uMin,* double *vMin,* double *uMax,* double *vMax*)

Set texture UV dimensions.

Parameters

in	*texture*	pointer to texture
	uMin	U minimum
	vMin	V minimum
	uMax	U maximum
	vMax	U maximum

JavaScript Public API

Definition at line 171 of file texture.c.

6.6.2.15 void textureDeinit (texture_t * *texture*)

Deinitialize texture and free memory.

Parameters

in	*texture*	pointer to texture

Definition at line 61 of file texture.c.

6.6.2.16 texture_t* textureInit (texture_t * *texture*)

Initialize texture and set to default values.

Parameters

in	*texture*	pointer to texture, if NULL then new texture will be created

Returns

pointer to texture

Definition at line 19 of file texture.c.

References setTextureDefaults().

6.7 GNU Rocket Sync Editor

Functions

- void syncEditorSetRowsPerBeat (int _rowsPerBeat)

  Set rows per beat.
- int syncEditorGetRowsPerBeat (void)

  Get rows per beat.
- void ∗ syncEditorGetTrack (const char ∗trackName)

  Get pointer to track data structure.
- double syncEditorGetTrackCurrentValue (void ∗trackPointer)

  Get current sync value in time.
- int isSyncEditor (void)

  Check if demo tool is connected to GNU Rocket.
- int syncEditorInit (void)

  Initialize sync editor.
- void syncEditorRun (void)

  Update sync values and timing per frame.
- void syncEditorDeinit (void)

  Deinitialize sync editor.

6.7.1 Detailed Description

6.7.2 Function Documentation

6.7.2.1 int isSyncEditor (void)

Check if demo tool is connected to GNU Rocket.

Returns

1 if tool is connected to GNU Rocket, 0 if not

Definition at line 134 of file synceditor.c.

Referenced by syncEditorDeinit(), and syncEditorRun().

6.7.2.2 void syncEditorDeinit (void)

Deinitialize sync editor.

Save created tracks if GNU Rocket connection was established.

Definition at line 203 of file synceditor.c.

References isSyncEditor().

6.7.2.3 int syncEditorGetRowsPerBeat (void)

Get rows per beat.

Returns

Get the rows per beat, default is 8

Definition at line 39 of file synceditor.c.

6.7.2.4 void∗ syncEditorGetTrack (const char ∗ *trackName*)

Get pointer to track data structure.

Parameters

trackName	[in] Track name

Returns

pointer to track data structure

Definition at line 61 of file synceditor.c.

6.7.2.5 double syncEditorGetTrackCurrentValue (void ∗ *trackPointer*)

Get current sync value in time.

Parameters

trackPointer	[in] pointer to track data structure

Returns

current sync value

Definition at line 75 of file synceditor.c.

6.7.2.6 int syncEditorInit (void)

Initialize sync editor.

If tool mode is enabled then attempt to connect to GNU Rocket.

Definition at line 155 of file synceditor.c.

6.7.2.7 void syncEditorSetRowsPerBeat (int *_rowsPerBeat*)

Set rows per beat.

This is to control the precision of sync editor.

Parameters

_rowsPerBeat	[in] set the rows per beat, default is 8

Definition at line 29 of file synceditor.c.

6.8 Global time handling functionality

Functions

- double timerGetBeatsPerMinute (void)

  Get average BPM.
- void timerSetBeatsPerMinute (double _bpm)

  Set average BPM.
- double timerGetCurrentBeat (void)

  Get beat from playing time according to average BPM.

6.8.1 Detailed Description

6.8.2 Function Documentation

6.8.2.1 double timerGetBeatsPerMinute (void)

Get average BPM.

Returns

average beats per minute

JavaScript Public API

Definition at line 227 of file timer.c.

Referenced by timerGetCurrentBeat().

6.8.2.2 double timerGetCurrentBeat (void)

Get beat from playing time according to average BPM.

Returns

beat from playing time according to average BPM

JavaScript Public API

Definition at line 249 of file timer.c.

