

GraphicsEngine

Generated by Doxygen 1.9.2

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 std Namespace Reference	9
6 Class Documentation	11
6.1 AABB Class Reference	11
6.1.1 Constructor & Destructor Documentation	11
6.1.1.1 AABB()	12
6.1.1.2 ~AABB()	12
6.1.2 Member Function Documentation	12
6.1.2.1 get_corners()	12
6.1.2.2 init()	12
6.1.2.3 render()	13
6.2 Camera Class Reference	14
6.2.1 Constructor & Destructor Documentation	15
6.2.1.1 Camera() [1/2]	15
6.2.1.2 Camera() [2/2]	15
6.2.1.3 ~Camera()	15
6.2.2 Member Function Documentation	15
6.2.2.1 calculate_viewmatrix()	15
6.2.2.2 get_camera_direction()	15
6.2.2.3 get_camera_position()	16
6.2.2.4 get_far_plane()	16
6.2.2.5 get_fov()	16
6.2.2.6 get_near_plane()	16
6.2.2.7 get_right_axis()	16
6.2.2.8 get_up_axis()	16
6.2.2.9 get_yaw()	16
6.2.2.10 key_control()	17
6.2.2.11 mouse_control()	17
6.2.2.12 set_camera_position()	17
6.2.2.13 set_far_plane()	17
6.2.2.14 set_fov()	17

6.2.2.15 set_near_plane()	17
6.3 CascadedShadowMap Class Reference	18
6.3.1 Constructor & Destructor Documentation	19
6.3.1.1 CascadedShadowMap()	19
6.3.1.2 ~CascadedShadowMap()	19
6.3.2 Member Function Documentation	19
6.3.2.1 get_id()	19
6.3.2.2 get_intensity()	19
6.3.2.3 get_num_active_cascades()	20
6.3.2.4 get_pcf_radius()	20
6.3.2.5 get_shadow_height()	20
6.3.2.6 get_shadow_width()	20
6.3.2.7 init()	20
6.3.2.8 read()	21
6.3.2.9 set_intensity()	21
6.3.2.10 set_pcf_radius()	21
6.3.2.11 write()	21
6.3.3 Member Data Documentation	21
6.3.3.1 FBO	21
6.3.3.2 glErrorChecker_ins	21
6.3.3.3 intensity	21
6.3.3.4 num_active_cascades	22
6.3.3.5 pcf_radius	22
6.3.3.6 shadow_height	22
6.3.3.7 shadow_map	22
6.3.3.8 shadow_width	22
6.4 ClampToEdgeMode Class Reference	23
6.4.1 Constructor & Destructor Documentation	24
6.4.1.1 ClampToEdgeMode()	24
6.4.1.2 ~ClampToEdgeMode()	24
6.4.2 Member Function Documentation	24
6.4.2.1 activate()	24
6.5 Clouds Class Reference	25
6.5.1 Constructor & Destructor Documentation	26
6.5.1.1 Clouds()	26
6.5.1.2 ~Clouds()	26
6.5.2 Member Function Documentation	26
6.5.2.1 get_cirrus_effect()	26
6.5.2.2 get_density()	26
6.5.2.3 get_mesh_scale()	27
6.5.2.4 get_model()	27
6.5.2.5 get_movement_direction()	27

6.5.2.6 get_movement_speed()	27
6.5.2.7 get_normal_model()	27
6.5.2.8 get_pillowness()	27
6.5.2.9 get_powder_effect()	27
6.5.2.10 get_rad()	27
6.5.2.11 get_scale()	28
6.5.2.12 get_shader_program()	28
6.5.2.13 init()	28
6.5.2.14 read()	28
6.5.2.15 render()	29
6.5.2.16 set_cirrus_effect()	29
6.5.2.17 set_density()	29
6.5.2.18 set_movement_direction()	29
6.5.2.19 set_movement_speed()	29
6.5.2.20 set_pillowness()	30
6.5.2.21 set_powder_effect()	30
6.5.2.22 set_scale() [1/2]	30
6.5.2.23 set_scale() [2/2]	30
6.5.2.24 set_translation()	30
6.5.2.25 update_window_params()	30
6.6 CloudsShaderProgram Class Reference	31
6.6.1 Constructor & Destructor Documentation	33
6.6.1.1 CloudsShaderProgram()	33
6.6.1.2 ~CloudsShaderProgram()	33
6.6.2 Member Function Documentation	33
6.6.2.1 get_model_location()	33
6.6.2.2 get_projection_location()	33
6.6.2.3 get_view_location()	33
6.6.2.4 retrieve_uniform_locations()	34
6.6.3 Member Data Documentation	34
6.6.3.1 uniform_model_location	34
6.6.3.2 uniform_projection_location	34
6.6.3.3 uniform_view_location	34
6.7 ComputeShaderProgram Class Reference	35
6.7.1 Constructor & Destructor Documentation	37
6.7.1.1 ComputeShaderProgram()	37
6.7.1.2 ~ComputeShaderProgram()	37
6.7.2 Member Function Documentation	37
6.7.2.1 get_cell_location()	37
6.7.2.2 get_num_cell_location()	38
6.7.2.3 reload()	38
6.7.2.4 retrieve_uniform_locations()	39

6.7.2.5 set_noise()	39
6.8 DirectionalLight Class Reference	39
6.8.1 Constructor & Destructor Documentation	42
6.8.1.1 DirectionalLight() [1/2]	42
6.8.1.2 DirectionalLight() [2/2]	42
6.8.1.3 ~DirectionalLight()	42
6.8.2 Member Function Documentation	43
6.8.2.1 calc_orthogonal_projections()	43
6.8.2.2 calculate_light_transform()	43
6.8.2.3 get_ambient_intensity()	43
6.8.2.4 get_cascaded_light_matrices()	43
6.8.2.5 get_cascaded_slots()	44
6.8.2.6 get_color()	44
6.8.2.7 get_diffuse_intensity()	44
6.8.2.8 get_direction()	44
6.8.2.9 get_light_view_matrix()	44
6.8.2.10 get_shadow_map()	44
6.8.2.11 set_ambient_intensity()	45
6.8.2.12 set_color()	45
6.8.2.13 set_diffuse_intensity()	45
6.8.2.14 set_direction()	45
6.8.2.15 update_shadow_map()	45
6.9 DirectionalLightUniformLocations Struct Reference	46
6.9.1 Member Data Documentation	46
6.9.1.1 uniform_ambient_intensity_location	46
6.9.1.2 uniform_color_location	46
6.9.1.3 uniform_diffuse_intensity_location	47
6.9.1.4 uniform_direction_location	47
6.9.1.5 uniform_shadow_intensity_location	47
6.10 DirectionalShadowMapPass Class Reference	47
6.10.1 Constructor & Destructor Documentation	48
6.10.1.1 DirectionalShadowMapPass() [1/2]	48
6.10.1.2 DirectionalShadowMapPass() [2/2]	49
6.10.1.3 ~DirectionalShadowMapPass()	49
6.10.2 Member Function Documentation	49
6.10.2.1 execute()	49
6.10.2.2 set_game_object_uniforms()	49
6.10.2.3 use_terrain_textures()	49
6.11 GameObject Class Reference	50
6.11.1 Constructor & Destructor Documentation	50
6.11.1.1 GameObject() [1/2]	51
6.11.1.2 GameObject() [2/2]	51

6.11.1.3 ~GameObject()	51
6.11.2 Member Function Documentation	51
6.11.2.1 get_aabb()	51
6.11.2.2 get_material_id()	51
6.11.2.3 get_model()	52
6.11.2.4 get_normal_world_trafo()	52
6.11.2.5 get_world_trafo()	52
6.11.2.6 init()	53
6.11.2.7 render()	53
6.11.2.8 rotate()	53
6.11.2.9 scale()	53
6.11.2.10 set_material_id()	54
6.11.2.11 translate()	54
6.12 GBuffer Class Reference	55
6.12.1 Constructor & Destructor Documentation	55
6.12.1.1 GBuffer() [1/2]	55
6.12.1.2 GBuffer() [2/2]	56
6.12.1.3 ~GBuffer()	56
6.12.2 Member Function Documentation	56
6.12.2.1 create()	56
6.12.2.2 get_id()	56
6.12.2.3 read()	56
6.12.2.4 update_window_params()	56
6.12.2.5 use_gbuffer()	57
6.13 GeometryPass Class Reference	57
6.13.1 Constructor & Destructor Documentation	58
6.13.1.1 GeometryPass() [1/2]	58
6.13.1.2 GeometryPass() [2/2]	59
6.13.1.3 ~GeometryPass()	59
6.13.2 Member Function Documentation	59
6.13.2.1 execute()	59
6.13.2.2 set_game_object_uniforms()	59
6.13.2.3 use_terrain_textures()	60
6.14 GeometryPassShaderProgram Class Reference	60
6.14.1 Constructor & Destructor Documentation	63
6.14.1.1 GeometryPassShaderProgram()	63
6.14.1.2 ~GeometryPassShaderProgram()	63
6.14.2 Member Function Documentation	63
6.14.2.1 get_biom_height_id()	63
6.14.2.2 get_biom_texture_location()	64
6.14.2.3 get_material_id_location()	64
6.14.2.4 get_max_height_id()	64

6.14.2.5 get_model_location()	64
6.14.2.6 get_normal_modal_location()	64
6.14.2.7 get_program_id()	64
6.14.2.8 get_projection_location()	64
6.14.2.9 get_terrain_id()	65
6.14.2.10 get_view_location()	65
6.14.2.11 retrieve_uniform_locations()	65
6.14.3 Member Data Documentation	65
6.14.3.1 uniform_biom_texture_locations	65
6.14.3.2 uniform_biomHeight_id	65
6.14.3.3 uniform_isTerrainValue_id	66
6.14.3.4 uniform_material_id_location	66
6.14.3.5 uniform_max_height_id	66
6.14.3.6 uniform_model_location	66
6.14.3.7 uniform_normal_model_location	66
6.14.3.8 uniform_projection_location	66
6.14.3.9 uniform_view_location	66
6.15 glErrorChecker Class Reference	67
6.15.1 Member Function Documentation	67
6.15.1.1 areErrorPrintAll()	67
6.15.1.2 arePreError()	68
6.16 std::hash< Vertex > Struct Reference	69
6.16.1 Member Function Documentation	70
6.16.1.1 operator()()	70
6.17 Light Class Reference	70
6.17.1 Constructor & Destructor Documentation	71
6.17.1.1 Light() [1/2]	71
6.17.1.2 Light() [2/2]	71
6.17.1.3 ~Light()	72
6.17.2 Member Data Documentation	72
6.17.2.1 ambient_intensity	72
6.17.2.2 color	72
6.17.2.3 diffuse_intensity	72
6.17.2.4 light_proj	72
6.18 LightingPass Class Reference	72
6.18.1 Constructor & Destructor Documentation	73
6.18.1.1 LightingPass()	73
6.18.1.2 ~LightingPass()	73
6.18.2 Member Function Documentation	73
6.18.2.1 execute()	73
6.18.2.2 init()	74
6.19 LightingPassShaderProgram Class Reference	74

6.19.1 Constructor & Destructor Documentation	77
6.19.1.1 LightingPassShaderProgram()	77
6.19.1.2 ~LightingPassShaderProgram()	77
6.19.2 Member Function Documentation	78
6.19.2.1 get_cascade_endpoint_location()	78
6.19.2.2 get_cirrus_effect_location()	78
6.19.2.3 get_cloud_powderness_effect()	78
6.19.2.4 get_directional_light_ambient_intensity_location()	78
6.19.2.5 get_directional_light_color_location()	78
6.19.2.6 get_directional_light_diffuse_intensity_location()	78
6.19.2.7 get_directional_light_direction_location()	78
6.19.2.8 get_directional_light_shadow_intensity_location()	79
6.19.2.9 get_directional_light_transform_location()	79
6.19.2.10 get_directional_shadow_map_location()	79
6.19.2.11 get_eye_position_location()	79
6.19.2.12 get_g_albedo_location()	79
6.19.2.13 get_g_clouds_location()	79
6.19.2.14 get_g_directional_light_position_location()	79
6.19.2.15 get_g_frag_depth_location()	80
6.19.2.16 get_g_normal_location()	80
6.19.2.17 get_g_position_location()	80
6.19.2.18 get_random_number_location()	80
6.19.2.19 get_uniform_absorption_location()	80
6.19.2.20 get_uniform_cloud_model()	80
6.19.2.21 get_uniform_cloud_offset()	80
6.19.2.22 get_uniform_cloud_rad_location()	81
6.19.2.23 get_uniform_cloud_scale_location()	81
6.19.2.24 get_uniform_cloud_threshold_location()	81
6.19.2.25 get_uniform_IOR_location()	81
6.19.2.26 get_uniform_material_id_location()	81
6.19.2.27 get_uniform_material_metallic_location()	81
6.19.2.28 get_uniform_material_roughness_location()	81
6.19.2.29 get_uniform_num_active_cascades_location()	82
6.19.2.30 get_uniform.omni_dir_shadow_map_location()	82
6.19.2.31 get_uniform_pcf_radius_location()	82
6.19.2.32 get_uniform_pillowness_location()	82
6.19.2.33 get_uniform_point_light_far_plane_location()	82
6.19.2.34 set_cloud_texture()	82
6.19.2.35 set_noise_textures()	83
6.19.2.36 set_point_lights()	83
6.19.3 Member Data Documentation	83
6.19.3.1 uniform_absorption_coeff_location	83

6.19.3.2 uniform_ambient_intensity	84
6.19.3.3 uniform_color	84
6.19.3.4 uniform_constant	84
6.19.3.5 uniform_diffuse_intensity	84
6.19.3.6 uniform_exponent	84
6.19.3.7 uniform_far_plane	84
6.19.3.8 uniform_IOR_location	84
6.19.3.9 uniform_linear	84
6.19.3.10 uniform_metallic_location	85
6.19.3.11 uniform_position	85
6.19.3.12 uniform_roughness_location	85
6.19.3.13 uniform_shadow_map	85
6.20 LoadingScreenShaderProgram Class Reference	85
6.20.1 Constructor & Destructor Documentation	88
6.20.1.1 LoadingScreenShaderProgram()	88
6.20.1.2 ~LoadingScreenShaderProgram()	88
6.21 Material Class Reference	88
6.21.1 Constructor & Destructor Documentation	89
6.21.1.1 Material() [1/2]	89
6.21.1.2 Material() [2/2]	89
6.21.1.3 ~Material()	89
6.21.2 Member Function Documentation	89
6.21.2.1 use_material()	89
6.22 Mesh Class Reference	90
6.22.1 Constructor & Destructor Documentation	90
6.22.1.1 Mesh() [1/2]	90
6.22.1.2 Mesh() [2/2]	91
6.22.1.3 ~Mesh()	91
6.22.2 Member Function Documentation	91
6.22.2.1 changeVertex()	91
6.22.2.2 expand()	92
6.22.2.3 getIndices()	92
6.22.2.4 getVertices()	92
6.22.2.5 render()	92
6.22.2.6 transform_Mesh()	93
6.23 MirroredRepeatMode Class Reference	93
6.23.1 Constructor & Destructor Documentation	94
6.23.1.1 MirroredRepeatMode()	94
6.23.1.2 ~MirroredRepeatMode()	94
6.23.2 Member Function Documentation	94
6.23.2.1 activate()	95
6.24 Model Class Reference	95

6.24.1 Constructor & Destructor Documentation	95
6.24.1.1 Model()	96
6.24.1.2 ~Model()	96
6.24.2 Member Function Documentation	96
6.24.2.1 create_render_context()	96
6.24.2.2 get_aabb()	96
6.24.2.3 get_base_dir()	96
6.24.2.4 load_model_in_ram()	97
6.24.2.5 render()	97
6.24.2.6 transform_model()	97
6.25 MyWindow Class Reference	98
6.25.1 Constructor & Destructor Documentation	98
6.25.1.1 MyWindow() [1/2]	99
6.25.1.2 MyWindow() [2/2]	99
6.25.1.3 ~MyWindow()	99
6.25.2 Member Function Documentation	99
6.25.2.1 get_buffer_height()	99
6.25.2.2 get_buffer_width()	99
6.25.2.3 get_keys()	99
6.25.2.4 get_should_close()	99
6.25.2.5 get_window()	100
6.25.2.6 get_x_change()	100
6.25.2.7 get_y_change()	100
6.25.2.8 initialized()	100
6.25.2.9 set_buffer_size()	100
6.25.2.10 swap_buffers()	100
6.25.2.11 update_viewport()	101
6.26 Noise Class Reference	101
6.26.1 Constructor & Destructor Documentation	101
6.26.1.1 Noise()	102
6.26.1.2 ~Noise()	102
6.26.2 Member Function Documentation	102
6.26.2.1 create_grad_noise()	102
6.26.2.2 create_worley_noise()	103
6.26.2.3 init()	103
6.26.2.4 read_grad_noise()	104
6.26.2.5 read_worley_noise()	104
6.26.2.6 set_num_cells()	104
6.26.2.7 update()	105
6.27 OmniDirShadowMap Class Reference	105
6.27.1 Constructor & Destructor Documentation	106
6.27.1.1 OmniDirShadowMap()	106

6.27.1.2 ~OmniDirShadowMap()	107
6.27.2 Member Function Documentation	107
6.27.2.1 init()	107
6.27.2.2 read()	107
6.27.2.3 write()	107
6.28 OmniDirShadowShaderProgram Class Reference	108
6.28.1 Constructor & Destructor Documentation	110
6.28.1.1 OmniDirShadowShaderProgram()	110
6.28.1.2 ~OmniDirShadowShaderProgram()	110
6.28.2 Member Function Documentation	110
6.28.2.1 get_far_plane_location()	110
6.28.2.2 get_model_location()	110
6.28.2.3 get_omni_light_pos_location()	110
6.28.2.4 reload()	111
6.28.2.5 set_light_matrices()	111
6.29 OmniShadowMapPass Class Reference	111
6.29.1 Constructor & Destructor Documentation	113
6.29.1.1 OmniShadowMapPass() [1/2]	113
6.29.1.2 OmniShadowMapPass() [2/2]	114
6.29.1.3 ~OmniShadowMapPass()	114
6.29.2 Member Function Documentation	114
6.29.2.1 execute()	114
6.29.2.2 set_game_object_uniforms()	114
6.29.2.3 use_terrain_textures()	114
6.30 PointLight Class Reference	115
6.30.1 Constructor & Destructor Documentation	117
6.30.1.1 PointLight() [1/2]	117
6.30.1.2 PointLight() [2/2]	117
6.30.1.3 ~PointLight()	117
6.30.2 Member Function Documentation	117
6.30.2.1 calculate_light_transform()	118
6.30.2.2 get_far_plane()	118
6.30.2.3 get_omni_shadow_map()	118
6.30.2.4 get_position()	118
6.30.2.5 set_position()	118
6.30.2.6 use_light()	118
6.30.3 Member Data Documentation	118
6.30.3.1 constant	119
6.30.3.2 exponent	119
6.30.3.3 far_plane	119
6.30.3.4 linear	119
6.30.3.5 omni_dir_shadow_map	119

6.30.3.6 position	119
6.31 Quad Class Reference	120
6.31.1 Constructor & Destructor Documentation	120
6.31.1.1 Quad()	120
6.31.1.2 ~Quad()	120
6.31.2 Member Function Documentation	120
6.31.2.1 init()	121
6.31.2.2 render()	121
6.32 RenderPassSceneDependend Class Reference	121
6.32.1 Constructor & Destructor Documentation	122
6.32.1.1 RenderPassSceneDependend()	122
6.32.1.2 ~RenderPassSceneDependend()	123
6.32.2 Member Function Documentation	123
6.32.2.1 set_game_object_uniforms()	123
6.32.2.2 use_terrain_textures()	123
6.33 RepeatMode Class Reference	124
6.33.1 Constructor & Destructor Documentation	125
6.33.1.1 RepeatMode()	125
6.33.1.2 ~RepeatMode()	125
6.33.2 Member Function Documentation	125
6.33.2.1 activate()	125
6.34 Rotation Struct Reference	125
6.34.1 Member Data Documentation	126
6.34.1.1 axis	126
6.34.1.2 degrees	126
6.35 Scene Class Reference	126
6.35.1 Constructor & Destructor Documentation	128
6.35.1.1 Scene() [1/2]	128
6.35.1.2 Scene() [2/2]	128
6.35.1.3 ~Scene()	128
6.35.2 Member Function Documentation	128
6.35.2.1 add_ambient_object()	128
6.35.2.2 add_space_ship()	128
6.35.2.3 get_clouds()	129
6.35.2.4 get_context_setup()	129
6.35.2.5 get_position_of_current_ship()	129
6.35.2.6 get_progress()	129
6.35.2.7 get_terrain_generator()	129
6.35.2.8 init()	129
6.35.2.9 is_loaded()	129
6.35.2.10 load_models()	130
6.35.2.11 render()	130

6.35.2.12 set_context_setup()	130
6.35.2.13 setup_game_object_context()	130
6.35.2.14 spwan()	130
6.35.2.15 update_current_space_ship()	131
6.36 ShaderProgram Class Reference	131
6.36.1 Constructor & Destructor Documentation	133
6.36.1.1 ShaderProgram()	133
6.36.1.2 ~ShaderProgram()	133
6.36.2 Member Function Documentation	133
6.36.2.1 add_shader()	134
6.36.2.2 clear_shader_program()	134
6.36.2.3 compile_compute_shader_program()	134
6.36.2.4 compile_program()	135
6.36.2.5 compile_shader_program() [1/2]	135
6.36.2.6 compile_shader_program() [2/2]	136
6.36.2.7 create_computer_shader_program_from_file()	137
6.36.2.8 create_from_files() [1/2]	137
6.36.2.9 create_from_files() [2/2]	138
6.36.2.10 read_file()	138
6.36.2.11 reload()	139
6.36.2.12 retrieve_uniform_locations()	139
6.36.2.13 use_shader_program()	139
6.36.2.14 validate_program()	140
6.36.3 Member Data Documentation	140
6.36.3.1 compute_location	140
6.36.3.2 fragment_location	140
6.36.3.3 geometry_location	140
6.36.3.4 gLErrorChecker_ins	141
6.36.3.5 program_id	141
6.36.3.6 vertex_location	141
6.37 ShadowMap Class Reference	141
6.37.1 Constructor & Destructor Documentation	143
6.37.1.1 ShadowMap()	143
6.37.1.2 ~ShadowMap()	144
6.37.2 Member Function Documentation	144
6.37.2.1 get_id()	144
6.37.2.2 get_shadow_height()	144
6.37.2.3 get_shadow_width()	144
6.37.2.4 init()	144
6.37.2.5 read()	144
6.37.2.6 write()	145
6.37.3 Member Data Documentation	145

6.37.3.1 FBO	145
6.37.3.2 shadow_height	145
6.37.3.3 shadow_map	145
6.37.3.4 shadow_width	145
6.38 ShadowMapShaderProgram Class Reference	146
6.38.1 Constructor & Destructor Documentation	148
6.38.1.1 ShadowMapShaderProgram()	148
6.38.1.2 ~ShadowMapShaderProgram()	148
6.38.2 Member Function Documentation	148
6.38.2.1 get_directional_light_transform_location()	148
6.38.2.2 get_model_location()	148
6.38.2.3 set_directional_light_transform()	148
6.39 SkyBox Class Reference	149
6.39.1 Constructor & Destructor Documentation	149
6.39.1.1 SkyBox() [1/2]	149
6.39.1.2 SkyBox() [2/2]	150
6.39.1.3 ~SkyBox()	150
6.39.2 Member Function Documentation	150
6.39.2.1 draw_sky_box()	150
6.39.2.2 reload()	151
6.40 SkyBoxShaderProgram Class Reference	151
6.40.1 Constructor & Destructor Documentation	153
6.40.1.1 SkyBoxShaderProgram()	153
6.40.1.2 ~SkyBoxShaderProgram()	153
6.40.2 Member Function Documentation	153
6.40.2.1 get_projection_location()	153
6.40.2.2 get_view_location()	153
6.41 Terrain_Generator Class Reference	154
6.41.1 Constructor & Destructor Documentation	155
6.41.1.1 Terrain_Generator()	155
6.41.1.2 ~Terrain_Generator()	155
6.41.2 Member Function Documentation	156
6.41.2.1 calc_smooth_normals()	156
6.41.2.2 changeMaxHeight()	156
6.41.2.3 expandTerrain()	156
6.41.2.4 generate_biome()	157
6.41.2.5 generate_indices()	157
6.41.2.6 generate_map_chunk()	157
6.41.2.7 generate_noise_map()	158
6.41.2.8 generate_normals()	159
6.41.2.9 generate_render_context()	160
6.41.2.10 generate_vertices()	160

6.41.2.11 generateMesh()	160
6.41.2.12 get_material_id()	161
6.41.2.13 get_normal_world_trafo()	161
6.41.2.14 get_textures()	161
6.41.2.15 get_world_trafo()	161
6.41.2.16 getNoiseParameters()	161
6.41.2.17 init()	162
6.41.2.18 load_plants()	162
6.41.2.19 regenerate()	162
6.41.2.20 render()	163
6.41.2.21 render_plants()	163
6.41.2.22 retreive_uniform_locations()	164
6.41.2.23 set_material_id()	164
6.41.2.24 set_world_trafo()	165
6.41.2.25 setNoiseParameters()	165
6.41.2.26 transform()	165
6.41.3 Member Data Documentation	165
6.41.3.1 aabbs	165
6.41.3.2 meshes	166
6.42 Terrain_Model Class Reference	166
6.42.1 Member Enumeration Documentation	167
6.42.1.1 M_TYPE	167
6.42.2 Constructor & Destructor Documentation	168
6.42.2.1 Terrain_Model()	168
6.42.3 Member Data Documentation	168
6.42.3.1 angle	168
6.42.3.2 pos	168
6.42.3.3 type	168
6.42.3.4 variation	168
6.42.3.5 xOffset	169
6.42.3.6 yOffset	169
6.43 Terrain_Texture Class Reference	169
6.43.1 Member Enumeration Documentation	170
6.43.1.1 BIOMETYPE	170
6.43.2 Constructor & Destructor Documentation	170
6.43.2.1 Terrain_Texture()	170
6.43.2.2 ~Terrain_Texture()	170
6.43.3 Member Function Documentation	171
6.43.3.1 clear_texture_context()	171
6.43.3.2 load_all_texture()	171
6.43.3.3 load_texture()	171
6.43.3.4 retreive_uniform_locations()	172

6.43.3.5 setHeights()	172
6.43.3.6 unbind_texture()	173
6.43.3.7 use_texture()	173
6.44 terrainType Struct Reference	174
6.44.1 Constructor & Destructor Documentation	174
6.44.1.1 terrainType()	174
6.44.2 Member Data Documentation	174
6.44.2.1 height	174
6.44.2.2 texCoord	175
6.45 Texture Class Reference	175
6.45.1 Constructor & Destructor Documentation	176
6.45.1.1 Texture() [1/2]	176
6.45.1.2 Texture() [2/2]	176
6.45.1.3 ~Texture()	176
6.45.2 Member Function Documentation	176
6.45.2.1 clear_texture_context()	176
6.45.2.2 get_filename()	177
6.45.2.3 get_id()	177
6.45.2.4 load_texture_with_alpha_channel()	177
6.45.2.5 load_texture_without_alpha_channel()	177
6.45.2.6 unbind_texture()	178
6.45.2.7 use_texture()	178
6.46 TextureWrappingMode Class Reference	178
6.46.1 Constructor & Destructor Documentation	179
6.46.1.1 TextureWrappingMode()	179
6.46.1.2 ~TextureWrappingMode()	179
6.46.2 Member Function Documentation	179
6.46.2.1 activate()	179
6.47 Vertex Struct Reference	180
6.47.1 Constructor & Destructor Documentation	180
6.47.1.1 Vertex() [1/2]	180
6.47.1.2 Vertex() [2/2]	181
6.47.2 Member Function Documentation	181
6.47.2.1 get_normal()	181
6.47.2.2 get_position()	181
6.47.2.3 get_tex_coors()	181
6.47.2.4 operator==()	181
6.47.3 Member Data Documentation	181
6.47.3.1 normal	181
6.47.3.2 position	182
6.47.3.3 texture_coords	182
6.48 ViewFrustumCulling Class Reference	182

6.48.1 Constructor & Destructor Documentation	182
6.48.1.1 ViewFrustumCulling()	183
6.48.1.2 ~ViewFrustumCulling()	183
6.48.2 Member Function Documentation	183
6.48.2.1 is_inside()	183
6.48.2.2 render_view_frustum()	183
7 File Documentation	185
7.1 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.cpp File Reference	185
7.2 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.h File Reference	185
7.3 AABB.h	186
7.4 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.cpp File Reference	187
7.5 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.h File Reference	187
7.6 Camera.h	188
7.7 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.cpp File Reference	189
7.8 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.h File Reference	189
7.9 CascadedShadowMap.h	190
7.10 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.cpp File Reference	191
7.11 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.h File Reference	191
7.12 ClampToEdgeMode.h	192
7.13 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.cpp File Reference	193
7.14 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.h File Reference	193
7.15 Clouds.h	194
7.16 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.cpp File Reference	196
7.17 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.h File Reference	196
7.18 CloudsShaderProgram.h	197
7.19 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.cpp File Reference	198
7.20 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.h File Reference	198
7.21 ComputeShaderProgram.h	199
7.22 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.cpp File Reference	200
7.23 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.h File Reference	200
7.24 DirectionalLight.h	201
7.25 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLightUniformLocations.h File Reference	202
7.26 DirectionalLightUniformLocations.h	202
7.27 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.cpp File Reference	203
7.28 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.h File Reference	203
7.29 DirectionalShadowMapPass.h	204
7.30 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.cpp File Reference	205
7.31 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.h File Reference	205
7.32 GameObject.h	206
7.33 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.cpp File Reference	206
7.34 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.h File Reference	207

7.35 GBuffer.h	207
7.36 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.cpp File Reference	208
7.37 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.h File Reference	208
7.38 GeometryPass.h	209
7.39 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShaderProgram.cpp File Reference	210
7.40 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShaderProgram.h File Reference	211
7.41 GeometryPassShaderProgram.h	212
7.42 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/glErrorChecker.h File Reference	212
7.43 glErrorChecker.h	213
7.44 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GlobalValues.h File Reference	214
7.44.1 Macro Definition Documentation	215
7.44.1.1 COMMONVALS	215
7.44.1.2 MAX_RESOLUTION_X	215
7.44.1.3 MAX_RESOLUTION_Y	215
7.44.1.4 NUM_BIOM_TEXTURES	215
7.44.1.5 NUM_MAX_CASCADES	215
7.44.1.6 NUM_MIN_CASCADES	216
7.44.2 Variable Documentation	216
7.44.2.1 G_BUFFER_SIZE	216
7.44.2.2 MAX_MATERIALS	216
7.44.2.3 MAX_POINT_LIGHTS	216
7.44.2.4 NUM_CELLS	216
7.44.2.5 NUM_CLOUDS	216
7.44.2.6 NUM_FRUSTUM_PLANES	216
7.44.2.7 NUM_NOISE_TEXTURES	217
7.45 GlobalValues.h	217
7.46 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GraphicsEngine.cpp File Reference	217
7.46.1 Macro Definition Documentation	220
7.46.1.1 STB_IMAGE_IMPLEMENTATION	220
7.46.1.2 TINYOBJLOADER_IMPLEMENTATION	220
7.46.2 Function Documentation	220
7.46.2.1 create_geometry_pass_shader_program()	220
7.46.2.2 create_lighting_pass_shader_program()	220
7.46.2.3 create_loading_screen_shader_program()	221
7.46.2.4 create_noise_textures()	221
7.46.2.5 create_omni_shadow_map_shader_program()	221
7.46.2.6 create_shader_programs()	222
7.46.2.7 create_shadow_map_shader_program()	222
7.46.2.8 main()	223
7.46.2.9 materials()	223
7.46.2.10 point_lights()	223
7.46.2.11 reload_noise_programs()	224

7.46.2.12 reload_shader_programs()	224
7.46.3 Variable Documentation	224
7.46.3.1 available_shadow_map_resolutions	224
7.46.3.2 cascaded_shadow_intensity	225
7.46.3.3 choosen_space_ship	225
7.46.3.4 cloud_cirrus_effect	225
7.46.3.5 cloud_density	225
7.46.3.6 cloud_mesh_scale	225
7.46.3.7 cloud_movement_direction	225
7.46.3.8 cloud_pillowness	225
7.46.3.9 cloud_powder_effect	225
7.46.3.10 cloud_scale	226
7.46.3.11 cloud_speed	226
7.46.3.12 clouds	226
7.46.3.13 delta_time	226
7.46.3.14 directional_light_starting_color	226
7.46.3.15 directional_light_starting_position	226
7.46.3.16 directional_shadow_map_pass	226
7.46.3.17 far_plane	226
7.46.3.18 far_plane_shadow	227
7.46.3.19 fov	227
7.46.3.20 g_buffer_geometry_pass_shader_program	227
7.46.3.21 g_buffer_lighting_pass_shader_program	227
7.46.3.22 gbuffer	227
7.46.3.23 geometry_pass	227
7.46.3.24 last_time	227
7.46.3.25 lighting_pass	227
7.46.3.26 loading_screen	228
7.46.3.27 loading_screen_finished	228
7.46.3.28 loading_screen_shader_program	228
7.46.3.29 loading_screen_tex	228
7.46.3.30 logo	228
7.46.3.31 main_camera	228
7.46.3.32 main_light	228
7.46.3.33 material_counter	228
7.46.3.34 near_plane	229
7.46.3.35 noise	229
7.46.3.36 num_shadow_cascades	229
7.46.3.37 omni_dir_shadow_shader_program	229
7.46.3.38 omni_shadow_map_pass	229
7.46.3.39 ornament1	229
7.46.3.40 pcf_radius	229

7.46.3.41 point_light_count	229
7.46.3.42 shadow_map_res_index	230
7.46.3.43 shadow_map_resolution	230
7.46.3.44 shadow_map_shader_program	230
7.46.3.45 shadow_resolution_changed	230
7.46.3.46 sound_volume	230
7.46.3.47 ssao_enabled	230
7.46.3.48 ssr_enabled	230
7.46.3.49 terrain_height	230
7.46.3.50 tGenerator	231
7.47 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.cpp File Reference	231
7.48 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.h File Reference	231
7.49 Light.h	232
7.50 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.cpp File Reference	233
7.51 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.h File Reference	233
7.52 LightingPass.h	234
7.53 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.cpp File Reference	235
7.54 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.h File Reference .	235
7.55 LightingPassShaderProgram.h	236
7.56 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.cpp File Reference	238
7.57 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.h File Reference	239
7.58 LoadingScreenShaderProgram.h	239
7.59 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.cpp File Reference	240
7.60 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.h File Reference	240
7.61 Material.h	241
7.62 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.cpp File Reference	242
7.63 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.h File Reference	242
7.64 Mesh.h	243
7.65 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.cpp File Reference . . .	244
7.66 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.h File Reference . . .	245
7.67 MirroredRepeatMode.h	246
7.68 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.cpp File Reference	246
7.69 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.h File Reference	246
7.70 Model.h	247
7.71 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.cpp File Reference	248
7.72 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.h File Reference	249
7.73 MyWindow.h	249
7.74 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.cpp File Reference	251
7.75 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.h File Reference	251
7.76 Noise.h	252
7.77 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.cpp File Reference . . .	253
7.78 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.h File Reference . . .	254

7.79 OmniDirShadowMap.h	255
7.80 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.cpp File Reference	255
7.81 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.h File Reference	256
7.82 OmniDirShadowShaderProgram.h	257
7.83 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.cpp File Reference	257
7.84 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.h File Reference	257
7.85 OmniShadowMapPass.h	258
7.86 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.cpp File Reference	259
7.87 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.h File Reference	259
7.87.1 Function Documentation	260
7.87.1.1 fade()	260
7.87.1.2 getPermutationVector()	261
7.87.1.3 grad()	261
7.87.1.4 lerp()	261
7.87.1.5 perlin_noise()	262
7.88 Perlin.h	262
7.89 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.cpp File Reference	264
7.90 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.h File Reference	264
7.91 PointLight.h	265
7.92 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.cpp File Reference	266
7.93 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.h File Reference	266
7.94 Quad.h	267
7.95 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.cpp File Reference	268
7.96 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.h File Reference	268
7.97 RenderPassSceneDependend.h	269
7.98 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.cpp File Reference	270
7.99 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.h File Reference	271
7.100 RepeatMode.h	272
7.101 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource.h File Reference	272
7.101.1 Macro Definition Documentation	272
7.101.1.1 IDI_ICON1	272
7.102 resource.h	272
7.103 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource1.h File Reference	273
7.104 resource1.h	273
7.105 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.cpp File Reference	273
7.106 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.h File Reference	273
7.107 Scene.h	274
7.108 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.cpp File Reference	276
7.109 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.h File Reference	276
7.110 ShaderProgram.h	277
7.111 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMap.cpp File Reference	278

7.112 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMap.h File Reference	278
7.113 ShadowMap.h	279
7.114 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.cpp File Reference	280
7.115 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.h File Reference .	281
7.116 ShadowMapShaderProgram.h	281
7.117 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.cpp File Reference	282
7.118 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.h File Reference	282
7.119 SkyBox.h	283
7.120 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.cpp File Reference .	284
7.121 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.h File Reference . .	285
7.122 SkyBoxShaderProgram.h	286
7.123 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp File Reference . . .	286
7.124 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.h File Reference	287
7.125 Terrain_generator.h	288
7.126 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_Model.h File Reference	290
7.127 Terrain_Model.h	290
7.128 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.cpp File Reference	291
7.129 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.h File Reference	292
7.130 Terrain_texture.h	292
7.131 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.cpp File Reference	293
7.132 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.h File Reference	294
7.133 Texture.h	295
7.134 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.cpp File Reference .	296
7.135 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.h File Reference . .	296
7.136 TextureWrappingMode.h	297
7.137 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Vertex.h File Reference	297
7.138 Vertex.h	299
7.139 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.cpp File Reference . .	300
7.140 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.h File Reference . . .	300
7.141 ViewFrustumCulling.h	301
Index	303

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

std	9
-------------------------------	-------------------

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

AABB	11
Camera	14
CascadedShadowMap	18
Clouds	25
DirectionalLightUniformLocations	46
GameObject	50
GBuffer	55
glErrorChecker	67
std::hash< Vertex >	69
Light	70
DirectionalLight	39
PointLight	115
LightingPass	72
Material	88
Mesh	90
Model	95
MyWindow	98
Noise	101
Quad	120
RenderPassSceneDependend	121
DirectionalShadowMapPass	47
GeometryPass	57
OmniShadowMapPass	111
Rotation	125
Scene	126
ShaderProgram	131
CloudsShaderProgram	31
ComputeShaderProgram	35
GeometryPassShaderProgram	60
LightingPassShaderProgram	74
LoadingScreenShaderProgram	85
OmniDirShadowShaderProgram	108
ShadowMapShaderProgram	146
SkyBoxShaderProgram	151

ShadowMap	141
OmniDirShadowMap	105
SkyBox	149
Terrain_Generator	154
Terrain_Model	166
Terrain_Texture	169
terrainType	174
Texture	175
TextureWrappingMode	178
ClampToEdgeMode	23
MirroredRepeatMode	93
RepeatMode	124
Vertex	180
ViewFrustumCulling	182

Chapter 3

Class Index

3.1 Class List

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

AABB	11
Camera	14
CascadedShadowMap	18
ClampToEdgeMode	23
Clouds	25
CloudsShaderProgram	31
ComputeShaderProgram	35
DirectionalLight	39
DirectionalLightUniformLocations	46
DirectionalShadowMapPass	47
GameObject	50
GBuffer	55
GeometryPass	57
GeometryPassShaderProgram	60
glErrorChecker	67
std::hash< Vertex >	69
Light	70
LightingPass	72
LightingPassShaderProgram	74
LoadingScreenShaderProgram	85
Material	88
Mesh	90
MirroredRepeatMode	93
Model	95
MyWindow	98
Noise	101
OmniDirShadowMap	105
OmniDirShadowShaderProgram	108
OmniShadowMapPass	111
PointLight	115
Quad	120
RenderPassSceneDependend	121
RepeatMode	124
Rotation	125
Scene	126

ShaderProgram	131
ShadowMap	141
ShadowMapShaderProgram	146
SkyBox	149
SkyBoxShaderProgram	151
Terrain_Generator	154
Terrain_Model	166
Terrain_Texture	169
terrainType	174
Texture	175
TextureWrappingMode	178
Vertex	180
ViewFrustumCulling	182

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.cpp	185
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.h	185
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.cpp	187
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.h	187
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.cpp	189
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.h	189
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.cpp	191
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.h	191
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.cpp	193
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.h	193
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.cpp	196
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.h	196
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.cpp	198
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.h	198
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.cpp	200
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.h	200
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLightUniformLocations.h	202
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.cpp	203
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.h	203
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.cpp	205
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.h	205
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.cpp	206
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.h	207
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.cpp	208
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.h	208
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShaderProgram.cpp	210
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShaderProgram.h	211
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/gLErrorChecker.h	212
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GlobalValues.h	214
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GraphicsEngine.cpp	217
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.cpp	231
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.h	231
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.cpp	233
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.h	233
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.cpp	235

C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.h	235
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.cpp	238
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.h	239
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.cpp	240
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.h	240
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.cpp	242
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.h	242
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.cpp	244
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.h	245
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.cpp	246
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.h	246
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.cpp	248
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.h	249
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.cpp	251
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.h	251
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.cpp	253
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.h	254
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.cpp	255
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.h	256
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.cpp	257
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.h	257
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.cpp	259
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.h	259
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.cpp	264
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.h	264
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.cpp	266
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.h	266
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.cpp	268
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.h	268
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.cpp	270
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.h	271
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource.h	272
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource1.h	273
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.cpp	273
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.h	273
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.cpp	276
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.h	276
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMap.cpp	278
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMap.h	278
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.cpp	280
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.h	281
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.cpp	282
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.h	282
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.cpp	284
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.h	285
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp	286
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.h	287
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_Model.h	290
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.cpp	291
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.h	292
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.cpp	293
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.h	294
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.cpp	296
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.h	296
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Vertex.h	297
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.cpp	300
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.h	300

Chapter 5

Namespace Documentation

5.1 std Namespace Reference

Classes

- struct [hash< Vertex >](#)

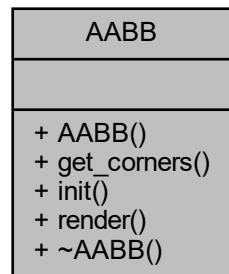
Chapter 6

Class Documentation

6.1 AABB Class Reference

```
#include <AABB.h>
```

Collaboration diagram for AABB:



Public Member Functions

- `AABB ()`
- `std::vector< glm::vec3 > get_corners (glm::mat4 model)`
- `void init (GLfloat minX, GLfloat maxX, GLfloat minY, GLfloat maxY, GLfloat minZ, GLfloat maxZ)`
- `void render ()`
- `~AABB ()`

6.1.1 Constructor & Destructor Documentation

6.1.1.1 **AABB()**

```
AABB::AABB ( )
```

6.1.1.2 **~AABB()**

```
AABB::~AABB ( )
```

6.1.2 Member Function Documentation

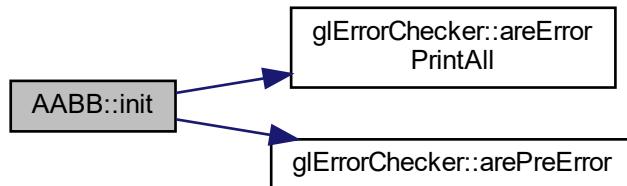
6.1.2.1 **get_corners()**

```
std::vector< glm::vec3 > AABB::get_corners (
    glm::mat4 model )
```

6.1.2.2 **init()**

```
void AABB::init (
    GLfloat minX,
    GLfloat maxX,
    GLfloat minY,
    GLfloat maxY,
    GLfloat minZ,
    GLfloat maxZ )
```

Here is the call graph for this function:



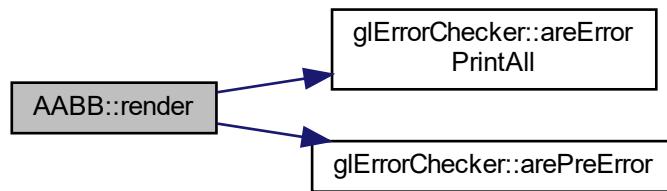
Here is the caller graph for this function:



6.1.2.3 render()

```
void AABB::render ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



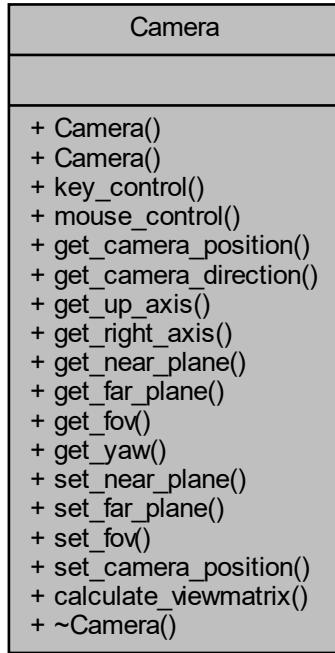
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[AABB.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[AABB.cpp](#)

6.2 Camera Class Reference

```
#include <Camera.h>
```

Collaboration diagram for Camera:



Public Member Functions

- [Camera \(\)](#)
- [Camera \(glm::vec3 start_position, glm::vec3 start_up, GLfloat start_yaw, GLfloat start_pitch, GLfloat start_move_speed, GLfloat start_turn_speed, GLfloat near_plane, GLfloat far_plane, GLfloat fov\)](#)
- [void key_control \(bool *keys, GLfloat delta_time\)](#)
- [void mouse_control \(GLfloat x_change, GLfloat y_change\)](#)
- [glm::vec3 get_camera_position \(\)](#)
- [glm::vec3 get_camera_direction \(\)](#)
- [glm::vec3 get_up_axis \(\)](#)
- [glm::vec3 get_right_axis \(\)](#)
- [GLfloat get_near_plane \(\)](#)
- [GLfloat get_far_plane \(\)](#)
- [GLfloat get_fov \(\)](#)
- [GLfloat get_yaw \(\)](#)
- [void set_near_plane \(GLfloat near_plane\)](#)
- [void set_far_plane \(GLfloat far_plane\)](#)
- [void set_fov \(GLfloat fov\)](#)
- [void set_camera_position \(glm::vec3 new_camera_position\)](#)
- [glm::mat4 calculate_viewmatrix \(\)](#)
- [~Camera \(\)](#)

6.2.1 Constructor & Destructor Documentation

6.2.1.1 Camera() [1/2]