References timerGetBeatsPerMinute().

6.8.2.3 void timerSetBeatsPerMinute (double *_bpm*)

Set average BPM.

Parameters

_bpm	[in] beats per minute

JavaScript Public API

Definition at line 238 of file timer.c.

Chapter 7

Data Structure Documentation

7.1 _player_texture Struct Reference

Data Fields

- int **valid**
- void ∗ **bitmap**
- float **scale_x**
- float **scale_y**
- texture_t ∗ **texture**
- texture_t ∗ **texture1_map**
- texture_t ∗ **texture1_mask**
- texture_t ∗ **texture2_map**
- texture_t ∗ **texture2_mask**
- texture_t ∗ **opacity_map**
- texture_t ∗ **opacity_mask**
- texture_t ∗ **bump_map**
- texture_t ∗ **bump_mask**
- texture_t ∗ **specular_map**
- texture_t ∗ **specular_mask**
- texture_t ∗ **shininess_map**
- texture_t ∗ **shininess_mask**
- texture_t ∗ **self_illum_map**
- texture_t ∗ **self_illum_mask**
- texture_t ∗ **reflection_map**
- texture_t ∗ **reflection_mask**

7.1.1 Detailed Description

Definition at line 129 of file 3dsplay.c.

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

- system/graphics/object/3ds/3dsplay.c

7.2 camera_t Struct Reference

Camera singleton data.

```
#include <camera.h>
```

Data Fields

- point3d_t **position**
- point3d_t **lookAt**
- point3d_t **up**
- object3d_t ∗ **positionObject**
- object3d_t ∗ **targetObject**
- double **fovy**
- double **aspect**
- double **zNear**
- double **zFar**

7.2.1 Detailed Description

Camera singleton data.

Definition at line 11 of file camera.h.

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

- system/graphics/camera.h

7.3 color_t Struct Reference

RGBA color information.

```
#include <datatypes.h>
```

Data Fields

- float **r**
- float **g**
- float **b**
- float **a**

7.3.1 Detailed Description

RGBA color information.

Floating point 0.0 - 1.0.

Definition at line 23 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.4 cubic Struct Reference

Data Fields

- float **a**
- float **b**
- float **c**
- float **d**

7.4.1 Detailed Description

Definition at line 5 of file cubicSpline.c.

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

- system/math/splines/cubic/cubicSpline.c
- system/ui/editors/splineEditor.c

7.5 cubic3d Struct Reference

Data Fields

- cubic **x**
- cubic **y**
- cubic **z**

7.5.1 Detailed Description

Definition at line 10 of file cubicSpline.c.

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

- system/math/splines/cubic/cubicSpline.c
- system/ui/editors/splineEditor.c

7.6 cubicSpline Struct Reference

Data Fields

- float **x**
- float **y**
- float **bX**
- float **bY**
- char **isCurve**
- struct cubicSpline ∗ **next**

7.6.1 Detailed Description

Definition at line 25 of file splineEditor.c.

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

- system/ui/editors/splineEditor.c

7.7 object3d_t::data Union Reference

Data Fields

- Lib3dsFile ∗ **file**
- GLUquadric ∗ **quadric**

7.7.1 Detailed Description

Definition at line 54 of file object3d.h.

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

- system/graphics/object/object3d.h

7.8 dimension_t Struct Reference

Dimension information.

```
#include <datatypes.h>
```

Data Fields

- int **width**
- int **height**

7.8.1 Detailed Description

Dimension information.

Definition at line 30 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.9 exprCalculation_t Struct Reference

Data Fields

- char ∗ **expression**
- exprVariable_t ∗ **variableHead**
- exprVariable_t ∗ **variableTail**
- exprOperator_t ∗ **operatorHead**
- exprOperator_t ∗ **operatorTail**
- int **isError**
- double **result**

7.9.1 Detailed Description

Definition at line 42 of file expr.h.