```
Camera::Camera ( )
```

6.2.1.2 Camera() [2/2]

```
Camera::Camera (
    glm::vec3 start_position,
    glm::vec3 start_up,
    GLfloat start_yaw,
    GLfloat start_pitch,
    GLfloat start_move_speed,
    GLfloat start_turn_speed,
    GLfloat near_plane,
    GLfloat far_plane,
    GLfloat fov )
```

6.2.1.3 ~Camera()

```
Camera::~Camera ( )
```

6.2.2 Member Function Documentation

6.2.2.1 calculate_viewmatrix()

```
glm::mat4 Camera::calculate_viewmatrix ( )
```

6.2.2.2 get_camera_direction()

```
glm::vec3 Camera::get_camera_direction ( )
```

6.2.2.3 `get_camera_position()`

```
glm::vec3 Camera::get_camera_position ( )
```

6.2.2.4 `get_far_plane()`

```
GLfloat Camera::get_far_plane ( )
```

6.2.2.5 `get_fov()`

```
GLfloat Camera::get_fov ( )
```

6.2.2.6 `get_near_plane()`

```
GLfloat Camera::get_near_plane ( )
```

6.2.2.7 `get_right_axis()`

```
glm::vec3 Camera::get_right_axis ( )
```

6.2.2.8 `get_up_axis()`

```
glm::vec3 Camera::get_up_axis ( )
```

6.2.2.9 `get_yaw()`

```
GLfloat Camera::get_yaw ( )
```

6.2.2.10 key_control()

```
void Camera::key_control (
    bool * keys,
    GLfloat delta_time )
```

6.2.2.11 mouse_control()

```
void Camera::mouse_control (
    GLfloat x_change,
    GLfloat y_change )
```

6.2.2.12 set_camera_position()

```
void Camera::set_camera_position (
    glm::vec3 new_camera_position )
```

6.2.2.13 set_far_plane()

```
void Camera::set_far_plane (
    GLfloat far_plane )
```

6.2.2.14 set_fov()

```
void Camera::set_fov (
    GLfloat fov )
```

6.2.2.15 set_near_plane()

```
void Camera::set_near_plane (
    GLfloat near_plane )
```

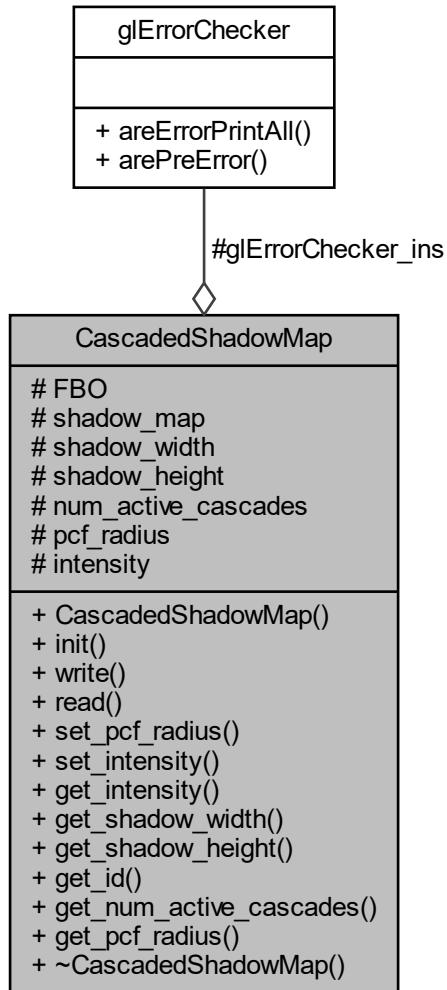
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Camera.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Camera.cpp](#)

6.3 CascadedShadowMap Class Reference

```
#include <CascadedShadowMap.h>
```

Collaboration diagram for CascadedShadowMap:



Public Member Functions

- [`CascadedShadowMap \(\)`](#)
- [`virtual bool init \(GLuint width, GLuint height, GLuint num_cascades\)`](#)
- [`virtual void write \(GLuint cascade_index\)`](#)
- [`virtual void read \(GLenum texture_unit\)`](#)
- [`void set_pcf_radius \(GLuint radius\)`](#)
- [`void set_intensity \(GLfloat intensity\)`](#)
- [`GLfloat get_intensity \(\)`](#)
- [`GLuint get_shadow_width \(\)`](#)

- GLuint `get_shadow_height()`
- GLuint `get_id(GLuint cascade_index)`
- GLuint `get_num_active_cascades()`
- GLuint `get_pcf_radius()`
- `~CascadedShadowMap()`

Protected Attributes

- GLuint `FBO`
- GLuint `shadow_map [NUM_MAX_CASCADES]`
- GLuint `shadow_width`
- GLuint `shadow_height`
- GLuint `num_active_cascades`
- GLuint `pcf_radius`
- GLfloat `intensity`
- `glErrorChecker glErrorChecker_ins`

6.3.1 Constructor & Destructor Documentation

6.3.1.1 CascadedShadowMap()

```
CascadedShadowMap::CascadedShadowMap( )
```

6.3.1.2 ~CascadedShadowMap()

```
CascadedShadowMap::~CascadedShadowMap( )
```

6.3.2 Member Function Documentation

6.3.2.1 get_id()

```
GLuint CascadedShadowMap::get_id(   
    GLuint cascade_index )
```

6.3.2.2 get_intensity()

```
GLfloat CascadedShadowMap::get_intensity( )
```

6.3.2.3 `get_num_active_cascades()`

```
GLuint CascadedShadowMap::get_num_active_cascades ( )
```

6.3.2.4 `get_pcf_radius()`

```
GLuint CascadedShadowMap::get_pcf_radius ( )
```

6.3.2.5 `get_shadow_height()`

```
GLuint CascadedShadowMap::get_shadow_height ( ) [inline]
```

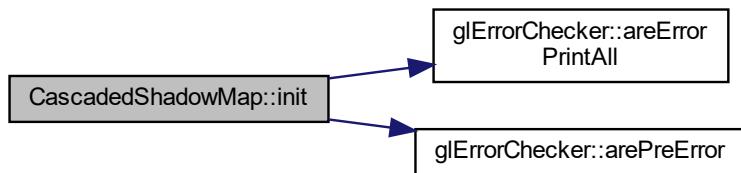
6.3.2.6 `get_shadow_width()`

```
GLuint CascadedShadowMap::get_shadow_width ( ) [inline]
```

6.3.2.7 `init()`

```
bool CascadedShadowMap::init (
    GLuint width,
    GLuint height,
    GLuint num_cascades ) [virtual]
```

Here is the call graph for this function:



6.3.2.8 `read()`

```
void CascadedShadowMap::read (
    GLenum texture_unit ) [virtual]
```

6.3.2.9 `set_intensity()`

```
void CascadedShadowMap::set_intensity (
    GLfloat intensity )
```

6.3.2.10 `set_pcf_radius()`

```
void CascadedShadowMap::set_pcf_radius (
    GLuint radius )
```

6.3.2.11 `write()`

```
void CascadedShadowMap::write (
    GLuint cascade_index ) [virtual]
```

6.3.3 Member Data Documentation

6.3.3.1 `FBO`

```
GLuint CascadedShadowMap::FBO [protected]
```

6.3.3.2 `glErrorChecker_ins`

```
glErrorChecker CascadedShadowMap::glErrorChecker_ins [protected]
```

6.3.3.3 `intensity`

```
GLfloat CascadedShadowMap::intensity [protected]
```

6.3.3.4 num_active_cascades

```
GLuint CascadedShadowMap::num_active_cascades [protected]
```

6.3.3.5 pcf_radius

```
GLuint CascadedShadowMap::pcf_radius [protected]
```

6.3.3.6 shadow_height

```
GLuint CascadedShadowMap::shadow_height [protected]
```

6.3.3.7 shadow_map

```
GLuint CascadedShadowMap::shadow_map [NUM_MAX_CASCADES] [protected]
```

6.3.3.8 shadow_width

```
GLuint CascadedShadowMap::shadow_width [protected]
```

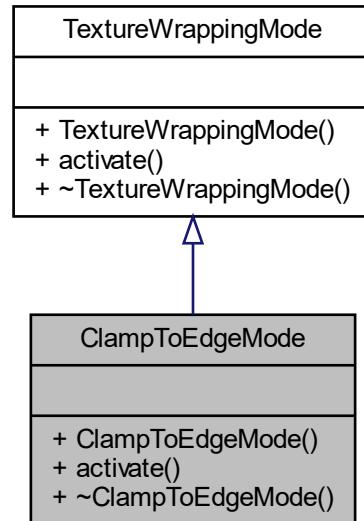
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[CascadedShadowMap.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[CascadedShadowMap.cpp](#)

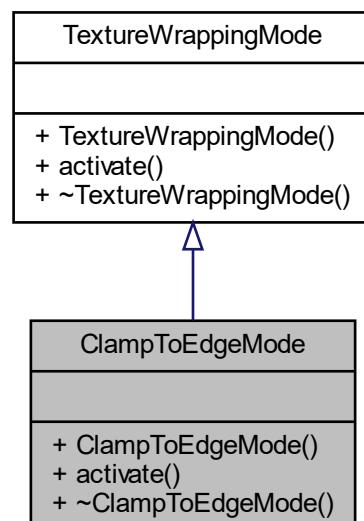
6.4 ClampToEdgeMode Class Reference

```
#include <ClampToEdgeMode.h>
```

Inheritance diagram for ClampToEdgeMode:



Collaboration diagram for ClampToEdgeMode:



Public Member Functions

- [ClampToEdgeMode \(\)](#)
- [void activate \(\)](#)
- [~ClampToEdgeMode \(\)](#)

6.4.1 Constructor & Destructor Documentation

6.4.1.1 ClampToEdgeMode()

```
ClampToEdgeMode::ClampToEdgeMode ( )
```

6.4.1.2 ~ClampToEdgeMode()

```
ClampToEdgeMode::~ClampToEdgeMode ( )
```

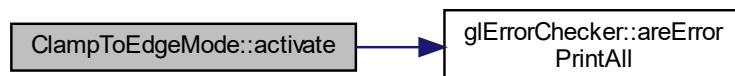
6.4.2 Member Function Documentation

6.4.2.1 activate()

```
void ClampToEdgeMode::activate ( ) [virtual]
```

Implements [TextureWrappingMode](#).

Here is the call graph for this function:



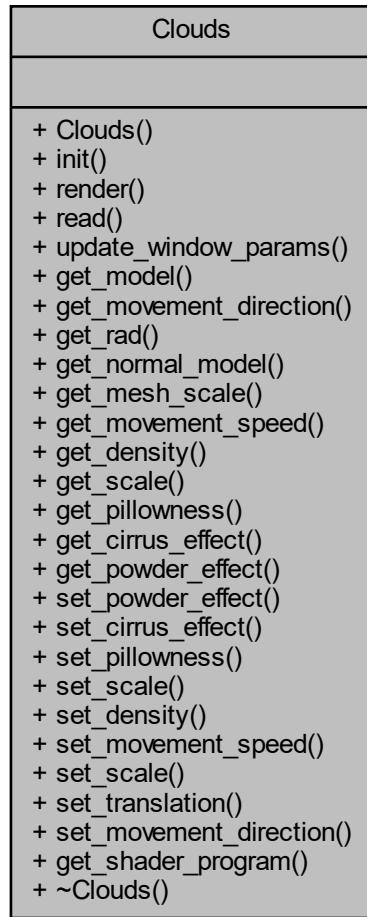
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ClampToEdgeMode.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ClampToEdgeMode.cpp](#)

6.5 Clouds Class Reference

```
#include <Clouds.h>
```

Collaboration diagram for Clouds:



Public Member Functions

- `Clouds ()`
- void `init (GLfloat window_width, GLfloat window_height, GLuint movement_speed)`
- void `render (glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat window_height)`
- void `read (GLuint index)`
- void `update_window_params (GLfloat window_width, GLfloat window_height)`
- `glm::mat4 get_model ()`
- `glm::vec3 get_movement_direction ()`
- `glm::vec3 get_rad ()`
- `glm::mat4 get_normal_model ()`

- `glm::vec3 get_mesh_scale ()`
- `GLfloat get_movement_speed ()`
- `GLfloat get_density ()`
- `GLfloat get_scale ()`
- `GLfloat get_pillowness ()`
- `GLfloat get_cirrus_effect ()`
- `bool get_powder_effect ()`
- `void set_powder_effect (bool cloud_powder_effect)`
- `void set_cirrus_effect (GLfloat cirrus_effect)`
- `void set_pillowness (GLfloat cloud_pillowness)`
- `void set_scale (GLfloat scale)`
- `void set_density (GLfloat density)`
- `void set_movement_speed (GLfloat speed)`
- `void set_scale (glm::vec3 scale)`
- `void set_translation (glm::vec3 translation)`
- `void set_movement_direction (glm::vec3 movement_dir)`
- `std::shared_ptr<CloudsShaderProgram> get_shader_program ()`
- `~Clouds ()`

6.5.1 Constructor & Destructor Documentation

6.5.1.1 Clouds()

```
Clouds::Clouds ( )
```

6.5.1.2 ~Clouds()

```
Clouds::~Clouds ( )
```

6.5.2 Member Function Documentation

6.5.2.1 get_cirrus_effect()

```
GLfloat Clouds::get_cirrus_effect ( )
```

6.5.2.2 get_density()

```
GLfloat Clouds::get_density ( )
```

6.5.2.3 get_mesh_scale()

```
glm::vec3 Clouds::get_mesh_scale ( )
```

6.5.2.4 get_model()

```
glm::mat4 Clouds::get_model ( )
```

6.5.2.5 get_movement_direction()

```
glm::vec3 Clouds::get_movement_direction ( )
```

6.5.2.6 get_movement_speed()

```
GLfloat Clouds::get_movement_speed ( )
```

6.5.2.7 get_normal_model()

```
glm::mat4 Clouds::get_normal_model ( )
```

6.5.2.8 get_pillowness()

```
GLfloat Clouds::get_pillowness ( )
```

6.5.2.9 get_powder_effect()

```
bool Clouds::get_powder_effect ( )
```

6.5.2.10 get_rad()

```
glm::vec3 Clouds::get_rad ( )
```

6.5.2.11 get_scale()

```
GLfloat Clouds::get_scale ( )
```

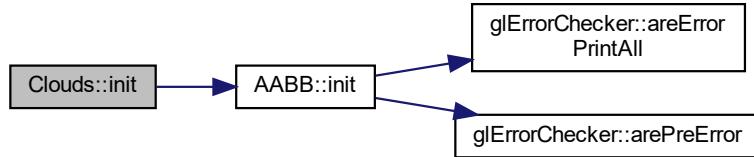
6.5.2.12 get_shader_program()

```
std::shared_ptr< CloudsShaderProgram > Clouds::get_shader_program ( )
```

6.5.2.13 init()

```
void Clouds::init (
    GLfloat window_width,
    GLfloat window_height,
    GLuint movement_speed )
```

Here is the call graph for this function:



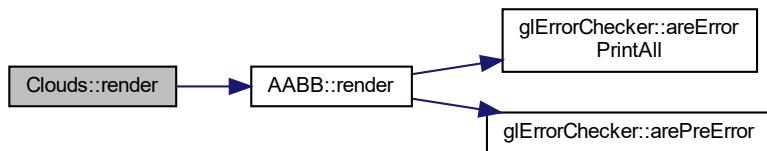
6.5.2.14 read()

```
void Clouds::read (
    GLuint index )
```

6.5.2.15 render()

```
void Clouds::render (
    glm::mat4 projection_matrix,
    glm::mat4 view_matrix,
    GLfloat window_width,
    GLfloat window_height )
```

Here is the call graph for this function:



6.5.2.16 set_cirrus_effect()

```
void Clouds::set_cirrus_effect (
    GLfloat cirrus_effect )
```

6.5.2.17 set_density()

```
void Clouds::set_density (
    GLfloat density )
```

6.5.2.18 set_movement_direction()

```
void Clouds::set_movement_direction (
    glm::vec3 movement_dir )
```

6.5.2.19 set_movement_speed()

```
void Clouds::set_movement_speed (
    GLfloat speed )
```

6.5.2.20 `set_pillowness()`

```
void Clouds::set_pillowness (
    GLfloat cloud_pillowness )
```

6.5.2.21 `set_powder_effect()`

```
void Clouds::set_powder_effect (
    bool cloud_powder_effect )
```

6.5.2.22 `set_scale()` [1/2]

```
void Clouds::set_scale (
    GLfloat scale )
```

6.5.2.23 `set_scale()` [2/2]

```
void Clouds::set_scale (
    glm::vec3 scale )
```

6.5.2.24 `set_translation()`

```
void Clouds::set_translation (
    glm::vec3 translation )
```

6.5.2.25 `update_window_params()`

```
void Clouds::update_window_params (
    GLfloat window_width,
    GLfloat window_height )
```

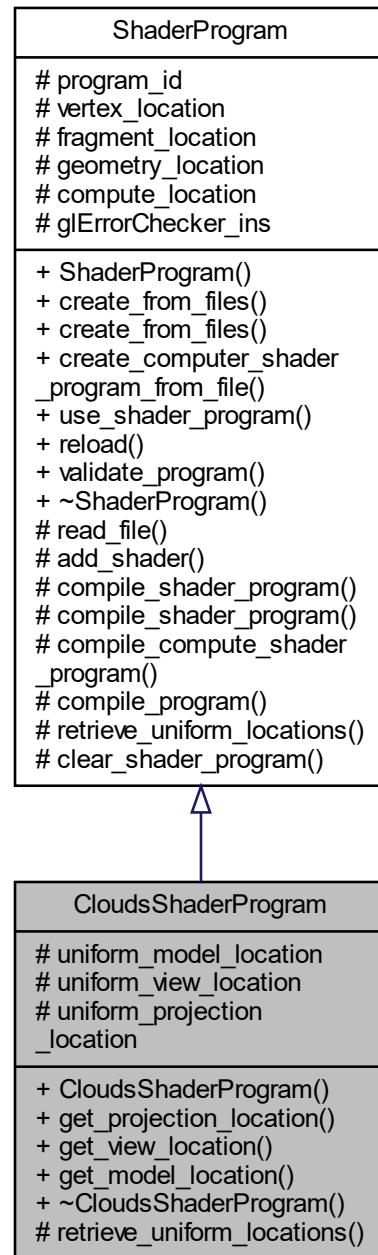
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Clouds.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Clouds.cpp](#)

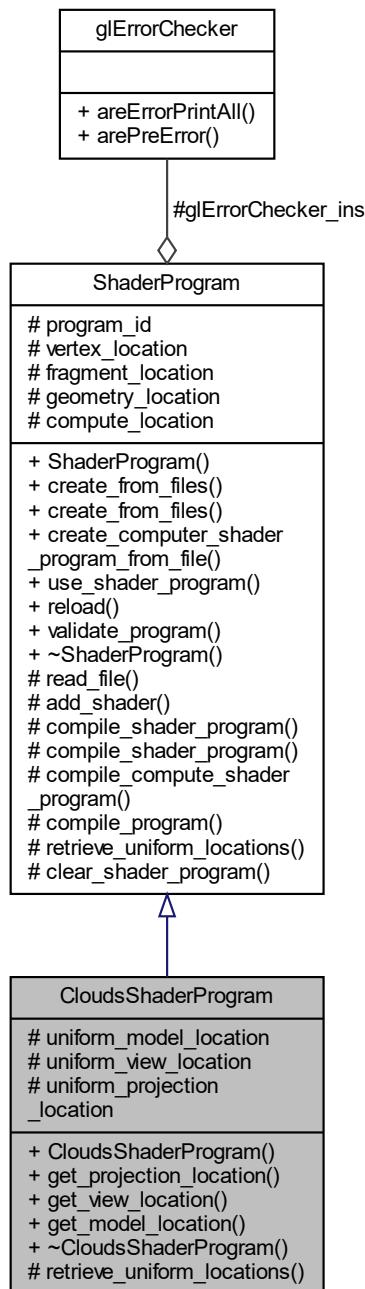
6.6 CloudsShaderProgram Class Reference

```
#include <CloudsShaderProgram.h>
```

Inheritance diagram for CloudsShaderProgram:



Collaboration diagram for CloudsShaderProgram:



Public Member Functions

- [CloudsShaderProgram \(\)](#)
- [GLuint get_projection_location \(\)](#)
- [GLuint get_view_location \(\)](#)
- [GLuint get_model_location \(\)](#)
- [~CloudsShaderProgram \(\)](#)

Protected Member Functions

- void `retrieve_uniform_locations()`

Protected Attributes

- GLuint `uniform_model_location`
- GLuint `uniform_view_location`
- GLuint `uniform_projection_location`

6.6.1 Constructor & Destructor Documentation

6.6.1.1 `CloudsShaderProgram()`

```
CloudsShaderProgram::CloudsShaderProgram( )
```

6.6.1.2 `~CloudsShaderProgram()`

```
CloudsShaderProgram::~CloudsShaderProgram( )
```

6.6.2 Member Function Documentation

6.6.2.1 `get_model_location()`

```
GLuint CloudsShaderProgram::get_model_location( )
```

6.6.2.2 `get_projection_location()`

```
GLuint CloudsShaderProgram::get_projection_location( )
```

6.6.2.3 `get_view_location()`

```
GLuint CloudsShaderProgram::get_view_location( )
```

6.6.2.4 `retrieve_uniform_locations()`

```
void CloudsShaderProgram::retrieve_uniform_locations ( ) [protected], [virtual]
```

Implements [ShaderProgram](#).

6.6.3 Member Data Documentation

6.6.3.1 `uniform_model_location`

```
GLuint CloudsShaderProgram::uniform_model_location [protected]
```

6.6.3.2 `uniform_projection_location`

```
GLuint CloudsShaderProgram::uniform_projection_location [protected]
```

6.6.3.3 `uniform_view_location`

```
GLuint CloudsShaderProgram::uniform_view_location [protected]
```

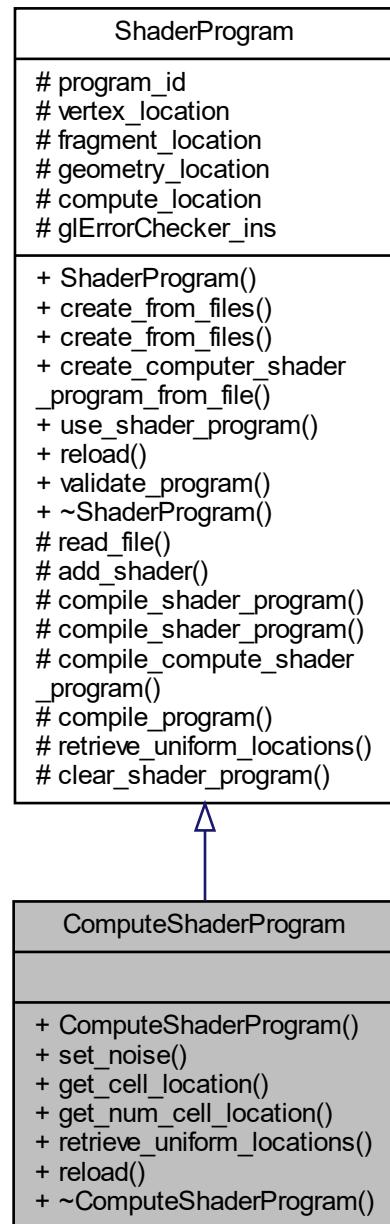
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[CloudsShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[CloudsShaderProgram.cpp](#)

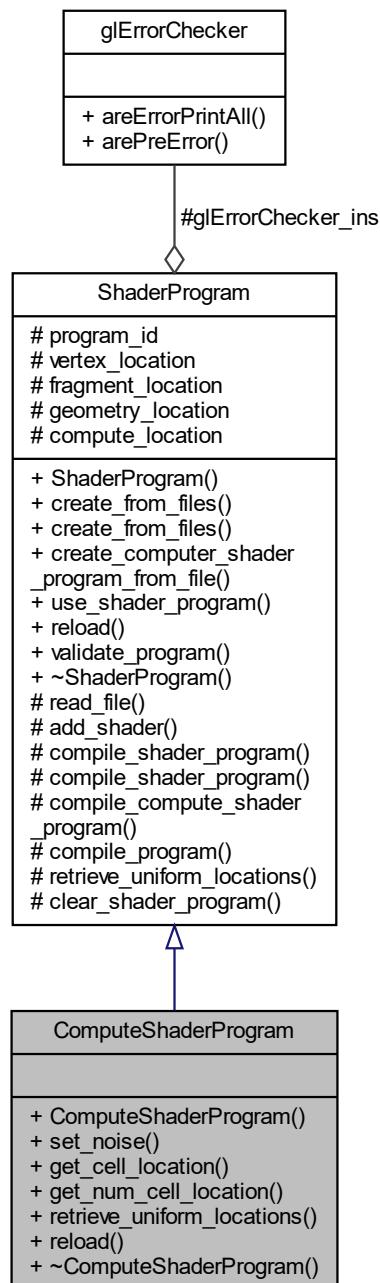
6.7 ComputeShaderProgram Class Reference

```
#include <ComputeShaderProgram.h>
```

Inheritance diagram for ComputeShaderProgram:



Collaboration diagram for ComputeShaderProgram:



Public Member Functions

- [ComputeShaderProgram \(\)](#)
- void [set_noise \(GLuint loc\)](#)
- GLuint [get_cell_location \(GLuint index\)](#)
- GLuint [get_num_cell_location \(GLuint index\)](#)
- void [retrieve_uniform_locations \(\)](#)

- void [reload \(\)](#)
- [~ComputeShaderProgram \(\)](#)

Additional Inherited Members

6.7.1 Constructor & Destructor Documentation

6.7.1.1 [ComputeShaderProgram\(\)](#)

```
ComputeShaderProgram::ComputeShaderProgram ( )
```

6.7.1.2 [~ComputeShaderProgram\(\)](#)

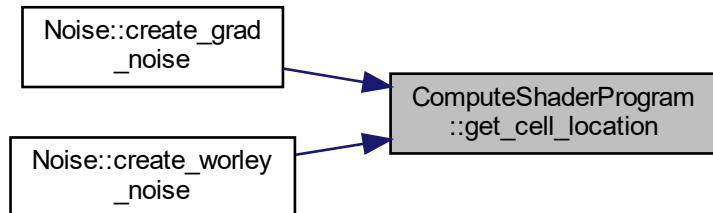
```
ComputeShaderProgram::~ComputeShaderProgram ( )
```

6.7.2 Member Function Documentation

6.7.2.1 [get_cell_location\(\)](#)

```
GLuint ComputeShaderProgram::get_cell_location (  
    GLuint index )
```

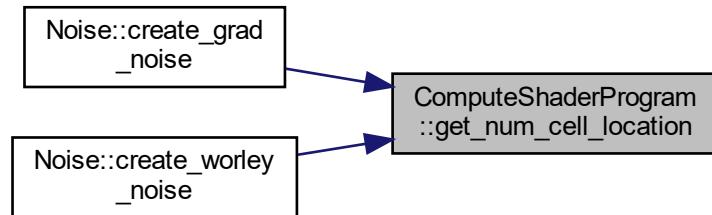
Here is the caller graph for this function:



6.7.2.2 get_num_cell_location()

```
GLuint ComputeShaderProgram::get_num_cell_location (
    GLuint index )
```

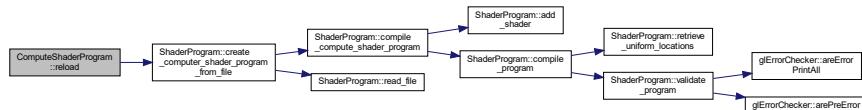
Here is the caller graph for this function:



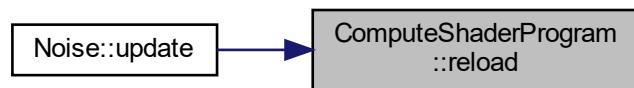
6.7.2.3 reload()

```
void ComputeShaderProgram::reload ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



6.7.2.4 retrieve_uniform_locations()

```
void ComputeShaderProgram::retrieve_uniform_locations ( ) [virtual]
```

Implements [ShaderProgram](#).

6.7.2.5 set_noise()

```
void ComputeShaderProgram::set_noise (  
    GLuint loc )
```

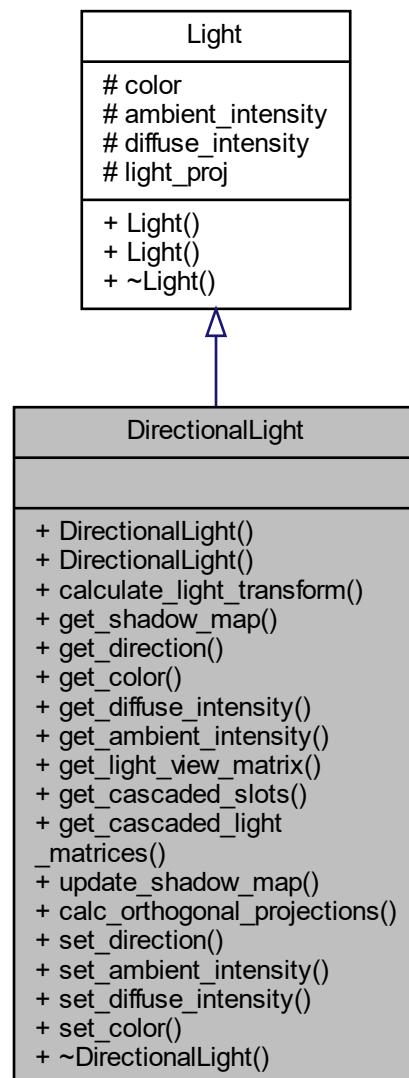
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ComputeShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ComputeShaderProgram.cpp](#)

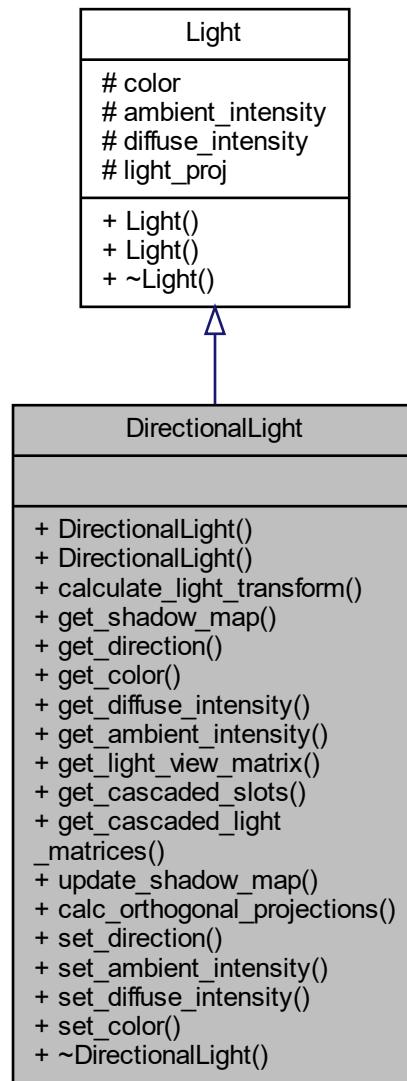
6.8 DirectionalLight Class Reference

```
#include <DirectionalLight.h>
```

Inheritance diagram for DirectionalLight:



Collaboration diagram for DirectionalLight:



Public Member Functions

- [DirectionalLight \(\)](#)
- [DirectionalLight \(GLuint shadow_width, GLuint shadow_height, GLfloat red, GLfloat green, GLfloat blue, GLfloat a_intensity, GLfloat d_intensity, GLfloat x_dir, GLfloat y_dir, GLfloat z_dir, GLfloat near_plane, GLfloat far_plane, GLfloat shadow_far_plane, int num_cascades\)](#)
- [glm::mat4 calculate_light_transform \(\)](#)
- [std::shared_ptr< CascadedShadowMap > get_shadow_map \(\)](#)
- [glm::vec3 get_direction \(\)](#)
- [glm::vec3 get_color \(\)](#)
- [float get_diffuse_intensity \(\)](#)
- [float get_ambient_intensity \(\)](#)

- `glm::mat4 get_light_view_matrix ()`
- `std::vector< GLfloat > get_cascaded_slots ()`
- `std::vector< glm::mat4 > & get_cascaded_light_matrices ()`
- `void update_shadow_map (GLfloat shadow_width, GLfloat shadow_height, GLuint num_cascades)`
- `void calc_orthogonal_projections (glm::mat4 camera_view_matrix, GLfloat window_width, GLfloat window_height, GLfloat fov, GLuint current_num_cascades)`
- `void set_direction (glm::vec3 direction)`
- `void set_ambient_intensity (float ambient_intensity)`
- `void set_diffuse_intensity (float diffuse_intensity)`
- `void set_color (glm::vec3 color)`
- `~DirectionalLight ()`

Additional Inherited Members

6.8.1 Constructor & Destructor Documentation

6.8.1.1 DirectionalLight() [1/2]

```
DirectionalLight::DirectionalLight ( )
```

6.8.1.2 DirectionalLight() [2/2]

```
DirectionalLight::DirectionalLight (
    GLuint shadow_width,
    GLuint shadow_height,
    GLfloat red,
    GLfloat green,
    GLfloat blue,
    GLfloat a_intensity,
    GLfloat d_intensity,
    GLfloat x_dir,
    GLfloat y_dir,
    GLfloat z_dir,
    GLfloat near_plane,
    GLfloat far_plane,
    GLfloat shadow_far_plane,
    int num_cascades )
```

6.8.1.3 ~DirectionalLight()

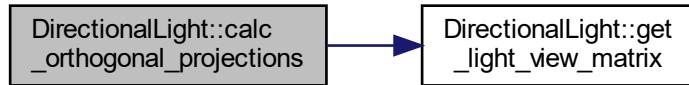
```
DirectionalLight::~DirectionalLight ( )
```

6.8.2 Member Function Documentation

6.8.2.1 calc_orthogonal_projections()

```
void DirectionalLight::calc_orthogonal_projections (
    glm::mat4 camera_view_matrix,
    GLfloat window_width,
    GLfloat window_height,
    GLfloat fov,
    GLuint current_num_cascades )
```

Here is the call graph for this function:



6.8.2.2 calculate_light_transform()

```
glm::mat4 DirectionalLight::calculate_light_transform ( )
```

6.8.2.3 get_ambient_intensity()

```
float DirectionalLight::get_ambient_intensity ( )
```

6.8.2.4 get_cascaded_light_matrices()

```
std::vector< glm::mat4 > & DirectionalLight::get_cascaded_light_matrices ( )
```

6.8.2.5 `get_cascaded_slots()`

```
std::vector< GLfloat > DirectionalLight::get_cascaded_slots ( )
```

6.8.2.6 `get_color()`

```
glm::vec3 DirectionalLight::get_color ( )
```

6.8.2.7 `get_diffuse_intensity()`

```
float DirectionalLight::get_diffuse_intensity ( )
```

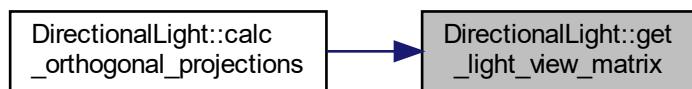
6.8.2.8 `get_direction()`

```
glm::vec3 DirectionalLight::get_direction ( )
```

6.8.2.9 `get_light_view_matrix()`

```
glm::mat4 DirectionalLight::get_light_view_matrix ( )
```

Here is the caller graph for this function:



6.8.2.10 `get_shadow_map()`

```
std::shared_ptr< CascadedShadowMap > DirectionalLight::get_shadow_map ( ) [inline]
```

6.8.2.11 set_ambient_intensity()

```
void DirectionalLight::set_ambient_intensity (
    float ambient_intensity )
```

6.8.2.12 set_color()

```
void DirectionalLight::set_color (
    glm::vec3 color )
```

6.8.2.13 set_diffuse_intensity()

```
void DirectionalLight::set_diffuse_intensity (
    float diffuse_intensity )
```

6.8.2.14 set_direction()

```
void DirectionalLight::set_direction (
    glm::vec3 direction )
```

6.8.2.15 update_shadow_map()

```
void DirectionalLight::update_shadow_map (
    GLfloat shadow_width,
    GLfloat shadow_height,
    GLuint num_cascades )
```

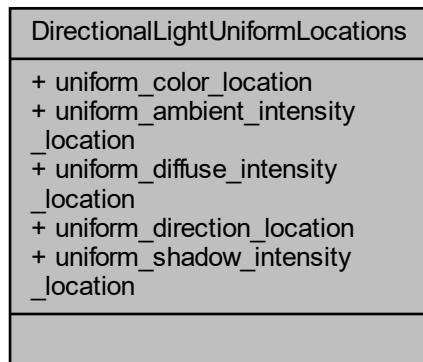
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.h
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.cpp

6.9 DirectionalLightUniformLocations Struct Reference

```
#include <DirectionalLightUniformLocations.h>
```

Collaboration diagram for DirectionalLightUniformLocations:



Public Attributes

- GLuint uniform_color_location
- GLuint uniform_ambient_intensity_location
- GLuint uniform_diffuse_intensity_location
- GLuint uniform_direction_location
- GLuint uniform_shadow_intensity_location

6.9.1 Member Data Documentation

6.9.1.1 uniform_ambient_intensity_location

```
GLuint DirectionalLightUniformLocations::uniform_ambient_intensity_location
```

6.9.1.2 uniform_color_location

```
GLuint DirectionalLightUniformLocations::uniform_color_location
```

6.9.1.3 uniform_diffuse_intensity_location

```
GLuint DirectionalLightUniformLocations::uniform_diffuse_intensity_location
```

6.9.1.4 uniform_direction_location

```
GLuint DirectionalLightUniformLocations::uniform_direction_location
```

6.9.1.5 uniform_shadow_intensity_location

```
GLuint DirectionalLightUniformLocations::uniform_shadow_intensity_location
```

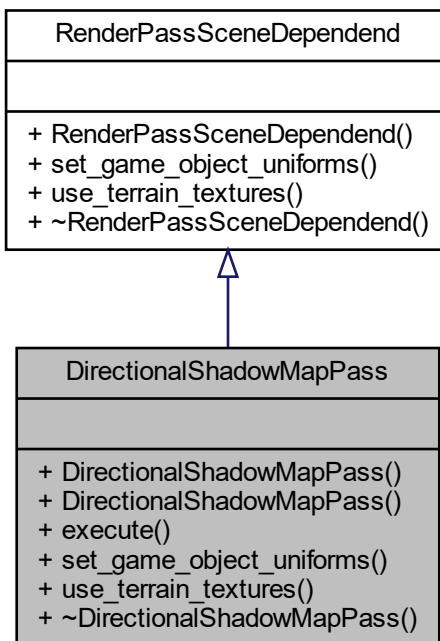
The documentation for this struct was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLightUniformLocations.h

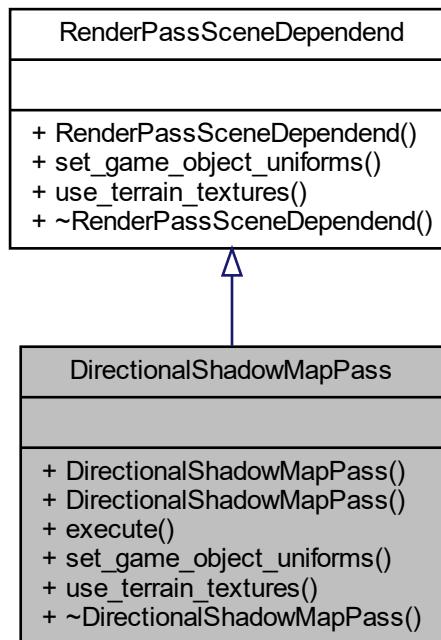
6.10 DirectionalShadowMapPass Class Reference

```
#include <DirectionalShadowMapPass.h>
```

Inheritance diagram for DirectionalShadowMapPass:



Collaboration diagram for DirectionalShadowMapPass:



Public Member Functions

- [DirectionalShadowMapPass \(\)](#)
- [DirectionalShadowMapPass \(std::shared_ptr< ShadowMapShaderProgram > shader_program\)](#)
- [void execute \(std::shared_ptr< DirectionalLight > d_light, glm::mat4 viewmatrix, bool first_person_mode, Scene *scene\)](#)
- [void set_game_object_uniforms \(glm::mat4 model, glm::mat4 normal_model, GLuint material_id\)](#)
- [bool use_terrain_textures \(\)](#)
- [~DirectionalShadowMapPass \(\)](#)

6.10.1 Constructor & Destructor Documentation

6.10.1.1 [DirectionalShadowMapPass\(\) \[1/2\]](#)

```
DirectionalShadowMapPass::DirectionalShadowMapPass ( )
```

6.10.1.2 `DirectionalShadowMapPass()` [2/2]

```
DirectionalShadowMapPass::DirectionalShadowMapPass ( std::shared_ptr< ShadowMapShaderProgram > shader_program )
```

6.10.1.3 `~DirectionalShadowMapPass()`

```
DirectionalShadowMapPass::~DirectionalShadowMapPass ( )
```

6.10.2 Member Function Documentation

6.10.2.1 `execute()`

```
void DirectionalShadowMapPass::execute ( std::shared_ptr< DirectionalLight > d_light, glm::mat4 viewmatrix, bool first_person_mode, Scene * scene )
```

6.10.2.2 `set_game_object_uniforms()`

```
void DirectionalShadowMapPass::set_game_object_uniforms ( glm::mat4 model, glm::mat4 normal_model, GLuint material_id ) [virtual]
```

Implements [RenderPassSceneDependend](#).

6.10.2.3 `use_terrain_textures()`

```
bool DirectionalShadowMapPass::use_terrain_textures ( ) [virtual]
```

Implements [RenderPassSceneDependend](#).

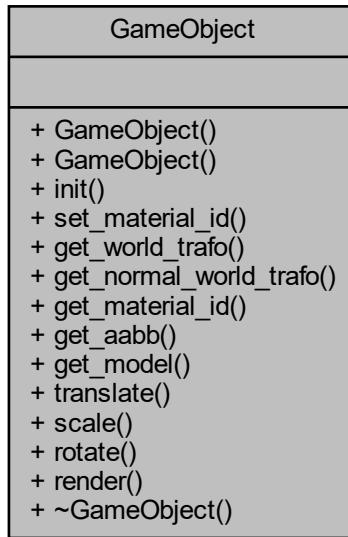
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[DirectionalShadowMapPass.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[DirectionalShadowMapPass.cpp](#)

6.11 GameObject Class Reference

```
#include <GameObject.h>
```

Collaboration diagram for GameObject:



Public Member Functions

- `GameObject ()`
- `GameObject (std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint material_id)`
- `void init (std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint material_id)`
- `void set_material_id (GLuint material_id)`
- `glm::mat4 get_world_trafo ()`
- `glm::mat4 get_normal_world_trafo ()`
- `GLuint get_material_id ()`
- `std::shared_ptr<AABB> get_aabb ()`
- `std::shared_ptr<Model> get_model ()`
- `void translate (glm::vec3 translate)`
- `void scale (GLfloat scale_factor)`
- `void rotate (Rotation rot)`
- `void render ()`
- `~GameObject ()`

6.11.1 Constructor & Destructor Documentation

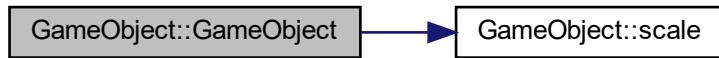
6.11.1.1 GameObject() [1/2]

```
GameObject::GameObject ( )
```

6.11.1.2 GameObject() [2/2]

```
GameObject::GameObject (
    std::string model_path,
    glm::vec3 translation,
    GLfloat scale,
    Rotation rot,
    GLuint material_id )
```

Here is the call graph for this function:



6.11.1.3 ~GameObject()

```
GameObject::~GameObject ( )
```

6.11.2 Member Function Documentation

6.11.2.1 get_aabb()

```
std::shared_ptr< AABB > GameObject::get_aabb ( )
```

6.11.2.2 get_material_id()

```
GLuint GameObject::get_material_id ( )
```

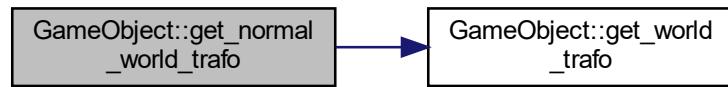
6.11.2.3 get_model()

```
std::shared_ptr< Model > GameObject::get_model ( )
```

6.11.2.4 get_normal_world_trafo()

```
glm::mat4 GameObject::get_normal_world_trafo ( )
```

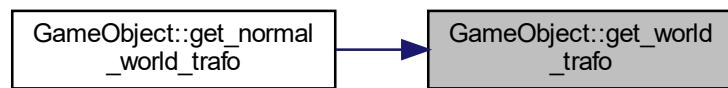
Here is the call graph for this function:



6.11.2.5 get_world_trafo()

```
glm::mat4 GameObject::get_world_trafo ( )
```

Here is the caller graph for this function:



6.11.2.6 init()

```
void GameObject::init (
    std::string model_path,
    glm::vec3 translation,
    GLfloat scale,
    Rotation rot,
    GLuint material_id )
```

Here is the call graph for this function:



6.11.2.7 render()

```
void GameObject::render ( )
```

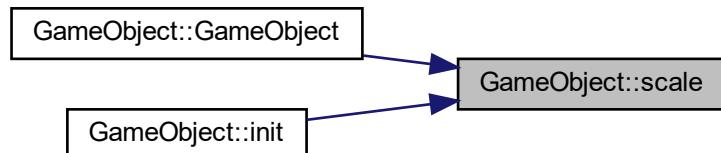
6.11.2.8 rotate()

```
void GameObject::rotate (
    Rotation rot )
```

6.11.2.9 scale()

```
void GameObject::scale (
    GLfloat scale_factor )
```

Here is the caller graph for this function:



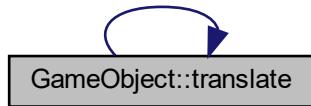
6.11.2.10 set_material_id()

```
void GameObject::set_material_id (
    GLuint material_id )
```