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

- system/math/general/expr.h

7.10 exprFunction_t Struct Reference

Data Fields

- int **type**
- const char ∗ **name**
- int **params**
- const char ∗ **description**

7.10.1 Detailed Description

Definition at line 4 of file expr.h.

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

- system/math/general/expr.h

7.11 exprOperator_t Struct Reference

Data Fields

- double **value**
- int **type**
- int **isLocked**
- int **isNegative**
- char ∗ **actual**
- exprVariable_t ∗ **variable**
- struct exprOperator_t ∗ **prev**
- struct exprOperator_t ∗ **next**

7.11.1 Detailed Description

Definition at line 32 of file expr.h.

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

- system/math/general/expr.h

7.12 exprVariable_t Struct Reference

Data Fields

- union {
  void ∗ **valuePointer**
  float ∗ **floatPointer**
  double ∗ **doublePointer**
  short ∗ **shortPointer**
  int ∗ **intPointer**
  long ∗ **longPointer**
  };

- char ∗ **name**
- struct exprVariable_t ∗ **prev**
- struct exprVariable_t ∗ **next**

7.12.1 Detailed Description

Definition at line 14 of file expr.h.

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

- system/math/general/expr.h

7.13 fbo_t Struct Reference

Frame Buffer Object (FBO) information.

```
#include <fbo.h>
```

Data Fields

- texture_t ∗ color

  FBO color data texture pointer.
- texture_t ∗ depth

  FBO depth data texture pointer.
- int storeDepth

  Store depth when rendering to FBO.
- int width

  FBO width.
- int height

  FBO height.
- double renderWidthPercent

  FBO render Width percent.
- double renderHeightPercent

  FBO height.
- GLuint id

  FBO ID.
- char ∗ name

  FBO logical name.

7.13.1 Detailed Description

Frame Buffer Object (FBO) information.

Definition at line 7 of file fbo.h.

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

- system/graphics/fbo.h

7.14 gui_component_t Struct Reference

Data Fields

- int **resize**
- int **pressed**
- int **selected**

- int **visible**
- int **enabled**
- int **x**
- int **y**
- int **w**
- int **h**
- int **type**
- int **radioGroup**
- int **textalign**
- int **mousePressedX**
- int **mousePressedY**
- float **red**
- float **green**
- float **blue**
- char ∗ **text**

7.14.1 Detailed Description

Definition at line 4 of file menu.h.

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

- system/ui/window/menu.h

7.15 gui_mouse_t Struct Reference

Data Fields

- int **middle**
- int **right**
- int **left**
- int **x**
- int **y**

7.15.1 Detailed Description

Definition at line 54 of file menu.c.

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

- system/ui/window/menu.c

7.16 imageData_t Struct Reference

Data Fields

- char ∗ **name**
- char ∗ **filename**
- unsigned int **w**
- unsigned int **h**
- unsigned int **channels**
- unsigned int ∗ **pixels**

7.16.1 Detailed Description

Definition at line 10 of file image.h.

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

- system/graphics/image/image.h

7.17 light_t Struct Reference

Data Fields

- unsigned int **id**
- int **enabled**
- point3d_t **position**
- point3d_t **direction**
- float **constantAttenuation**
- float **linearAttenuation**
- float **quadricAttenuation**
- float **spotExponent**
- float **spotCutOff**
- color_t **ambient**
- color_t **diffuse**
- color_t **specular**
- object3d_t ∗ **positionObject**

7.17.1 Detailed Description

Definition at line 6 of file lighting.h.

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

- system/graphics/object/lighting.h

7.18 matrix33_t Struct Reference

3x3 matrix

```
#include <datatypes.h>
```

Data Fields

- float **m** [9]

7.18.1 Detailed Description

3x3 matrix

Definition at line 44 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.19 matrix44_t Struct Reference

4x4 matrix

```
#include <datatypes.h>
```

Data Fields

- float **m** [16]

7.19.1 Detailed Description

4x4 matrix

Definition at line 37 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.20 memory_t Struct Reference

Data Fields

- void ∗ **ptr**
- void(∗ **deinit**)(void ∗)
- unsigned int **type**
- struct memory_t ∗ **next**

7.20.1 Detailed Description

Definition at line 5 of file memory.c.