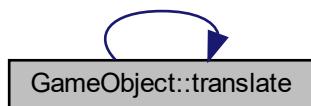
6.11.2.11 translate()

```
void GameObject::translate (
    glm::vec3 translate )
```

Here is the call graph for this function:



Here is the caller graph for this function:



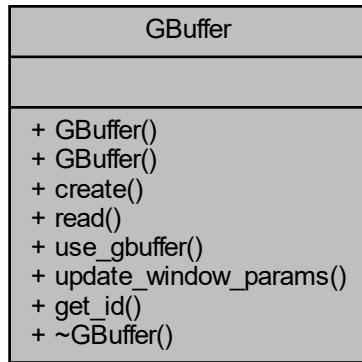
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GameObject.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GameObject.cpp](#)

6.12 GBuffer Class Reference

```
#include <GBuffer.h>
```

Collaboration diagram for GBuffer:



Public Member Functions

- [GBuffer \(\)](#)
- [GBuffer \(GLint window_width, GLint window_height\)](#)
- void [create \(\)](#)
- void [read \(GLint start_buffer_index\)](#)
- void [use_gbuffer \(GLuint g_buffer_lighting_uniform_position_location, GLuint g_buffer_lighting_uniform_normal_location, GLuint g_buffer_lighting_uniform_albedo_location, GLuint g_buffer_frag_depth_location, GLuint g_buffer_material_id_location\)](#)
- void [update_window_params \(GLfloat window_width, GLfloat window_height\)](#)
- GLint [get_id \(\)](#)
- [~GBuffer \(\)](#)

6.12.1 Constructor & Destructor Documentation

6.12.1.1 GBuffer() [1/2]

```
GBuffer::GBuffer( )
```

6.12.1.2 GBuffer() [2/2]

```
GBuffer::GBuffer (
    GLint window_width,
    GLint window_height )
```

6.12.1.3 ~GBuffer()

```
GBuffer::~GBuffer ( )
```

6.12.2 Member Function Documentation

6.12.2.1 create()

```
void GBuffer::create ( )
```

6.12.2.2 get_id()

```
GLint GBuffer::get_id ( )
```

6.12.2.3 read()

```
void GBuffer::read (
    GLint start_buffer_index )
```

6.12.2.4 update_window_params()

```
void GBuffer::update_window_params (
    GLfloat window_width,
    GLfloat window_height )
```

6.12.2.5 use_gbuffer()

```
void GBuffer::use_gbuffer (
    GLuint g_buffer_lighting_uniform_position_location,
    GLuint g_buffer_lighting_uniform_normal_location,
    GLuint g_buffer_lighting_uniform_albedo_location,
    GLuint g_buffer_frag_depth_location,
    GLuint g_buffer_material_id_location )
```

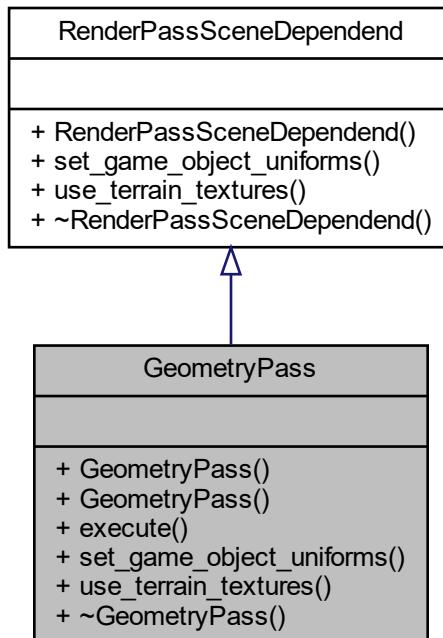
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GBuffer.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GBuffer.cpp](#)

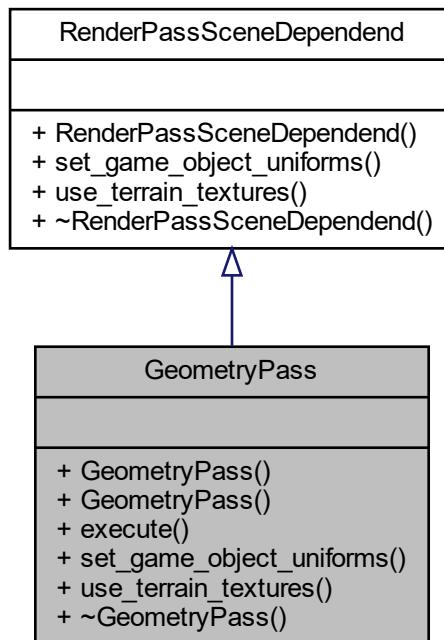
6.13 GeometryPass Class Reference

```
#include <GeometryPass.h>
```

Inheritance diagram for GeometryPass:



Collaboration diagram for GeometryPass:



Public Member Functions

- [GeometryPass \(\)](#)
- [GeometryPass \(std::shared_ptr< GeometryPassShaderProgram > shader_program\)](#)
- [void execute \(glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat window_height, GLuint gbuffer_id, bool first_person_mode, GLfloat delta_time, Scene *scene\)](#)
- [void set_game_object_uniforms \(glm::mat4 model, glm::mat4 normal_model, GLuint material_id\)](#)
- [bool use_terrain_textures \(\)](#)
- [~GeometryPass \(\)](#)

6.13.1 Constructor & Destructor Documentation

6.13.1.1 GeometryPass() [1/2]

```
GeometryPass::GeometryPass ( )
```

6.13.1.2 GeometryPass() [2/2]

```
GeometryPass::GeometryPass (
    std::shared_ptr< GeometryPassShaderProgram > shader_program )
```

6.13.1.3 ~GeometryPass()

```
GeometryPass::~GeometryPass ( )
```

6.13.2 Member Function Documentation

6.13.2.1 execute()

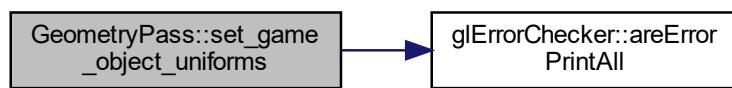
```
void GeometryPass::execute (
    glm::mat4 projection_matrix,
    glm::mat4 view_matrix,
    GLfloat window_width,
    GLfloat window_height,
    GLuint gbuffer_id,
    bool first_person_mode,
    GLfloat delta_time,
    Scene * scene )
```

6.13.2.2 set_game_object_uniforms()

```
void GeometryPass::set_game_object_uniforms (
    glm::mat4 model,
    glm::mat4 normal_model,
    GLuint material_id ) [virtual]
```

Implements [RenderPassSceneDependend](#).

Here is the call graph for this function:



6.13.2.3 `use_terrain_textures()`

```
bool GeometryPass::use_terrain_textures ( ) [virtual]
```

Implements [RenderPassSceneDependend](#).

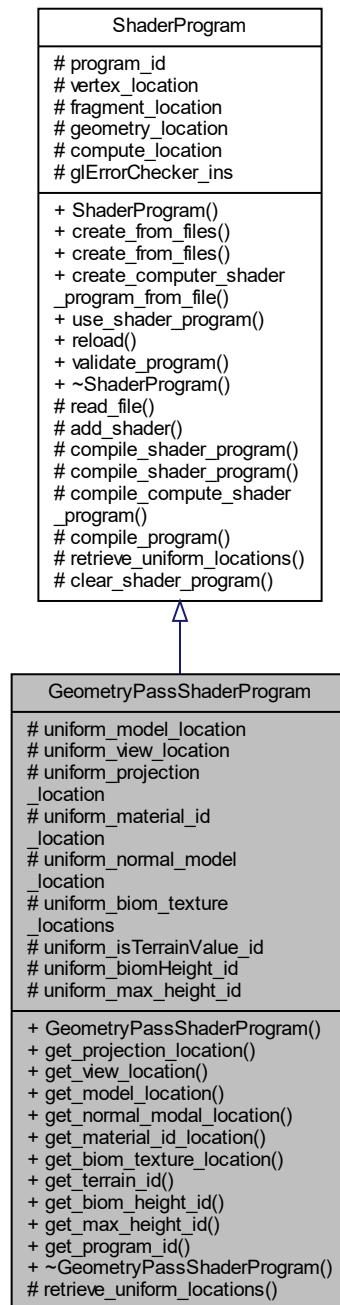
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GeometryPass.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GeometryPass.cpp](#)

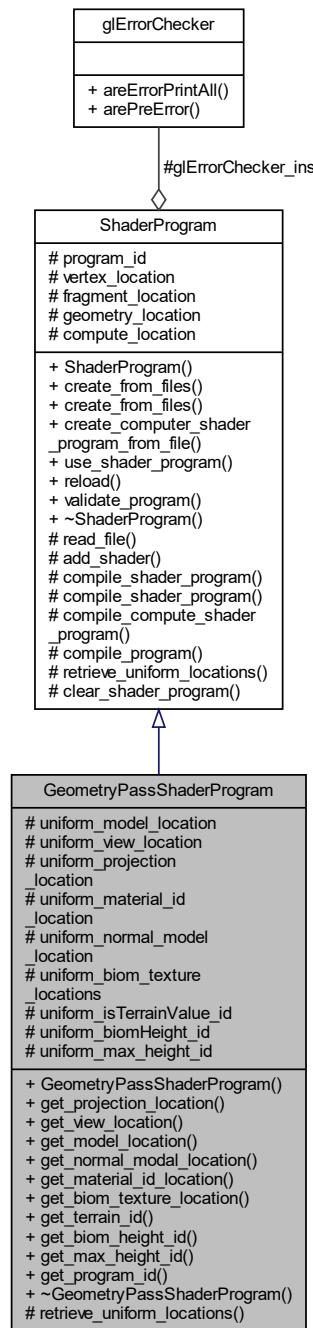
6.14 GeometryPassShaderProgram Class Reference

```
#include <GeometryPassShaderProgram.h>
```

Inheritance diagram for GeometryPassShaderProgram:



Collaboration diagram for GeometryPassShaderProgram:



Public Member Functions

- [GeometryPassShaderProgram \(\)](#)
- [GLuint get_projection_location \(\)](#)
- [GLuint get_view_location \(\)](#)
- [GLuint get_model_location \(\)](#)
- [GLuint get_normal_modal_location \(\)](#)

- GLuint `get_material_id_location ()`
- GLuint `get_biom_texture_location (GLuint id)`
- GLuint `get_terrain_id ()`
- GLuint `get_biom_height_id ()`
- GLuint `get_max_height_id ()`
- GLuint `get_program_id ()`
- `~GeometryPassShaderProgram ()`

Protected Member Functions

- void `retrieve_uniform_locations ()`

Protected Attributes

- GLuint `uniform_model_location`
- GLuint `uniform_view_location`
- GLuint `uniform_projection_location`
- GLuint `uniform_material_id_location`
- GLuint `uniform_normal_model_location`
- GLuint `uniform_biom_texture_locations [NUM BIOM_TEXTURES]`
- GLuint `uniform_isTerrainValue_id`
- GLuint `uniform_biomHeight_id`
- GLuint `uniform_max_height_id`

6.14.1 Constructor & Destructor Documentation

6.14.1.1 GeometryPassShaderProgram()

```
GeometryPassShaderProgram::GeometryPassShaderProgram ( )
```

6.14.1.2 ~GeometryPassShaderProgram()

```
GeometryPassShaderProgram::~GeometryPassShaderProgram ( )
```

6.14.2 Member Function Documentation

6.14.2.1 get_biom_height_id()

```
GLuint GeometryPassShaderProgram::get_biom_height_id ( )
```

6.14.2.2 `get_biom_texture_location()`

```
GLuint GeometryPassShaderProgram::get_biom_texture_location (
    GLuint id )
```

6.14.2.3 `get_material_id_location()`

```
GLuint GeometryPassShaderProgram::get_material_id_location ( )
```

6.14.2.4 `get_max_height_id()`

```
GLuint GeometryPassShaderProgram::get_max_height_id ( )
```

6.14.2.5 `get_model_location()`

```
GLuint GeometryPassShaderProgram::get_model_location ( )
```

6.14.2.6 `get_normal_modal_location()`

```
GLuint GeometryPassShaderProgram::get_normal_modal_location ( )
```

6.14.2.7 `get_program_id()`

```
GLuint GeometryPassShaderProgram::get_program_id ( ) [inline]
```

6.14.2.8 `get_projection_location()`

```
GLuint GeometryPassShaderProgram::get_projection_location ( )
```

6.14.2.9 get_terrain_id()

```
GLuint GeometryPassShaderProgram::get_terrain_id ( )
```

6.14.2.10 get_view_location()

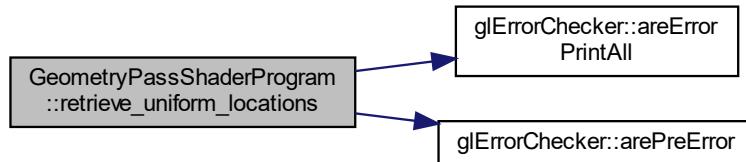
```
GLuint GeometryPassShaderProgram::get_view_location ( )
```

6.14.2.11 retrieve_uniform_locations()

```
void GeometryPassShaderProgram::retrieve_uniform_locations ( ) [protected], [virtual]
```

Implements [ShaderProgram](#).

Here is the call graph for this function:



6.14.3 Member Data Documentation

6.14.3.1 uniform_biom_texture_locations

```
GLuint GeometryPassShaderProgram::uniform_biom_texture_locations[NUM\_BIOM\_TEXTURES] [protected]
```

6.14.3.2 uniform_biomHeight_id

```
GLuint GeometryPassShaderProgram::uniform_biomHeight_id [protected]
```

6.14.3.3 uniform_isTerrainValue_id

```
GLuint GeometryPassShaderProgram::uniform_isTerrainValue_id [protected]
```

6.14.3.4 uniform_material_id_location

```
GLuint GeometryPassShaderProgram::uniform_material_id_location [protected]
```

6.14.3.5 uniform_max_height_id

```
GLuint GeometryPassShaderProgram::uniform_max_height_id [protected]
```

6.14.3.6 uniform_model_location

```
GLuint GeometryPassShaderProgram::uniform_model_location [protected]
```

6.14.3.7 uniform_normal_model_location

```
GLuint GeometryPassShaderProgram::uniform_normal_model_location [protected]
```

6.14.3.8 uniform_projection_location

```
GLuint GeometryPassShaderProgram::uniform_projection_location [protected]
```

6.14.3.9 uniform_view_location

```
GLuint GeometryPassShaderProgram::uniform_view_location [protected]
```

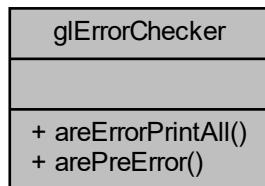
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GeometryPassShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GeometryPassShaderProgram.cpp](#)

6.15 glErrorChecker Class Reference

```
#include <glErrorChecker.h>
```

Collaboration diagram for glErrorChecker:



Public Member Functions

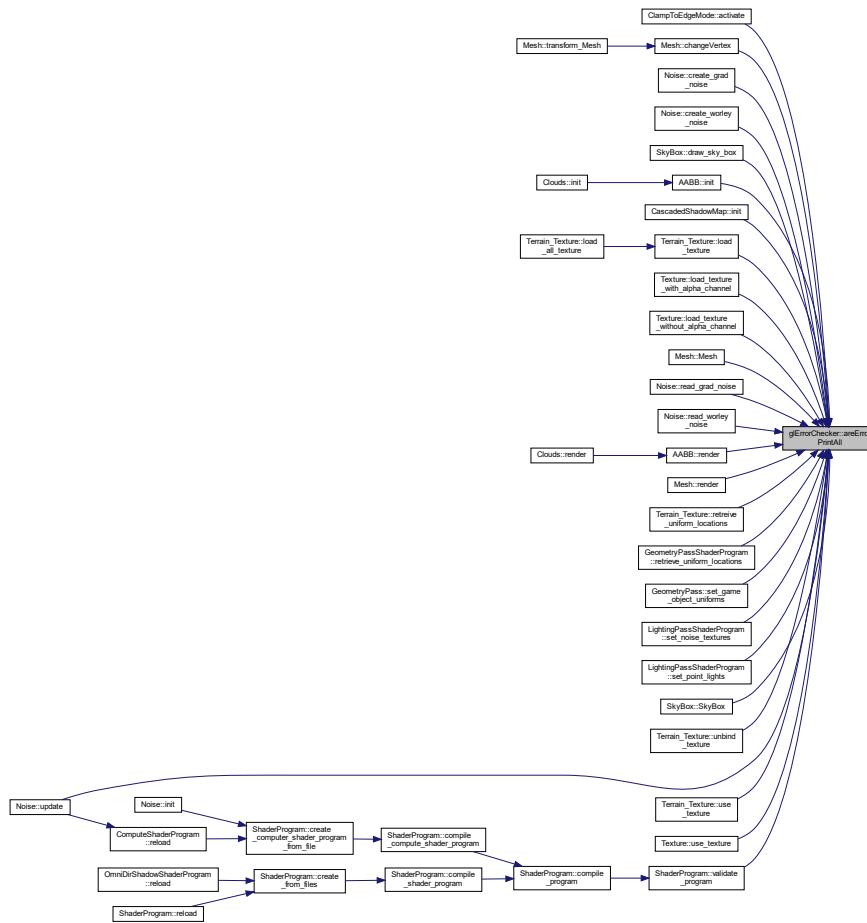
- bool [areErrorPrintAll](#) (std::string AdditionalArrayMessage="Empty")
- bool [arePreError](#) (std::string AdditionalArrayMessage="Empty")

6.15.1 Member Function Documentation

6.15.1.1 [areErrorPrintAll\(\)](#)

```
bool glErrorChecker::areErrorPrintAll (
    std::string AdditionalArrayMessage = "Empty" ) [inline]
```

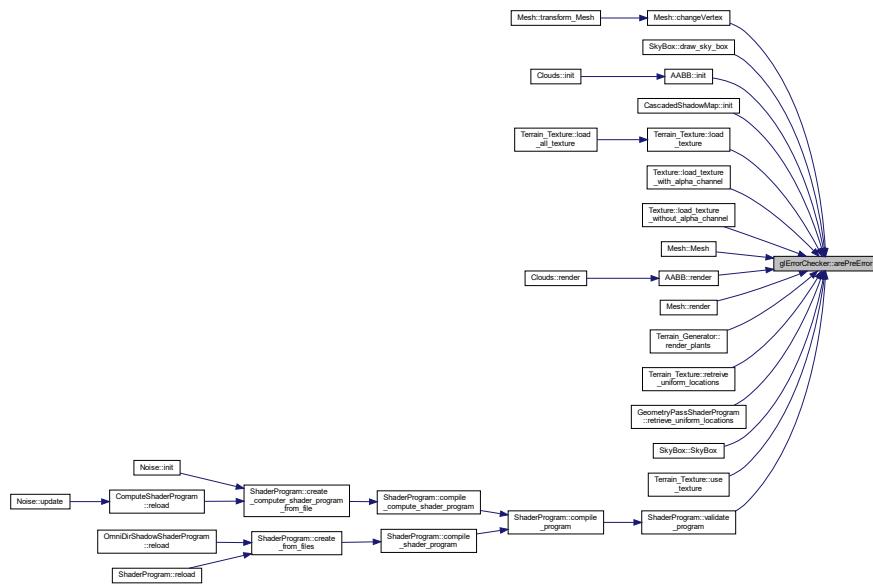
Here is the caller graph for this function:



6.15.1.2 arePreError()

```
bool gLErrorChecker::arePreError (
    std::string AdditionalArrayMessage = "Empty" ) [inline]
```

Here is the caller graph for this function:



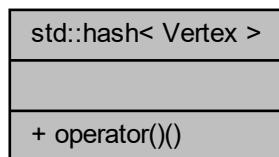
The documentation for this class was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[glErrorChecker.h](#)

6.16 std::hash< Vertex > Struct Reference

```
#include <Vertex.h>
```

Collaboration diagram for `std::hash< Vertex >`:



Public Member Functions

- `size_t operator() (Vertex const &vert) const`

6.16.1 Member Function Documentation

6.16.1.1 operator()()

```
size_t std::hash< Vertex >::operator() (
    Vertex const & vert ) const [inline]
```

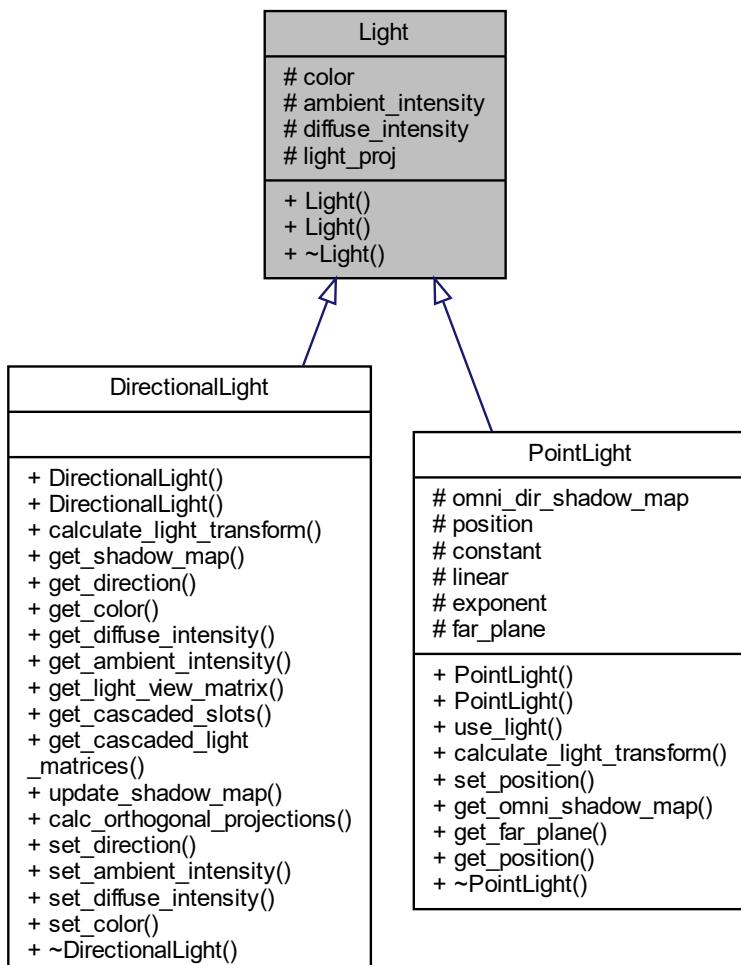
The documentation for this struct was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Vertex.h

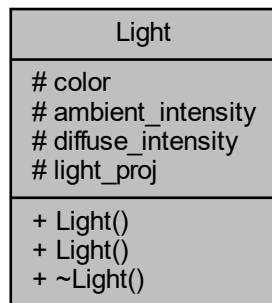
6.17 Light Class Reference

```
#include <Light.h>
```

Inheritance diagram for Light:



Collaboration diagram for Light:



Public Member Functions

- [Light \(\)](#)
- [Light \(GLfloat shadow_width, GLfloat shadow_height, GLfloat red, GLfloat green, GLfloat blue, GLfloat a_intensity, GLfloat d_intensity\)](#)
- [~Light \(\)](#)

Protected Attributes

- `glm::vec3 color`
- `float ambient_intensity`
- `float diffuse_intensity`
- `glm::mat4 light_proj`

6.17.1 Constructor & Destructor Documentation

6.17.1.1 Light() [1/2]

```
Light::Light ( )
```

6.17.1.2 Light() [2/2]

```
Light::Light (
    GLfloat shadow_width,
    GLfloat shadow_height,
    GLfloat red,
    GLfloat green,
    GLfloat blue,
    GLfloat a_intensity,
    GLfloat d_intensity )
```

6.17.1.3 ~Light()

```
Light::~Light ( )
```

6.17.2 Member Data Documentation

6.17.2.1 ambient_intensity

```
float Light::ambient_intensity [protected]
```

6.17.2.2 color

```
glm::vec3 Light::color [protected]
```

6.17.2.3 diffuse_intensity

```
float Light::diffuse_intensity [protected]
```

6.17.2.4 light_proj

```
glm::mat4 Light::light_proj [protected]
```

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

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Light.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Light.cpp](#)

6.18 LightingPass Class Reference

```
#include <LightingPass.h>
```

Collaboration diagram for LightingPass:

LightingPass
+ LightingPass() + init() + execute() + ~LightingPass()

Public Member Functions

- [LightingPass \(\)](#)
- void [init \(std::shared_ptr<LightingPassShaderProgram> shader_program\)](#)
- void [execute \(glm::mat4 projection_matrix, glm::mat4 view_matrix, std::shared_ptr<GBuffer> gbuffer, std::shared_ptr<DirectionalLight> main_light, std::vector<std::shared_ptr<PointLight>> &point_lights, GLuint point_light_count, glm::vec3 camera_position, GLuint material_counter, std::vector<std::shared_ptr<Material>> &materials, std::shared_ptr<Noise> noise, std::shared_ptr<Clouds> cloud, float delta_time\)](#)
- [~LightingPass \(\)](#)

6.18.1 Constructor & Destructor Documentation

6.18.1.1 LightingPass()

```
LightingPass::LightingPass ( )
```

6.18.1.2 ~LightingPass()

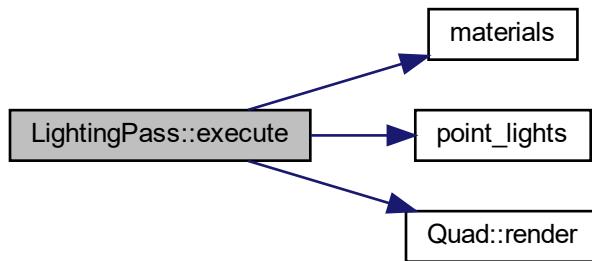
```
LightingPass::~LightingPass ( )
```

6.18.2 Member Function Documentation

6.18.2.1 execute()

```
void LightingPass::execute (
    glm::mat4 projection_matrix,
    glm::mat4 view_matrix,
    std::shared_ptr<GBuffer> gbuffer,
    std::shared_ptr<DirectionalLight> main_light,
    std::vector<std::shared_ptr<PointLight>> &point_lights,
    GLuint point_light_count,
    glm::vec3 camera_position,
    GLuint material_counter,
    std::vector<std::shared_ptr<Material>> &materials,
    std::shared_ptr<Noise> noise,
    std::shared_ptr<Clouds> cloud,
    float delta_time )
```

Here is the call graph for this function:



6.18.2.2 init()

```
void LightingPass::init (
    std::shared_ptr< LightingPassShaderProgram > shader_program )
```

Here is the call graph for this function:



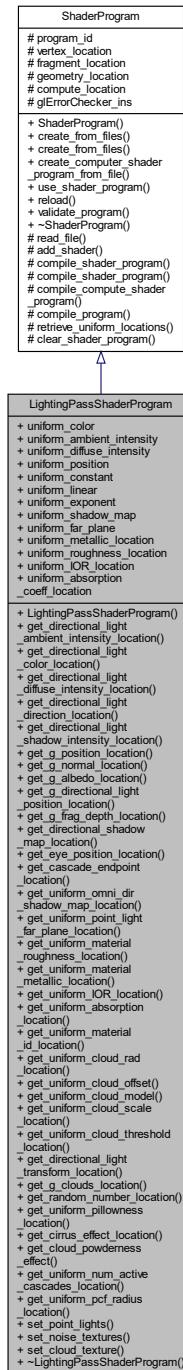
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LightingPass.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LightingPass.cpp](#)

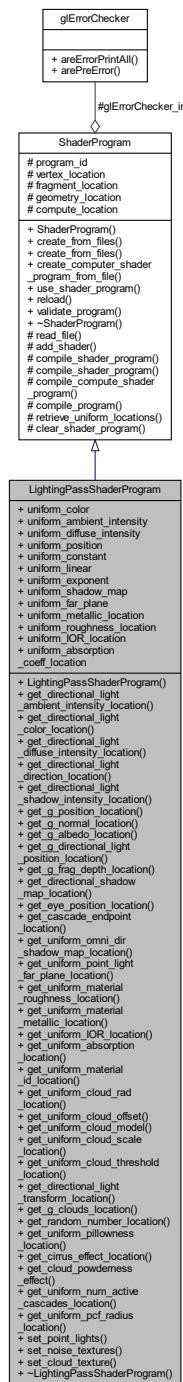
6.19 LightingPassShaderProgram Class Reference

```
#include <LightingPassShaderProgram.h>
```

Inheritance diagram for LightingPassShaderProgram:



Collaboration diagram for LightingPassShaderProgram:



Public Member Functions

- [LightingPassShaderProgram \(\)](#)
- [GLuint get_directional_light_ambient_intensity_location \(\)](#)
- [GLuint get_directional_light_color_location \(\)](#)
- [GLuint get_directional_light_diffuse_intensity_location \(\)](#)
- [GLuint get_directional_light_direction_location \(\)](#)

- GLuint `get_directional_light_shadow_intensity_location ()`
- GLuint `get_g_position_location ()`
- GLuint `get_g_normal_location ()`
- GLuint `get_g_albedo_location ()`
- GLuint `get_g_directional_light_position_location (GLuint index)`
- GLuint `get_g_frag_depth_location ()`
- GLuint `get_directional_shadow_map_location (GLuint index)`
- GLuint `get_eye_position_location ()`
- GLuint `get_cascade_endpoint_location (GLuint index)`
- GLuint `get_uniform_omni_dir_shadow_map_location (GLuint index)`
- GLuint `get_uniform_point_light_far_plane_location (GLuint index)`
- GLuint `get_uniform_material_roughness_location (GLuint index)`
- GLuint `get_uniform_material_metallic_location (GLuint index)`
- GLuint `get_uniform_IOR_location (GLuint index)`
- GLuint `get_uniform_absorption_location (GLuint index)`
- GLuint `get_uniform_material_id_location ()`
- GLuint `get_uniform_cloud_rad_location ()`
- GLuint `get_uniform_cloud_offset ()`
- GLuint `get_uniform_cloud_model ()`
- GLuint `get_uniform_cloud_scale_location ()`
- GLuint `get_uniform_cloud_threshold_location ()`
- GLuint `get_directional_light_transform_location (GLuint index)`
- GLuint `get_g_clouds_location ()`
- GLuint `get_random_number_location ()`
- GLuint `get_uniform_pillowness_location ()`
- GLuint `get_cirrus_effect_location ()`
- GLuint `get_cloud_powderness_effect ()`
- GLuint `get_uniform_num_active_cascades_location ()`
- GLuint `get_uniform_pcf_radius_location ()`
- void `set_point_lights (std::vector< std::shared_ptr< PointLight > > &p_light, unsigned int light_count, unsigned int texture_unit, unsigned int offset)`
- void `set_noise_textures (GLuint start)`
- void `set_cloud_texture (GLuint index)`
- `~LightingPassShaderProgram ()`

Additional Inherited Members

6.19.1 Constructor & Destructor Documentation

6.19.1.1 LightingPassShaderProgram()

```
LightingPassShaderProgram::LightingPassShaderProgram ( )
```

6.19.1.2 ~LightingPassShaderProgram()

```
LightingPassShaderProgram::~LightingPassShaderProgram ( )
```

6.19.2 Member Function Documentation

6.19.2.1 get_cascade_endpoint_location()

```
GLuint LightingPassShaderProgram::get_cascade_endpoint_location (
    GLuint index )
```

6.19.2.2 get_cirrus_effect_location()

```
GLuint LightingPassShaderProgram::get_cirrus_effect_location ( )
```

6.19.2.3 get_cloud_powderness_effect()

```
GLuint LightingPassShaderProgram::get_cloud_powderness_effect ( )
```

6.19.2.4 get_directional_light_ambient_intensity_location()

```
GLuint LightingPassShaderProgram::get_directional_light_ambient_intensity_location ( )
```

6.19.2.5 get_directional_light_color_location()

```
GLuint LightingPassShaderProgram::get_directional_light_color_location ( )
```

6.19.2.6 get_directional_light_diffuse_intensity_location()

```
GLuint LightingPassShaderProgram::get_directional_light_diffuse_intensity_location ( )
```

6.19.2.7 get_directional_light_direction_location()

```
GLuint LightingPassShaderProgram::get_directional_light_direction_location ( )
```

6.19.2.8 get_directional_light_shadow_intensity_location()

```
GLuint LightingPassShaderProgram::get_directional_light_shadow_intensity_location ( )
```

6.19.2.9 get_directional_light_transform_location()

```
GLuint LightingPassShaderProgram::get_directional_light_transform_location (  
    GLuint index )
```

6.19.2.10 get_directional_shadow_map_location()

```
GLuint LightingPassShaderProgram::get_directional_shadow_map_location (   
    GLuint index )
```

6.19.2.11 get_eye_position_location()

```
GLuint LightingPassShaderProgram::get_eye_position_location ( )
```

6.19.2.12 get_g_albedo_location()

```
GLuint LightingPassShaderProgram::get_g_albedo_location ( )
```

6.19.2.13 get_g_clouds_location()

```
GLuint LightingPassShaderProgram::get_g_clouds_location ( )
```

6.19.2.14 get_g_directional_light_position_location()

```
GLuint LightingPassShaderProgram::get_g_directional_light_position_location (   
    GLuint index )
```

6.19.2.15 get_g_frag_depth_location()

```
GLuint LightingPassShaderProgram::get_g_frag_depth_location ( )
```

6.19.2.16 get_g_normal_location()

```
GLuint LightingPassShaderProgram::get_g_normal_location ( )
```

6.19.2.17 get_g_position_location()

```
GLuint LightingPassShaderProgram::get_g_position_location ( )
```

6.19.2.18 get_random_number_location()

```
GLuint LightingPassShaderProgram::get_random_number_location ( )
```

6.19.2.19 get_uniform_absorption_location()

```
GLuint LightingPassShaderProgram::get_uniform_absorption_location (   
    GLuint index )
```

6.19.2.20 get_uniform_cloud_model()

```
GLuint LightingPassShaderProgram::get_uniform_cloud_model ( )
```

6.19.2.21 get_uniform_cloud_offset()

```
GLuint LightingPassShaderProgram::get_uniform_cloud_offset ( )
```

6.19.2.22 get_uniform_cloud_rad_location()

```
GLuint LightingPassShaderProgram::get_uniform_cloud_rad_location ( )
```

6.19.2.23 get_uniform_cloud_scale_location()

```
GLuint LightingPassShaderProgram::get_uniform_cloud_scale_location ( )
```

6.19.2.24 get_uniform_cloud_threshold_location()

```
GLuint LightingPassShaderProgram::get_uniform_cloud_threshold_location ( )
```

6.19.2.25 get_uniform_IOR_location()

```
GLuint LightingPassShaderProgram::get_uniform_IOR_location (  
    GLuint index )
```

6.19.2.26 get_uniform_material_id_location()

```
GLuint LightingPassShaderProgram::get_uniform_material_id_location ( )
```

6.19.2.27 get_uniform_material_metallic_location()

```
GLuint LightingPassShaderProgram::get_uniform_material_metallic_location (   
    GLuint index )
```

6.19.2.28 get_uniform_material_roughness_location()

```
GLuint LightingPassShaderProgram::get_uniform_material_roughness_location (   
    GLuint index )
```

6.19.2.29 get_uniform_num_active_cascades_location()

```
GLuint LightingPassShaderProgram::get_uniform_num_active_cascades_location ( )
```

6.19.2.30 get_uniform_omni_dir_shadow_map_location()

```
GLuint LightingPassShaderProgram::get_uniform_omni_dir_shadow_map_location (
    GLuint index )
```

6.19.2.31 get_uniform_pcf_radius_location()

```
GLuint LightingPassShaderProgram::get_uniform_pcf_radius_location ( )
```

6.19.2.32 get_uniform_pillowness_location()

```
GLuint LightingPassShaderProgram::get_uniform_pillowness_location ( )
```

6.19.2.33 get_uniform_point_light_far_plane_location()

```
GLuint LightingPassShaderProgram::get_uniform_point_light_far_plane_location (
    GLuint index )
```

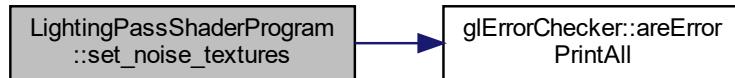
6.19.2.34 set_cloud_texture()

```
void LightingPassShaderProgram::set_cloud_texture (
    GLuint index )
```

6.19.2.35 set_noise_textures()

```
void LightingPassShaderProgram::set_noise_textures (
    GLuint start )
```

Here is the call graph for this function:



6.19.2.36 set_point_lights()

```
void LightingPassShaderProgram::set_point_lights (
    std::vector< std::shared_ptr< PointLight > > & p_light,
    unsigned int light_count,
    unsigned int texture_unit,
    unsigned int offset )
```

Here is the call graph for this function:



6.19.3 Member Data Documentation

6.19.3.1 uniform_absorption_coeff_location

```
GLuint LightingPassShaderProgram::uniform_absorption_coeff_location
```

6.19.3.2 uniform_ambient_intensity

```
GLuint LightingPassShaderProgram::uniform_ambient_intensity
```

6.19.3.3 uniform_color

```
GLuint LightingPassShaderProgram::uniform_color
```

6.19.3.4 uniform_constant

```
GLuint LightingPassShaderProgram::uniform_constant
```

6.19.3.5 uniform_diffuse_intensity

```
GLuint LightingPassShaderProgram::uniform_diffuse_intensity
```

6.19.3.6 uniform_exponent

```
GLuint LightingPassShaderProgram::uniform_exponent
```

6.19.3.7 uniform_far_plane

```
GLuint LightingPassShaderProgram::uniform_far_plane
```

6.19.3.8 uniform_IOR_location

```
GLuint LightingPassShaderProgram::uniform_IOR_location
```

6.19.3.9 uniform_linear

```
GLuint LightingPassShaderProgram::uniform_linear
```

6.19.3.10 uniform_metallic_location

```
GLuint LightingPassShaderProgram::uniform_metallic_location
```

6.19.3.11 uniform_position

```
GLuint LightingPassShaderProgram::uniform_position
```

6.19.3.12 uniform_roughness_location

```
GLuint LightingPassShaderProgram::uniform_roughness_location
```

6.19.3.13 uniform_shadow_map

```
GLuint LightingPassShaderProgram::uniform_shadow_map
```

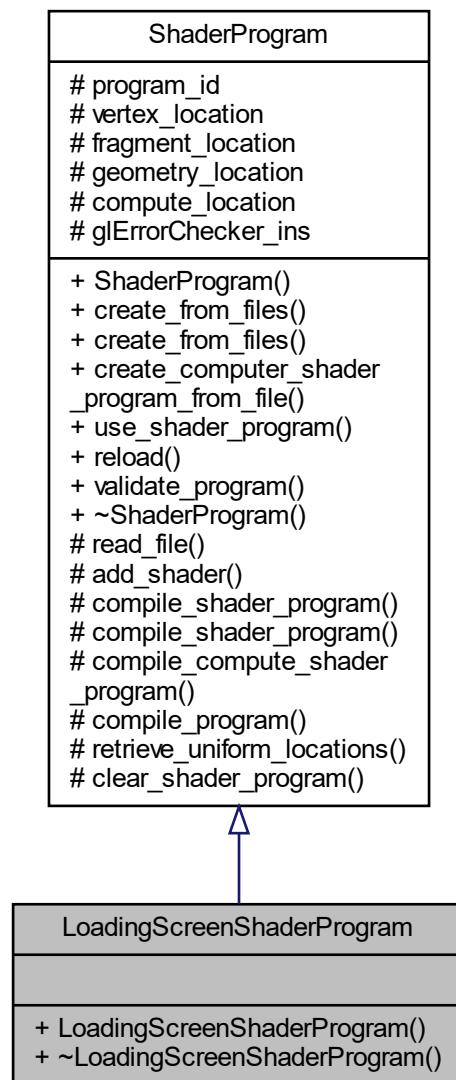
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LightingPassShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LightingPassShaderProgram.cpp](#)

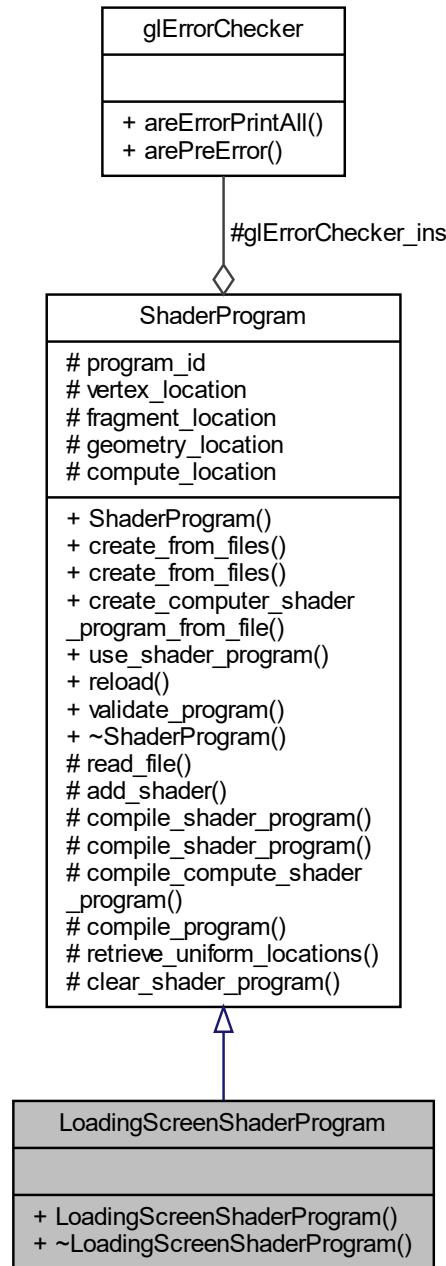
6.20 LoadingScreenShaderProgram Class Reference

```
#include <LoadingScreenShaderProgram.h>
```

Inheritance diagram for LoadingScreenShaderProgram:



Collaboration diagram for LoadingScreenShaderProgram:



Public Member Functions

- [LoadingScreenShaderProgram \(\)](#)
- [~LoadingScreenShaderProgram \(\)](#)

Additional Inherited Members

6.20.1 Constructor & Destructor Documentation

6.20.1.1 LoadingScreenShaderProgram()

```
LoadingScreenShaderProgram::LoadingScreenShaderProgram ()
```

6.20.1.2 ~LoadingScreenShaderProgram()

```
LoadingScreenShaderProgram::~LoadingScreenShaderProgram ()
```

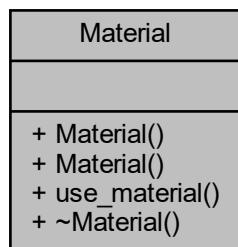
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LoadingScreenShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[LoadingScreenShaderProgram.cpp](#)

6.21 Material Class Reference

```
#include <Material.h>
```

Collaboration diagram for Material:



Public Member Functions

- [Material \(\)](#)
- [Material \(GLfloat metallic, GLfloat roughness, GLfloat IOR, GLfloat absorption_coeff\)](#)
- void [use_material \(GLuint uniform_metallic_location, GLuint uniform_roughness_location, GLuint uniform_IOR_location, GLfloat uniform_absorption_coeff_location\)](#)
- [~Material \(\)](#)

6.21.1 Constructor & Destructor Documentation

6.21.1.1 Material() [1/2]

```
Material::Material ( )
```

6.21.1.2 Material() [2/2]

```
Material::Material (
    GLfloat metallic,
    GLfloat roughness,
    GLfloat IOR,
    GLfloat absorption_coef )
```

6.21.1.3 ~Material()

```
Material::~Material ( )
```

6.21.2 Member Function Documentation

6.21.2.1 use_material()

```
void Material::use_material (
    GLuint uniform_metallic_location,
    GLuint uniform_roughness_location,
    GLuint uniform_IOR_location,
    GLfloat uniform_absorption_coef_location )
```

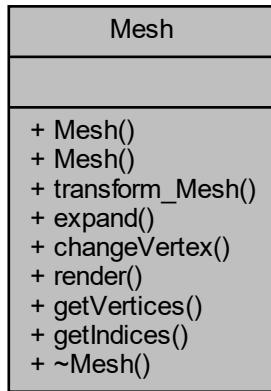
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Material.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Material.cpp](#)

6.22 Mesh Class Reference

```
#include <Mesh.h>
```

Collaboration diagram for Mesh:



Public Member Functions

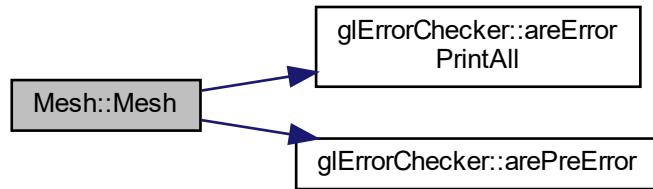
- `Mesh (std::vector< Vertex > vertices, std::vector< unsigned int > indices)`
- `Mesh ()`
- `glm::mat4 transform_Mesh (glm::vec3 translate_vec, glm::vec3 scale=glm::vec3(1.0f), float angle=0.0f, glm::vec3 rotateAxis=glm::vec3(1.0f, 0.0f, 0.0f))`
- `void expand (std::vector< Vertex > vertices, std::vector< unsigned int > indices)`
- `void changeVertex (std::vector< Vertex > vertices)`
- `void render ()`
- `std::vector< Vertex > getVertices ()`
- `std::vector< unsigned int > getIndices ()`
- `~Mesh ()`

6.22.1 Constructor & Destructor Documentation

6.22.1.1 Mesh() [1/2]

```
Mesh::Mesh (
    std::vector< Vertex > vertices,
    std::vector< unsigned int > indices )
```

Here is the call graph for this function:



6.22.1.2 `Mesh()` [2/2]

```
Mesh::Mesh ( )
```

6.22.1.3 `~Mesh()`

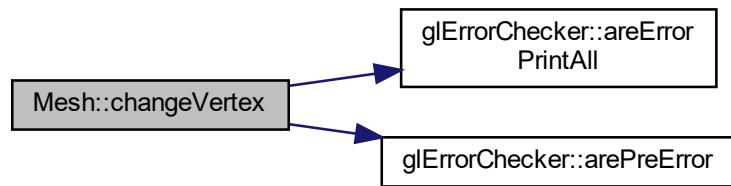
```
Mesh::~Mesh ( )
```

6.22.2 Member Function Documentation

6.22.2.1 `changeVertex()`

```
void Mesh::changeVertex ( std::vector< Vertex > vertices )
```

Here is the call graph for this function:



Here is the caller graph for this function:



6.22.2.2 expand()

```
void Mesh::expand (
    std::vector< Vertex > vertices,
    std::vector< unsigned int > indices )
```

6.22.2.3 getIndices()

```
std::vector< unsigned int > Mesh::getIndices ( ) [inline]
```

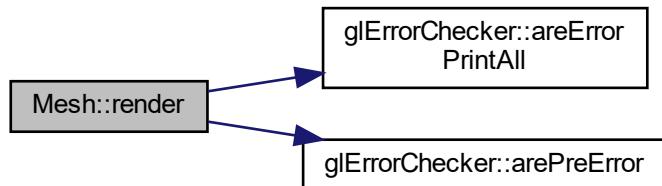
6.22.2.4 getVertices()

```
std::vector< Vertex > Mesh::getVertices ( ) [inline]
```

6.22.2.5 render()

```
void Mesh::render ( )
```

Here is the call graph for this function:



6.22.2.6 transform_Mesh()

```
glm::mat4 Mesh::transform_Mesh (
    glm::vec3 translate_vec,
    glm::vec3 scale = glm::vec3(1.0f),
    float angle = 0.0f,
    glm::vec3 rotateAxis = glm::vec3(1.0f, 0.0f, 0.0f) )
```

Here is the call graph for this function:



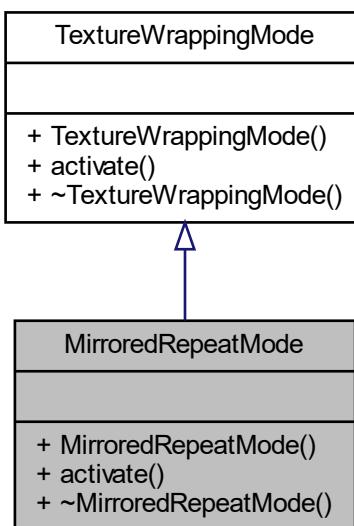
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Mesh.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Mesh.cpp](#)

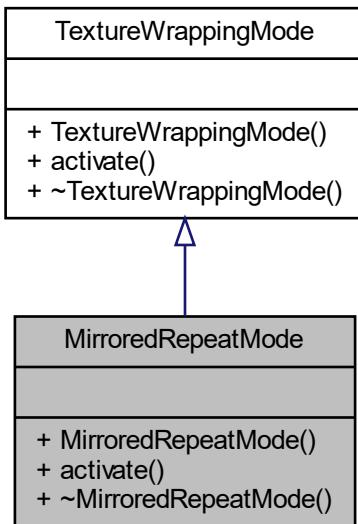
6.23 MirroredRepeatMode Class Reference

```
#include <MirroredRepeatMode.h>
```

Inheritance diagram for MirroredRepeatMode:



Collaboration diagram for MirroredRepeatMode:



Public Member Functions

- [MirroredRepeatMode \(\)](#)
- void [activate \(\)](#)
- [~MirroredRepeatMode \(\)](#)

6.23.1 Constructor & Destructor Documentation

6.23.1.1 [MirroredRepeatMode\(\)](#)

```
MirroredRepeatMode::MirroredRepeatMode ( )
```

6.23.1.2 [~MirroredRepeatMode\(\)](#)

```
MirroredRepeatMode::~MirroredRepeatMode ( )
```

6.23.2 Member Function Documentation

6.23.2.1 activate()

```
void MirroredRepeatMode::activate ( ) [virtual]
```

Implements [TextureWrappingMode](#).