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

- system/datatypes/memory.c

7.21 mesh_extension Struct Reference

Data Fields

- vbo_t ∗ **vbo**
- Lib3dsMaterial ∗ **material**
- GLuint **list**

7.21.1 Detailed Description

Definition at line 161 of file 3dsplay.c.

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

- system/graphics/object/3ds/3dsplay.c

7.22 object3d_t Struct Reference

Data Structures

- union data
- union shape

Data Fields

- char ∗ **filename**
- int **tex_mode**
- int **clearZBuffer**
- int **objectType**
- int **useObjectCamera**
- int **useObjectLighting**
- const char ∗ **camera**
- point3d_t **position**
- point3d_t **scale**
- point3d_t **pivot**
- point3d_t **degrees**
- point3d_t **angle**
- color_t **color**
- union object3d_t::data **data**
- union object3d_t::shape **shape**

7.22.1 Detailed Description

Definition at line 37 of file object3d.h.

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

- system/graphics/object/object3d.h

7.23 object_shape_cylinder_t Struct Reference

Data Fields

- double **base**
- double **top**
- double **height**
- int **slices**
- int **stacks**

7.23.1 Detailed Description

Definition at line 21 of file object3d.h.

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

- system/graphics/object/object3d.h

7.24 object_shape_disk_t Struct Reference

Data Fields

- double **inner**
- double **outer**
- int **slices**
- int **loops**

7.24.1 Detailed Description

Definition at line 14 of file object3d.h.

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

- system/graphics/object/object3d.h

7.25 object_shape_sphere_t Struct Reference

Data Fields

- double **radius**
- int **lats**
- int **longs**

7.25.1 Detailed Description

Definition at line 29 of file object3d.h.

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

- system/graphics/object/object3d.h

7.26 particle_t Struct Reference

Data Fields

- texture_t ∗ **texture**
- int **active**
- float **startTime**
- float **duration**
- float **progress**
- float **initTime**
- float **alpha**
- point3d_t **position**
- point3d_t **startPosition**
- point3d_t **endPosition**
- point3d_t **scale**
- point3d_t **startScale**
- point3d_t **endScale**
- point3d_t **angle**
- point3d_t **startAngle**

- point3d_t **endAngle**
- point3d_t **pivot**
- color_t **color**

7.26.1 Detailed Description

Definition at line 4 of file particle.h.

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

- system/graphics/particle/particle.h

7.27 particleContainer_t Struct Reference

Data Fields

- particle_t ∗ **particleList**
- unsigned int **particleCount**
- texture_t ∗∗ **particleDefaultTextureList**
- unsigned int **particleDefaultTextureCount**
- float **particleDurationMin**
- float **particleDurationMax**
- float **particleFadeInTime**
- float **particleFadeOutTime**
- point3d_t **positionMin**
- point3d_t **positionMax**
- point3d_t **position**
- point3d_t **particleScaleMin**
- point3d_t **particleScaleMax**
- point3d_t **particleAngleMin**
- point3d_t **particleAngleMax**
- point3d_t **particlePivot**
- point3d_t **direction**
- color_t **particleColor**
- float **startTime**
- float **duration**
- int **perspective3d**
- void(∗ **initParticle**)(particleContainer_t ∗, particle_t ∗)
- void ∗ **initParticleClientData**
- void(∗ **updateParticle**)(particleContainer_t ∗, particle_t ∗)
- void ∗ **updateParticleClientData**
- void(∗ **updateParticleContainer**)(particleContainer_t ∗)
- void ∗ **updateParticleContainerClientData**

7.27.1 Detailed Description

Definition at line 26 of file particle.h.