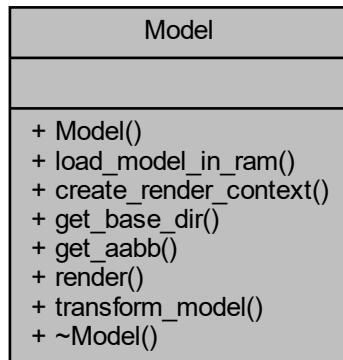
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[MirroredRepeatMode.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[MirroredRepeatMode.cpp](#)

6.24 Model Class Reference

```
#include <Model.h>
```

Collaboration diagram for Model:



Public Member Functions

- [Model \(\)](#)
- void [load_model_in_ram](#) (std::string model_path)
- void [create_render_context \(\)](#)
- std::string [get_base_dir](#) (const std::string &filepath)
- std::shared_ptr< [AABB](#) > [get_aabb \(\)](#)
- void [render \(\)](#)
- void [transform_model](#) (glm::vec3 translate_vec, glm::vec3 scale=glm::vec3(1.0f), float angle=0.0f, glm::vec3 rotateAxis=glm::vec3(1.0f, 0.0f, 0.0f))
- [~Model \(\)](#)

6.24.1 Constructor & Destructor Documentation

6.24.1.1 Model()

```
Model::Model ( )
```

6.24.1.2 ~Model()

```
Model::~Model ( )
```

6.24.2 Member Function Documentation

6.24.2.1 create_render_context()

```
void Model::create_render_context ( )
```

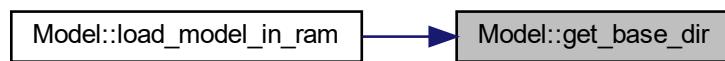
6.24.2.2 get_aabb()

```
std::shared_ptr< AABB > Model::get_aabb ( )
```

6.24.2.3 get_base_dir()

```
std::string Model::get_base_dir (
    const std::string & filepath )
```

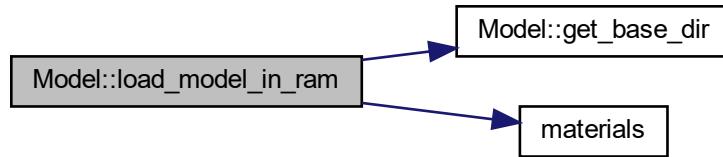
Here is the caller graph for this function:



6.24.2.4 load_model_in_ram()

```
void Model::load_model_in_ram ( std::string model_path )
```

Here is the call graph for this function:



6.24.2.5 render()

```
void Model::render ( )
```

6.24.2.6 transform_model()

```
void Model::transform_model ( glm::vec3 translate_vec, glm::vec3 scale = glm::vec3(1.0f), float angle = 0.0f, glm::vec3 rotateAxis = glm::vec3(1.0f, 0.0f, 0.0f) )
```

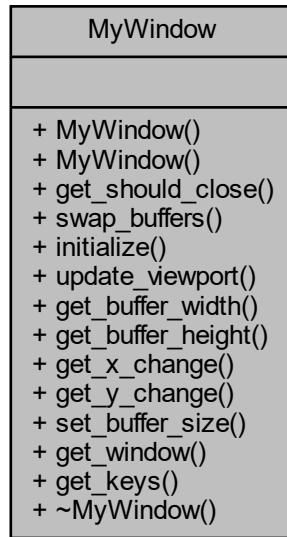
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Model.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Model.cpp](#)

6.25 MyWindow Class Reference

```
#include <MyWindow.h>
```

Collaboration diagram for MyWindow:



Public Member Functions

- [MyWindow \(\)](#)
- [MyWindow \(GLint window_width, GLint window_height\)](#)
- [bool get_should_close \(\)](#)
- [void swap_buffers \(\)](#)
- [int initialize \(\)](#)
- [void update_viewport \(\)](#)
- [GLfloat get_buffer_width \(\)](#)
- [GLfloat get_buffer_height \(\)](#)
- [GLfloat get_x_change \(\)](#)
- [GLfloat get_y_change \(\)](#)
- [void set_buffer_size \(GLfloat window_buffer_width, GLfloat window_buffer_height\)](#)
- [GLFWwindow * get_window \(\)](#)
- [bool * get_keys \(\)](#)
- [~MyWindow \(\)](#)

6.25.1 Constructor & Destructor Documentation

6.25.1.1 MyWindow() [1/2]

```
MyWindow::MyWindow ( )
```

6.25.1.2 MyWindow() [2/2]

```
MyWindow::MyWindow (
    GLint window_width,
    GLint window_height )
```

6.25.1.3 ~MyWindow()

```
MyWindow::~MyWindow ( )
```

6.25.2 Member Function Documentation

6.25.2.1 get_buffer_height()

```
GLfloat MyWindow::get_buffer_height ( ) [inline]
```

6.25.2.2 get_buffer_width()

```
GLfloat MyWindow::get_buffer_width ( ) [inline]
```

6.25.2.3 get_keys()

```
bool * MyWindow::get_keys ( ) [inline]
```

6.25.2.4 get_should_close()

```
bool MyWindow::get_should_close ( ) [inline]
```

6.25.2.5 `get_window()`

```
GLFWwindow * MyWindow::get_window ( ) [inline]
```

6.25.2.6 `get_x_change()`

```
GLfloat MyWindow::get_x_change ( )
```

6.25.2.7 `get_y_change()`

```
GLfloat MyWindow::get_y_change ( )
```

6.25.2.8 `initialize()`

```
int MyWindow::initialize ( )
```

Here is the caller graph for this function:



6.25.2.9 `set_buffer_size()`

```
void MyWindow::set_buffer_size (
    GLfloat window_buffer_width,
    GLfloat window_buffer_height )
```

6.25.2.10 `swap_buffers()`

```
void MyWindow::swap_buffers ( ) [inline]
```

6.25.2.11 update_viewport()

```
void MyWindow::update_viewport ( )
```

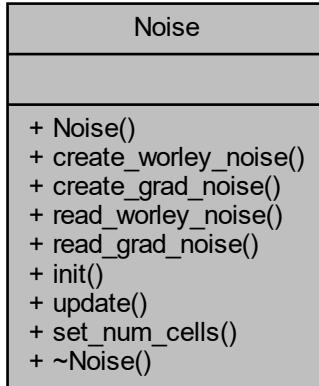
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[MyWindow.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[MyWindow.cpp](#)

6.26 Noise Class Reference

```
#include <Noise.h>
```

Collaboration diagram for Noise:



Public Member Functions

- [Noise \(\)](#)
- void [create_worley_noise \(\)](#)
- void [create_grad_noise \(\)](#)
- void [read_worley_noise \(GLenum start_buffer_index\)](#)
- void [read_grad_noise \(GLenum start_buffer_index\)](#)
- void [init \(\)](#)
- void [update \(\)](#)
- void [set_num_cells \(GLuint num_cells_per_axis, GLuint index\)](#)
- [~Noise \(\)](#)

6.26.1 Constructor & Destructor Documentation

6.26.1.1 Noise()

```
Noise::Noise ( )
```

6.26.1.2 ~Noise()

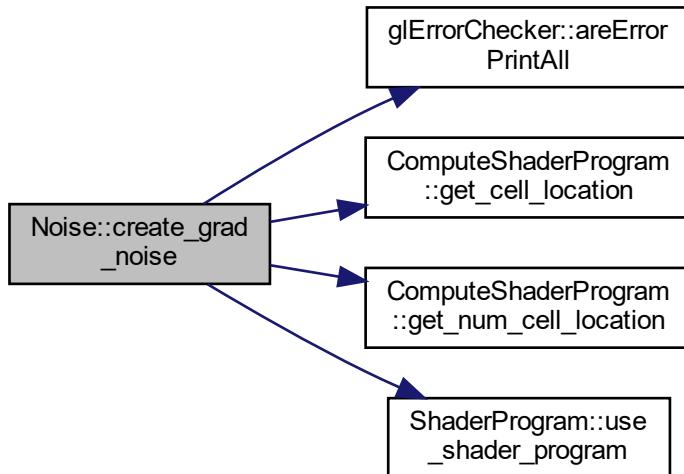
```
Noise::~Noise ( )
```

6.26.2 Member Function Documentation

6.26.2.1 create_grad_noise()

```
void Noise::create_grad_noise ( )
```

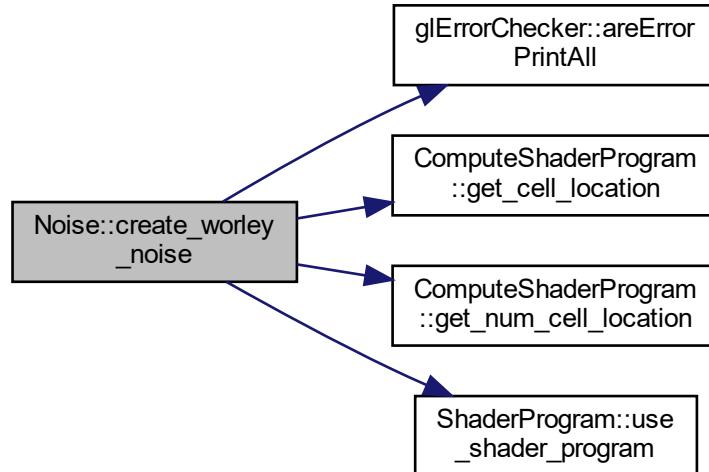
Here is the call graph for this function:



6.26.2.2 create_worley_noise()

```
void Noise::create_worley_noise ( )
```

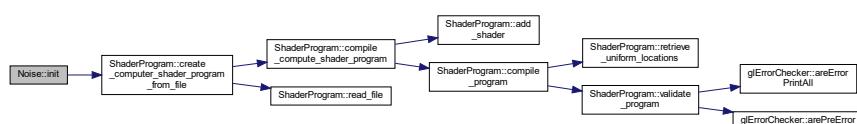
Here is the call graph for this function:



6.26.2.3 init()

```
void Noise::init ( )
```

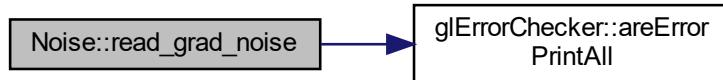
Here is the call graph for this function:



6.26.2.4 `read_grad_noise()`

```
void Noise::read_grad_noise (
    GLenum start_buffer_index )
```

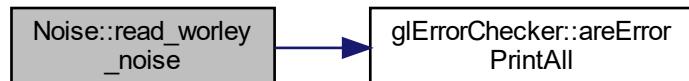
Here is the call graph for this function:



6.26.2.5 `read_worley_noise()`

```
void Noise::read_worley_noise (
    GLenum start_buffer_index )
```

Here is the call graph for this function:



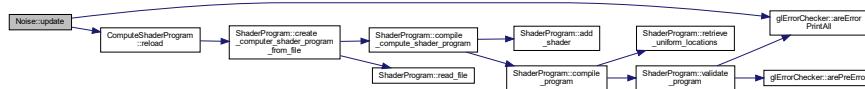
6.26.2.6 `set_num_cells()`

```
void Noise::set_num_cells (
    GLuint num_cells_per_axis,
    GLuint index )
```

6.26.2.7 update()

```
void Noise::update ( )
```

Here is the call graph for this function:



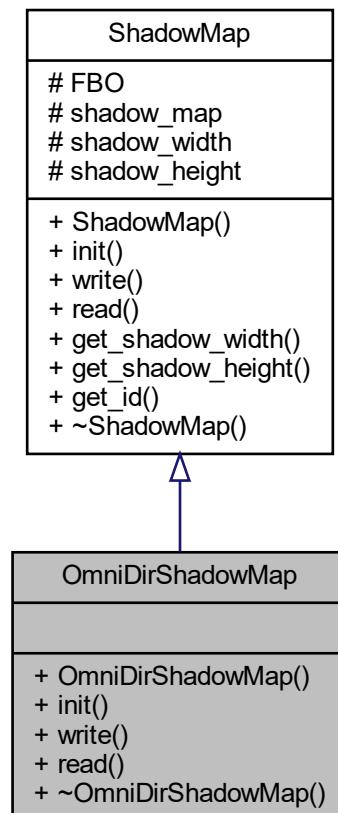
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Noise.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Noise.cpp](#)

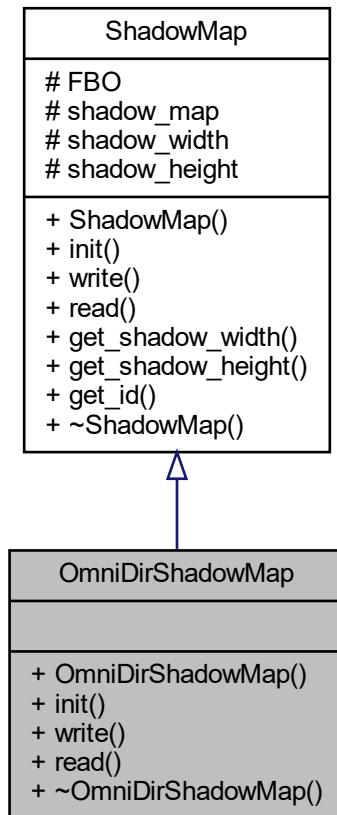
6.27 OmniDirShadowMap Class Reference

```
#include <OmniDirShadowMap.h>
```

Inheritance diagram for OmniDirShadowMap:



Collaboration diagram for OmniDirShadowMap:



Public Member Functions

- [OmniDirShadowMap \(\)](#)
- [bool init \(GLuint width, GLuint height\)](#)
- [void write \(\)](#)
- [void read \(GLenum texture_unit\)](#)
- [~OmniDirShadowMap \(\)](#)

Additional Inherited Members

6.27.1 Constructor & Destructor Documentation

6.27.1.1 OmniDirShadowMap()

```
OmniDirShadowMap::OmniDirShadowMap ( )
```

6.27.1.2 ~OmniDirShadowMap()

```
OmniDirShadowMap::~OmniDirShadowMap ( )
```

6.27.2 Member Function Documentation

6.27.2.1 init()

```
bool OmniDirShadowMap::init (
    GLuint width,
    GLuint height ) [virtual]
```

Reimplemented from [ShadowMap](#).

6.27.2.2 read()

```
void OmniDirShadowMap::read (
    GLenum texture_unit ) [virtual]
```

Reimplemented from [ShadowMap](#).

6.27.2.3 write()

```
void OmniDirShadowMap::write ( ) [virtual]
```

Reimplemented from [ShadowMap](#).

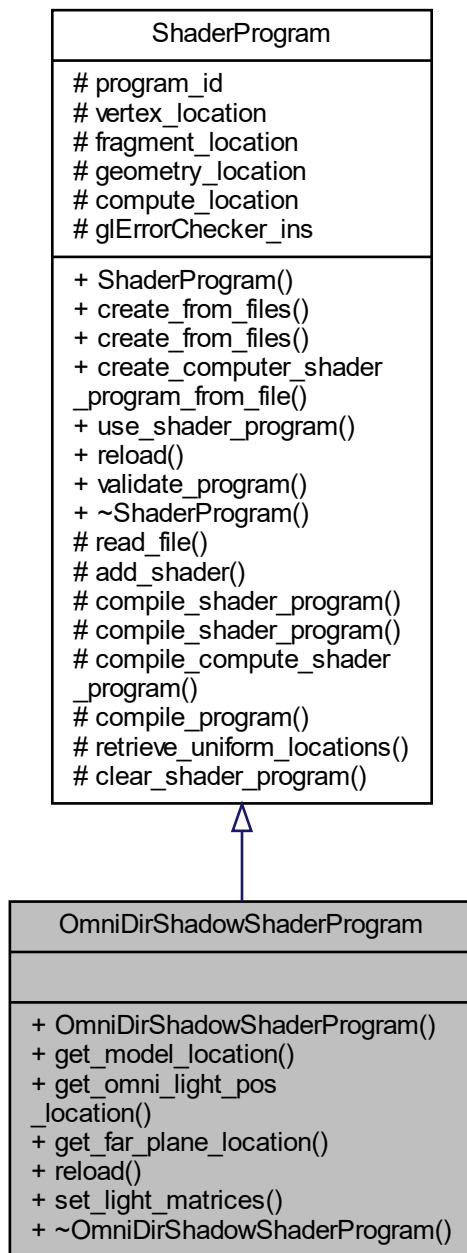
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniDirShadowMap.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniDirShadowMap.cpp](#)

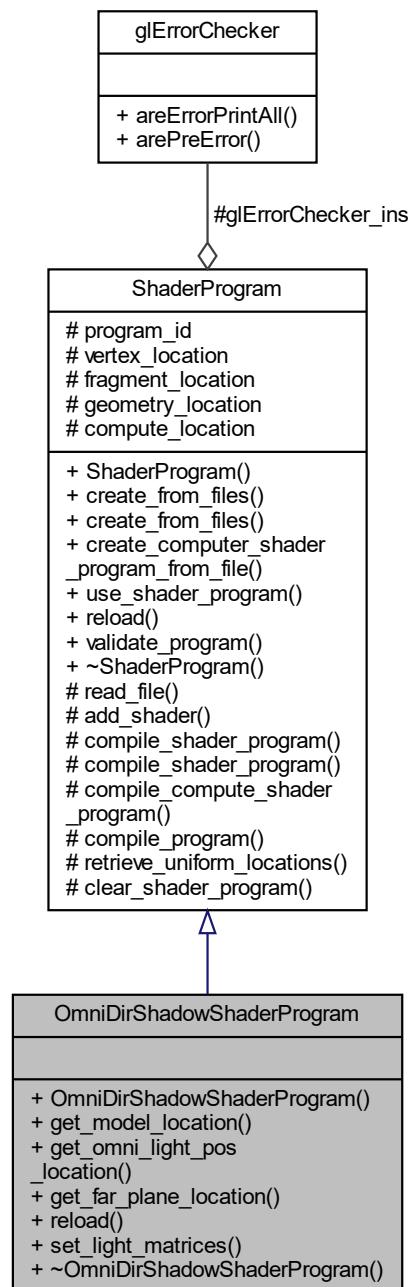
6.28 OmniDirShadowShaderProgram Class Reference

```
#include <OmniDirShadowShaderProgram.h>
```

Inheritance diagram for OmniDirShadowShaderProgram:



Collaboration diagram for OmniDirShadowShaderProgram:



Public Member Functions

- [OmniDirShadowShaderProgram \(\)](#)
- [GLuint get_model_location \(\)](#)
- [GLuint get_omni_light_pos_location \(\)](#)
- [GLuint get_far_plane_location \(\)](#)
- [void reload \(\)](#)

- void [set_light_matrices](#) (std::vector<glm::mat4> light_matrices)
- [~OmniDirShadowShaderProgram](#) ()

Additional Inherited Members

6.28.1 Constructor & Destructor Documentation

6.28.1.1 [OmniDirShadowShaderProgram\(\)](#)

```
OmniDirShadowShaderProgram::OmniDirShadowShaderProgram ( )
```

6.28.1.2 [~OmniDirShadowShaderProgram\(\)](#)

```
OmniDirShadowShaderProgram::~OmniDirShadowShaderProgram ( )
```

6.28.2 Member Function Documentation

6.28.2.1 [get_far_plane_location\(\)](#)

```
GLuint OmniDirShadowShaderProgram::get_far_plane_location ( )
```

6.28.2.2 [get_model_location\(\)](#)

```
GLuint OmniDirShadowShaderProgram::get_model_location ( )
```

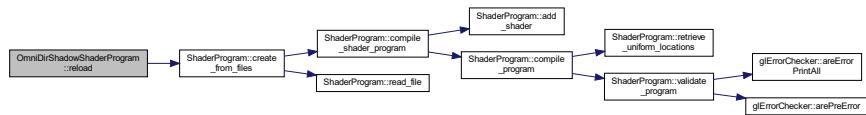
6.28.2.3 [get_omni_light_pos_location\(\)](#)

```
GLuint OmniDirShadowShaderProgram::get_omni_light_pos_location ( )
```