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

- system/graphics/particle/particle.h

7.28 playerEffect Struct Reference

Data Fields

- char ∗ **name**
- char ∗ **reference**
- time_t **fileLastModifiedTime**
- void(∗ **init**)(playerScene ∗)
- void(∗ **run**)(playerScene ∗)
- void(∗ **deinit**)(playerScene ∗)
- int **initialized**
- int **type**
- struct playerEffect ∗ **next**

7.28.1 Detailed Description

Definition at line 15 of file player.h.

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

- system/player/player.h

7.29 playerScene Struct Reference

Data Fields

- char ∗ **name**
- float **start**
- float **end**
- playerEffect ∗ **effect**
- sceneTime_t **time**
- unsigned int **variablesSize**
- struct playerScene ∗ **next**
- struct playerScene ∗ **playerSceneHead**
- struct playerScene ∗ **playerSceneTail**

7.29.1 Detailed Description

Definition at line 39 of file player.h.

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

- system/player/player.h

7.30 point Struct Reference

Data Fields

- float **x**
- float **y**
- float **bX**
- float **bY**

7.30.1 Detailed Description

Definition at line 20 of file splineEditor.c.

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

- system/ui/editors/splineEditor.c

7.31 point2d_t Struct Reference

2D coordinate

```
#include <datatypes.h>
```

Data Fields

- float **x**
- float **y**

7.31.1 Detailed Description

2D coordinate

Definition at line 9 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.32 point3d Struct Reference

Data Fields

- float **x**
- float **y**
- float **z**

7.32.1 Detailed Description

Definition at line 12 of file spline.h.

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

- system/math/splines/spline.h

7.33 point3d_t Struct Reference

3D coordinate

```
#include <datatypes.h>
```

Data Fields

- float **x**
- float **y**
- float **z**

7.33.1 Detailed Description

3D coordinate

Definition at line 16 of file datatypes.h.

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

- system/datatypes/datatypes.h

7.34 sceneTime_t Struct Reference

Data Fields

- float **percent**
- float **absolute**
- float **now**
- float **start**
- float **end**
- float **interval**
- float **triggerStart**
- float **triggerEnd**

7.34.1 Detailed Description

Definition at line 28 of file player.h.

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

- system/player/player.h

7.35 shader_t Struct Reference

Data Fields

- char ∗ **name**
- char ∗ **filename**
- unsigned int **id**
- int **type**
- time_t **fileLastModifiedTime**

7.35.1 Detailed Description

Definition at line 4 of file shader.h.

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

- system/graphics/shader/shader.h

7.36 shaderProgram_t Struct Reference

Data Fields

- char ∗ **name**
- unsigned int **id**
- shader_t ∗∗ **attachedShaders**
- unsigned int **attachedShadersCount**

7.36.1 Detailed Description

Definition at line 12 of file shader.h.

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

- system/graphics/shader/shader.h

7.37 object3d_t::shape Union Reference

Data Fields

- object_shape_disk_t **disk**
- object_shape_cylinder_t **cylinder**
- object_shape_sphere_t **sphere**

7.37.1 Detailed Description

Definition at line 59 of file object3d.h.

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

- system/graphics/object/object3d.h

7.38 song_t Struct Reference

Data Fields

- unsigned int **length**
- char ∗ **name**
- char ∗ **filename**

7.38.1 Detailed Description

Definition at line 62 of file sound.c.

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

- system/audio/sound.c

7.39 spline Struct Reference

Data Fields

- unsigned int **size**
- unsigned int **detail**
- float **width**
- splinePoint ∗ **splinePointTail**
- splinePoint ∗ **splinePointHead**
- struct spline ∗ **next**

7.39.1 Detailed Description

Definition at line 24 of file spline.h.

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

- system/math/splines/spline.h

7.40 splineContainer Struct Reference

Data Fields

- unsigned int **size**
- unsigned int **pointsCount**
- splineLayer ∗ **splineLayerTail**
- splineLayer ∗ **splineLayerHead**
- struct splineContainer ∗ **next**

7.40.1 Detailed Description

Definition at line 48 of file spline.h.

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

- system/math/splines/spline.h

7.41 splineLayer Struct Reference

Data Fields

- unsigned int **size**
- spline ∗ **splineTail**
- spline ∗ **splineHead**
- struct splineLayer ∗ **next**

7.41.1 Detailed Description

Definition at line 37 of file spline.h.

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

- system/math/splines/spline.h

7.42 splinePoint Struct Reference

Data Fields

- float **x**
- float **y**
- float **z**
- struct splinePoint ∗ **next**

7.42.1 Detailed Description

Definition at line 16 of file spline.h.

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

- system/math/splines/spline.h

7.43 sync_cb Struct Reference

Data Fields

- void(∗ **pause**)(void ∗, int)
- void(∗ **set_row**)(void ∗, int)
- int(∗ **is_playing**)(void ∗)

7.43.1 Detailed Description

Definition at line 22 of file sync.h.

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

- system/rocket/sync.h

7.44 sync_device Struct Reference

Data Fields

- char ∗ **base**
- struct sync_track ∗∗ **tracks**
- size_t **num_tracks**
- int **row**
- SOCKET **sock**
- struct sync_io_cb **io_cb**

7.44.1 Detailed Description

Definition at line 42 of file device.h.

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

- system/rocket/device.h

7.45 sync_io_cb Struct Reference

Data Fields

- void ∗(∗ **open**)(const char ∗filename, const char ∗mode)
- size_t(∗ **read**)(void ∗ptr, size_t size, size_t nitems, void ∗stream)
- int(∗ **close**)(void ∗stream)

7.45.1 Detailed Description

Definition at line 32 of file sync.h.

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

- system/rocket/sync.h

7.46 sync_track Struct Reference

Data Fields

- char ∗ **name**
- struct track_key ∗ **keys**
- int **num_keys**

7.46.1 Detailed Description

Definition at line 22 of file track.h.

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

- system/rocket/track.h

7.47 texture_t Struct Reference

Data Fields

- char ∗ **name**
- unsigned int **w**
- unsigned int **h**
- unsigned int **customWidth**
- unsigned int **customHeight**
- unsigned int **center**
- unsigned int **id**
- unsigned int **hasAlpha**
- unsigned int **hasCustomDimensions**
- unsigned int **canvasWidth**
- unsigned int **canvasHeight**
- double **x**
- double **y**
- double **z**
- double **pivotX**
- double **pivotY**

- double **pivotZ**
- double **scaleW**
- double **scaleH**
- double **angleX**
- double **angleY**
- double **angleZ**
- double **degreesX**
- double **degreesY**
- double **degreesZ**
- double **uMin**
- double **vMin**
- double **uMax**
- double **vMax**
- int **perspective3d**
- unsigned int **srcBlend**
- unsigned int **dstBlend**
- unsigned int **multiTextureId** [MAX_TEXTURE_UNITS]

7.47.1 Detailed Description

Definition at line 13 of file texture.h.

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

- system/graphics/texture.h

7.48 track_key Struct Reference

Data Fields

- int **row**
- float **value**
- enum key_type **type**

7.48.1 Detailed Description

Definition at line 16 of file track.h.

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

- system/rocket/track.h

7.49 vbo_t Struct Reference

Data Fields

- GLuint **id**
- GLuint **vertexId**
- GLuint **normalId**
- GLuint **texCoordId**
- unsigned int **count**

7.49.1 Detailed Description

Definition at line 4 of file vbo.h.

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

- system/graphics/object/vbo.h

Chapter 8

Example Documentation

8.1 player.c

Previews a *3DS* file using OpenGL.

```
Syntax: player filename
```

Warning

To compile this program you must have OpenGL and glut installed.

Index

_player_texture, 39

Camera functionality, 18
- getCamera, 18
- setCameraLookAt, 18
- setCameraPerspective, 19
- setCameraPosition, 19
- setCameraPositionObject, 19
- setCameraTargetObject, 19
- setCameraUpVector, 20

camera_t, 39
color_t, 40
cubic, 40
cubic3d, 41
cubicSpline, 41

deinitParticleContainer
- Particle handling, 28

dimension_t, 42
drawParticleContainer
- Particle handling, 28

drawTexture
- Texture functionality, 30

exprCalculation_t, 42
exprFunction_t, 43
exprOperator_t, 43
exprVariable_t, 43

fbo_t, 44
fboBind
- Frame Buffer Object (FBO), 21

fboBindTextures
- Frame Buffer Object (FBO), 21

fboDeinit
- Frame Buffer Object (FBO), 22

fboGenerateFramebuffer
- Frame Buffer Object (FBO), 22

fboGetHeight
- Frame Buffer Object (FBO), 22

fboGetWidth
- Frame Buffer Object (FBO), 22

fboInit
- Frame Buffer Object (FBO), 23

fboSetDimensions
- Frame Buffer Object (FBO), 23

fboSetRenderDimensions
- Frame Buffer Object (FBO), 23

fboStoreDepth
- Frame Buffer Object (FBO), 23

fboUpdateViewport
- Frame Buffer Object (FBO), 25

Frame Buffer Object (FBO), 21
- fboBind, 21
- fboBindTextures, 21
- fboDeinit, 22
- fboGenerateFramebuffer, 22
- fboGetHeight, 22
- fboGetWidth, 22
- fboInit, 23
- fboSetDimensions, 23
- fboSetRenderDimensions, 23
- fboStoreDepth, 23
- fboUpdateViewport, 25

GNU Rocket Sync Editor, 35
- isSyncEditor, 35
- syncEditorDeinit, 35
- syncEditorGetRowsPerBeat, 35
- syncEditorGetTrack, 35
- syncEditorGetTrackCurrentValue, 37
- syncEditorInit, 37
- syncEditorSetRowsPerBeat, 37

getCamera
- Camera functionality, 18

getPlaylistMusic
- Sound functionality., 12

Global time handling functionality, 38
- timerGetBeatsPerMinute, 38
- timerGetCurrentBeat, 38
- timerSetBeatsPerMinute, 38

gui_component_t, 44
gui_mouse_t, 45

imageData_t, 45
initParticleContainer
- Particle handling, 28

initParticleContainerParticles
- Particle handling, 29

isSyncEditor
- GNU Rocket Sync Editor, 35

light_t, 46

matrix33_t, 46
matrix44_t, 47
memory_t, 47
mesh_extension, 47

object3d_t, 48
object3d_t::data, 41

object3d_t::shape, 54
object_shape_cylinder_t, 48
object_shape_disk_t, 49
object_shape_sphere_t, 49

Particle handling, 28
    deinitParticleContainer, 28
    drawParticleContainer, 28
    initParticleContainer, 28
    initParticleContainerParticles, 29
particle_t, 49
particleContainer_t, 50
perspective2dBegin
    Screen functionality, 26
perspective2dEnd
    Screen functionality, 26
playerEffect, 51
playerScene, 51
point, 51
point2d_t, 52
point3d, 52
point3d_t, 52

sceneTime_t, 53
Screen functionality, 26
    perspective2dBegin, 26
    perspective2dEnd, 26
    viewReset, 26
setCameraLookAt
    Camera functionality, 18
setCameraPerspective
    Camera functionality, 19
setCameraPosition
    Camera functionality, 19
setCameraPositionObject
    Camera functionality, 19
setCameraTargetObject
    Camera functionality, 19
setCameraUpVector
    Camera functionality, 20
setCustomDimensionToTexture
    Texture functionality, 31
setPlaylistMusic
    Sound functionality., 12
setTextureBlendFunc
    Texture functionality, 31
setTextureCanvasDimensions
    Texture functionality, 31
setTextureCenterAlignment
    Texture functionality, 31
setTextureDefaults
    Texture functionality, 32
setTexturePerspective3d
    Texture functionality, 32
setTexturePivot
    Texture functionality, 32
setTexturePosition
    Texture functionality, 32
setTextureRotation
    Texture functionality, 33
setTextureScale
    Texture functionality, 33
setTextureSizeToScreenSize
    Texture functionality, 33
setTextureUnitTexture
    Texture functionality, 34
setTextureUvDimensions
    Texture functionality, 34
shader_t, 53
shaderProgram_t, 54
song_t, 54
Sound functionality., 11
    getPlaylistMusic, 12
    setPlaylistMusic, 12
    soundAddSongToPlaylist, 12
    soundCheckPlayer, 13
    soundClearPlaylist, 13
    soundGetCurrentSong, 13
    soundGetPlaylistSize, 13
    soundGetSongCurrentPlayTime, 13
    soundGetSongFilename, 14
    soundGetSongLength, 14
    soundGetSongName, 14
    soundGetTrackNumber, 14
    soundIsMute, 15
    soundIsPlaying, 15
    soundLoadPlaylist, 15
    soundLoadSong, 15
    soundMute, 16
    soundNextTrack, 16
    soundPlaySong, 16
    soundPreviousTrack, 16
    soundSetPosition, 16
    soundStop, 17
soundAddSongToPlaylist
    Sound functionality., 12
soundCheckPlayer
    Sound functionality., 13
soundClearPlaylist
    Sound functionality., 13
soundGetCurrentSong
    Sound functionality., 13
soundGetPlaylistSize
    Sound functionality., 13
soundGetSongCurrentPlayTime
    Sound functionality., 13
soundGetSongFilename
    Sound functionality., 14
soundGetSongLength
    Sound functionality., 14
soundGetSongName
    Sound functionality., 14
soundGetTrackNumber
    Sound functionality., 14
soundIsMute
    Sound functionality., 15
soundIsPlaying

Sound functionality., 15
soundLoadPlaylist
Sound functionality., 15
soundLoadSong
Sound functionality., 15
soundMute
Sound functionality., 16
soundNextTrack
Sound functionality., 16
soundPlaySong
Sound functionality., 16
soundPreviousTrack
Sound functionality., 16
soundSetPosition
Sound functionality., 16
soundStop
Sound functionality., 17
spline, 55
splineContainer, 55
splineLayer, 55
splinePoint, 56
sync_cb, 56
sync_device, 56
sync_io_cb, 57
sync_track, 57
syncEditorDeinit
GNU Rocket Sync Editor, 35
syncEditorGetRowsPerBeat
GNU Rocket Sync Editor, 35
syncEditorGetTrack
GNU Rocket Sync Editor, 35
syncEditorGetTrackCurrentValue
GNU Rocket Sync Editor, 37
syncEditorInit
GNU Rocket Sync Editor, 37
syncEditorSetRowsPerBeat
GNU Rocket Sync Editor, 37

Texture functionality, 30
drawTexture, 30
setCustomDimensionToTexture, 31
setTextureBlendFunc, 31
setTextureCanvasDimensions, 31
setTextureCenterAlignment, 31
setTextureDefaults, 32
setTexturePerspective3d, 32
setTexturePivot, 32
setTexturePosition, 32
setTextureRotation, 33
setTextureScale, 33
setTextureSizeToScreenSize, 33
setTextureUnitTexture, 34
setTextureUvDimensions, 34
textureDeinit, 34
textureInit, 34
texture_t, 57
textureDeinit
Texture functionality, 34
textureInit
Texture functionality, 34
timerGetBeatsPerMinute
Global time handling functionality, 38
timerGetCurrentBeat
Global time handling functionality, 38
timerSetBeatsPerMinute
Global time handling functionality, 38
track_key, 58

vbo_t, 58
viewReset
Screen functionality, 26