6.28.2.4 reload()

```
void OmniDirShadowShaderProgram::reload ( )
```

Here is the call graph for this function:



6.28.2.5 set_light_matrices()

```
void OmniDirShadowShaderProgram::set_light_matrices (
    std::vector< glm::mat4 > light_matrices )
```

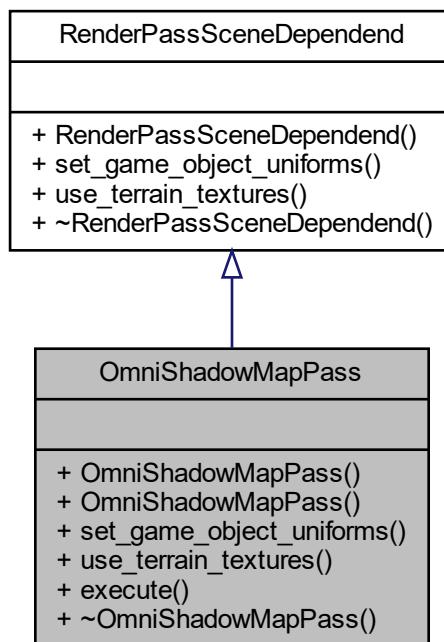
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniDirShadowShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniDirShadowShaderProgram.cpp](#)

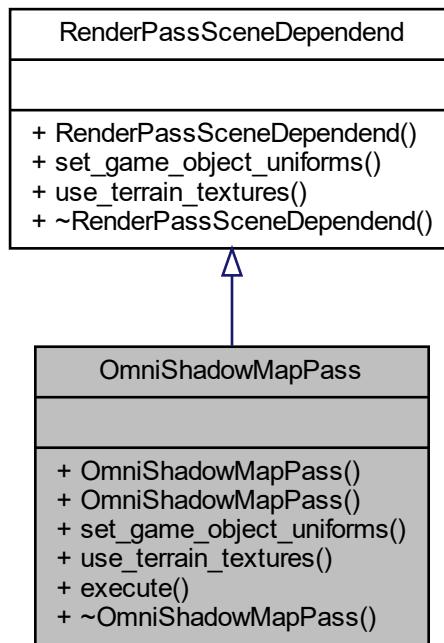
6.29 OmniShadowMapPass Class Reference

```
#include <OmniShadowMapPass.h>
```

Inheritance diagram for OmniShadowMapPass:



Collaboration diagram for OmniShadowMapPass:



Public Member Functions

- `OmniShadowMapPass ()`
- `OmniShadowMapPass (std::shared_ptr<OmniDirShadowShaderProgram> shader_program)`
- `void set_game_object_uniforms (glm::mat4 model, glm::mat4 normal_model, GLuint material_id)`
- `bool use_terrain_textures ()`
- `void execute (std::shared_ptr<PointLight> p_light, bool first_person_mode, Scene *scene)`
- `~OmniShadowMapPass ()`

6.29.1 Constructor & Destructor Documentation

6.29.1.1 OmniShadowMapPass() [1/2]

```
OmniShadowMapPass::OmniShadowMapPass ( )
```

6.29.1.2 OmniShadowMapPass() [2/2]

```
OmniShadowMapPass::OmniShadowMapPass (
    std::shared_ptr< OmniDirShadowShaderProgram > shader_program )
```

6.29.1.3 ~OmniShadowMapPass()

```
OmniShadowMapPass::~OmniShadowMapPass ( )
```

6.29.2 Member Function Documentation

6.29.2.1 execute()

```
void OmniShadowMapPass::execute (
    std::shared_ptr< PointLight > p_light,
    bool first_person_mode,
    Scene * scene )
```

6.29.2.2 set_game_object_uniforms()

```
void OmniShadowMapPass::set_game_object_uniforms (
    glm::mat4 model,
    glm::mat4 normal_model,
    GLuint material_id ) [virtual]
```

Implements [RenderPassSceneDependend](#).

6.29.2.3 use_terrain_textures()

```
bool OmniShadowMapPass::use_terrain_textures ( ) [virtual]
```

Implements [RenderPassSceneDependend](#).

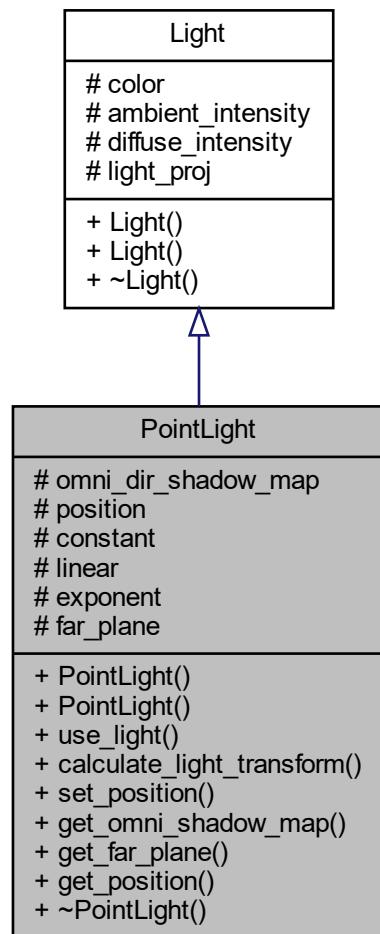
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniShadowMapPass.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[OmniShadowMapPass.cpp](#)

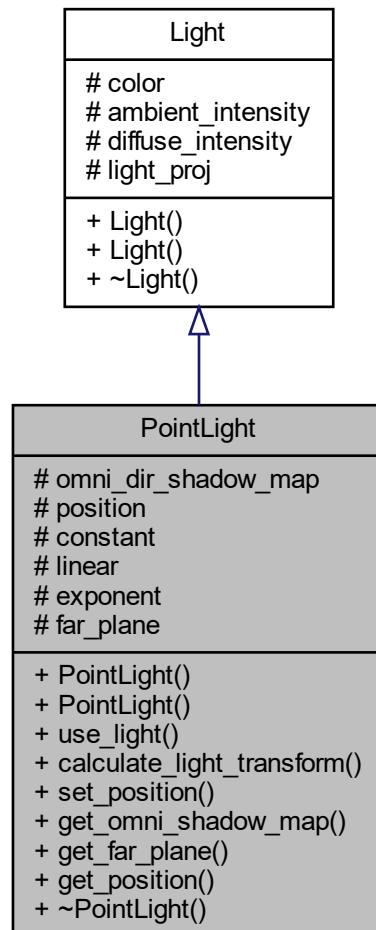
6.30 PointLight Class Reference

```
#include <PointLight.h>
```

Inheritance diagram for PointLight:



Collaboration diagram for PointLight:



Public Member Functions

- `PointLight ()`
- `PointLight (GLfloat shadow_width, GLfloat shadow_height, GLfloat near, GLfloat far, GLfloat red, GLfloat green, GLfloat blue, GLfloat a_intensity, GLfloat d_intensity, GLfloat x_pos, GLfloat y_pos, GLfloat z_pos, GLfloat con, GLfloat lin, GLfloat exp)`
- `void use_light (GLuint ambient_intensity_location, GLuint ambient_color_location, GLuint diffuse_intensity_location, GLuint position_location, GLuint constant_location, GLuint linear_location, GLuint exponent_location)`
- `std::vector<glm::mat4> calculate_light_transform ()`
- `void set_position (glm::vec3 position)`
- `std::shared_ptr<OmniDirShadowMap> get_omni_shadow_map ()`
- `GLfloat get_far_plane ()`
- `glm::vec3 get_position ()`
- `~PointLight ()`

Protected Attributes

- std::shared_ptr< OmniDirShadowMap > omni_dir_shadow_map
- glm::vec3 position
- GLfloat constant
- GLfloat linear
- GLfloat exponent
- GLfloat far_plane

6.30.1 Constructor & Destructor Documentation

6.30.1.1 PointLight() [1/2]

```
PointLight::PointLight ( )
```

6.30.1.2 PointLight() [2/2]

```
PointLight::PointLight (
    GLfloat shadow_width,
    GLfloat shadow_height,
    GLfloat near,
    GLfloat far,
    GLfloat red,
    GLfloat green,
    GLfloat blue,
    GLfloat a_intensity,
    GLfloat d_intensity,
    GLfloat x_pos,
    GLfloat y_pos,
    GLfloat z_pos,
    GLfloat con,
    GLfloat lin,
    GLfloat exp )
```

6.30.1.3 ~PointLight()

```
PointLight::~PointLight ( )
```

6.30.2 Member Function Documentation

6.30.2.1 calculate_light_transform()

```
std::vector< glm::mat4 > PointLight::calculate_light_transform ( )
```

6.30.2.2 get_far_plane()

```
GLfloat PointLight::get_far_plane ( )
```

6.30.2.3 get_omni_shadow_map()

```
std::shared_ptr< OmniDirShadowMap > PointLight::get_omni_shadow_map ( ) [inline]
```

6.30.2.4 get_position()

```
glm::vec3 PointLight::get_position ( )
```

6.30.2.5 set_position()

```
void PointLight::set_position (  
    glm::vec3 position )
```

6.30.2.6 use_light()

```
void PointLight::use_light (   
    GLuint ambient_intensity_location,  
    GLuint ambient_color_location,  
    GLuint diffuse_intensity_location,  
    GLuint position_location,  
    GLuint constant_location,  
    GLuint linear_location,  
    GLuint exponent_location )
```

6.30.3 Member Data Documentation

6.30.3.1 constant

```
GLfloat PointLight::constant [protected]
```

6.30.3.2 exponent

```
GLfloat PointLight::exponent [protected]
```

6.30.3.3 far_plane

```
GLfloat PointLight::far_plane [protected]
```

6.30.3.4 linear

```
GLfloat PointLight::linear [protected]
```

6.30.3.5 omni_dir_shadow_map

```
std::shared_ptr<OmniDirShadowMap> PointLight::omni_dir_shadow_map [protected]
```

6.30.3.6 position

```
glm::vec3 PointLight::position [protected]
```

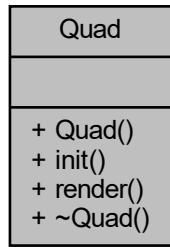
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[PointLight.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[PointLight.cpp](#)

6.31 Quad Class Reference

```
#include <Quad.h>
```

Collaboration diagram for Quad:



Public Member Functions

- [Quad \(\)](#)
- void [init \(\)](#)
- void [render \(\)](#)
- [~Quad \(\)](#)

6.31.1 Constructor & Destructor Documentation

6.31.1.1 Quad()

```
Quad::Quad ( )
```

6.31.1.2 ~Quad()

```
Quad::~Quad ( )
```

6.31.2 Member Function Documentation

6.31.2.1 init()

```
void Quad::init ( )
```

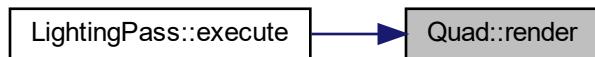
Here is the caller graph for this function:



6.31.2.2 render()

```
void Quad::render ( )
```

Here is the caller graph for this function:



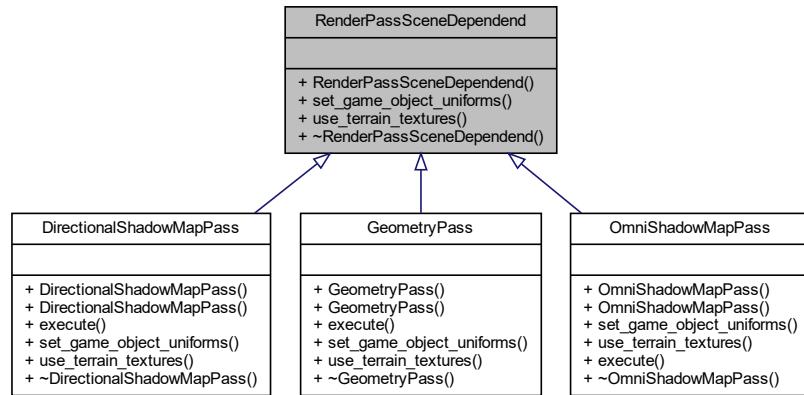
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Quad.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Quad.cpp](#)

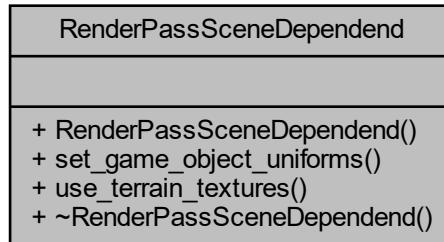
6.32 RenderPassSceneDependend Class Reference

```
#include <RenderPassSceneDependend.h>
```

Inheritance diagram for RenderPassSceneDependend:



Collaboration diagram for RenderPassSceneDependend:



Public Member Functions

- [RenderPassSceneDependend \(\)](#)
- virtual void [set_game_object_uniforms \(glm::mat4 model, glm::mat4 normal_model, GLuint material_id\)=0](#)
- virtual bool [use_terrain_textures \(\)=0](#)
- [~RenderPassSceneDependend \(\)](#)

6.32.1 Constructor & Destructor Documentation

6.32.1.1 RenderPassSceneDependend()

```
RenderPassSceneDependend::RenderPassSceneDependend ( )
```

6.32.1.2 ~RenderPassSceneDependend()

```
RenderPassSceneDependend::~RenderPassSceneDependend ( )
```

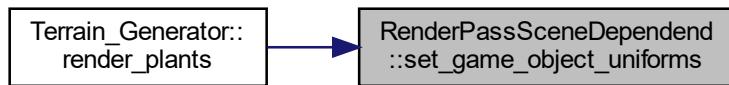
6.32.2 Member Function Documentation

6.32.2.1 set_game_object_uniforms()

```
virtual void RenderPassSceneDependend::set_game_object_uniforms (
    glm::mat4 model,
    glm::mat4 normal_model,
    GLuint material_id ) [pure virtual]
```

Implemented in [DirectionalShadowMapPass](#), [GeometryPass](#), and [OmniShadowMapPass](#).

Here is the caller graph for this function:



6.32.2.2 use_terrain_textures()

```
virtual bool RenderPassSceneDependend::use_terrain_textures ( ) [pure virtual]
```

Implemented in [DirectionalShadowMapPass](#), [GeometryPass](#), and [OmniShadowMapPass](#).

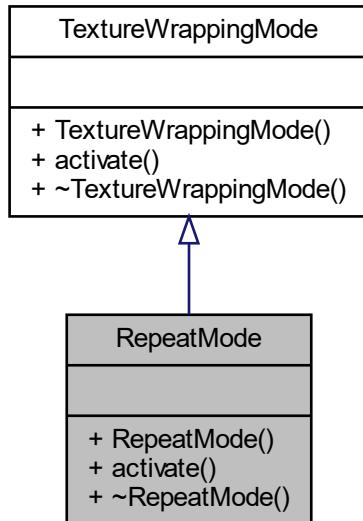
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[RenderPassSceneDependend.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[RenderPassSceneDependend.cpp](#)

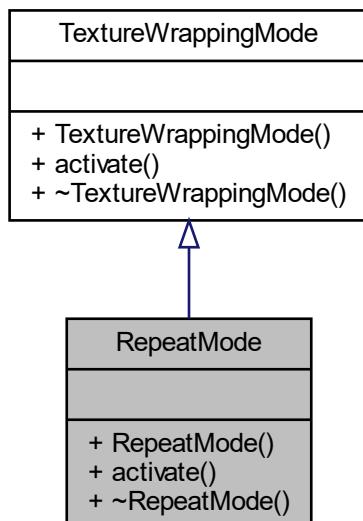
6.33 RepeatMode Class Reference

```
#include <RepeatMode.h>
```

Inheritance diagram for RepeatMode:



Collaboration diagram for RepeatMode:



Public Member Functions

- [RepeatMode \(\)](#)
- void [activate \(\)](#)
- [~RepeatMode \(\)](#)

6.33.1 Constructor & Destructor Documentation

6.33.1.1 RepeatMode()

```
RepeatMode::RepeatMode ( )
```

6.33.1.2 ~RepeatMode()

```
RepeatMode::~RepeatMode ( )
```

6.33.2 Member Function Documentation

6.33.2.1 activate()

```
void RepeatMode::activate ( ) [virtual]
```

Implements [TextureWrappingMode](#).

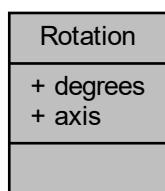
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[RepeatMode.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[RepeatMode.cpp](#)

6.34 Rotation Struct Reference

```
#include <GlobalValues.h>
```

Collaboration diagram for Rotation:



Public Attributes

- GLfloat [degrees](#)
- glm::vec3 [axis](#)

6.34.1 Member Data Documentation

6.34.1.1 axis

```
glm::vec3 Rotation::axis
```

6.34.1.2 degrees

```
GLfloat Rotation::degrees
```

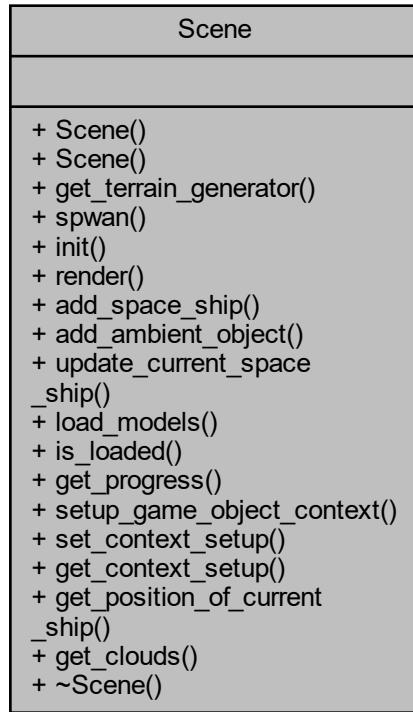
The documentation for this struct was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[GlobalValues.h](#)

6.35 Scene Class Reference

```
#include <Scene.h>
```

Collaboration diagram for Scene:



Public Member Functions

- `Scene ()`
- `Scene (const Scene &other)`
- `std::shared_ptr< Terrain_Generator > get_terrain_generator ()`
- `std::thread spwan ()`
- `void init (std::shared_ptr< Camera > main_camera, MyWindow *main_window, std::shared_ptr< Terrain_Generator > terrain_generator, std::shared_ptr< Clouds > clouds)`
- `void render (RenderPassSceneDependend *render_pass, bool first_person_mode)`
- `void add_space_ship (std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint material_id)`
- `void add_ambient_object (std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint material_id)`
- `void update_current_space_ship (GLuint selected_space_ship)`
- `void load_models ()`
- `bool is_loaded ()`
- `GLfloat get_progress ()`
- `void setup_game_object_context ()`
- `void set_context_setup (bool context_setup)`
- `bool get_context_setup ()`
- `glm::vec3 get_position_of_current_ship ()`
- `std::shared_ptr< Clouds > get_clouds ()`
- `~Scene ()`

6.35.1 Constructor & Destructor Documentation

6.35.1.1 Scene() [1/2]

```
Scene::Scene ( )
```

6.35.1.2 Scene() [2/2]

```
Scene::Scene (
    const Scene & other )
```

6.35.1.3 ~Scene()

```
Scene::~Scene ( )
```

6.35.2 Member Function Documentation

6.35.2.1 add_ambient_object()

```
void Scene::add_ambient_object (
    std::string model_path,
    glm::vec3 translation,
    GLfloat scale,
    Rotation rot,
    GLuint material_id )
```

6.35.2.2 add_space_ship()

```
void Scene::add_space_ship (
    std::string model_path,
    glm::vec3 translation,
    GLfloat scale,
    Rotation rot,
    GLuint material_id )
```

6.35.2.3 get_clouds()

```
std::shared_ptr< Clouds > Scene::get_clouds ( )
```

6.35.2.4 get_context_setup()

```
bool Scene::get_context_setup ( )
```

6.35.2.5 get_position_of_current_ship()

```
glm::vec3 Scene::get_position_of_current_ship ( )
```

6.35.2.6 get_progress()

```
GLfloat Scene::get_progress ( )
```

6.35.2.7 get_terrain_generator()

```
std::shared_ptr< Terrain_Generator > Scene::get_terrain_generator ( )
```

6.35.2.8 init()

```
void Scene::init ( 
    std::shared_ptr< Camera > main_camera,
    MyWindow * main_window,
    std::shared_ptr< Terrain_Generator > terrain_generator,
    std::shared_ptr< Clouds > clouds )
```

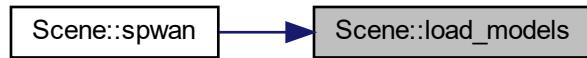
6.35.2.9 is_loaded()

```
bool Scene::is_loaded ( )
```

6.35.2.10 load_models()

```
void Scene::load_models ( )
```

Here is the caller graph for this function:



6.35.2.11 render()

```
void Scene::render (
    RenderPassSceneDependend * render_pass,
    bool first_person_mode )
```

6.35.2.12 set_context_setup()

```
void Scene::set_context_setup (
    bool context_setup )
```

6.35.2.13 setup_game_object_context()

```
void Scene::setup_game_object_context ( )
```

6.35.2.14 spwan()

```
std::thread Scene::spwan ( ) [inline]
```

Here is the call graph for this function:



6.35.2.15 update_current_space_ship()

```
void Scene::update_current_space_ship (
```

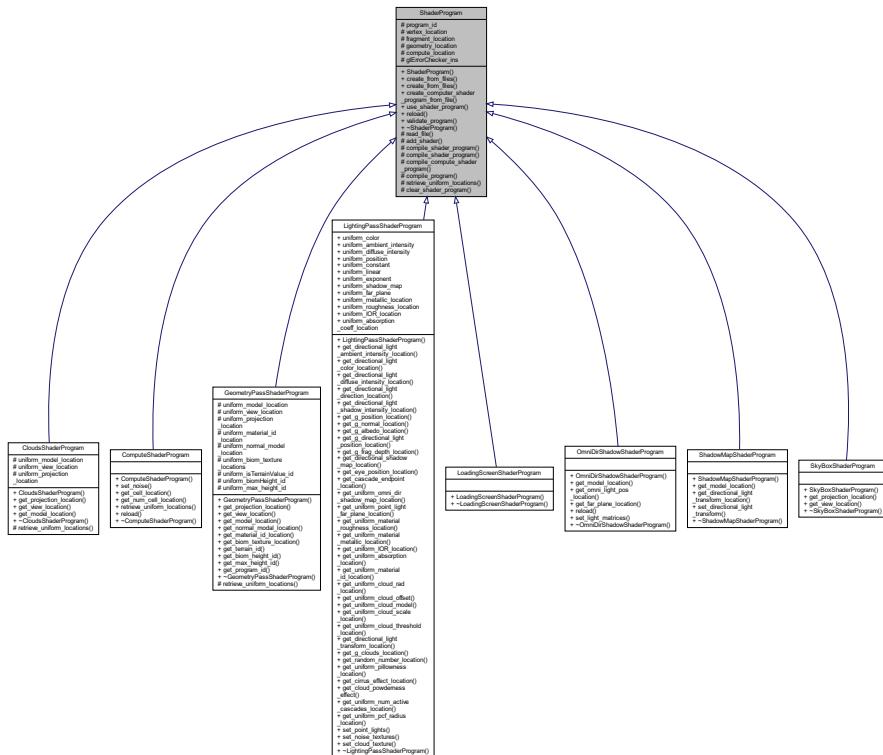
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Scene.h](#)
 - C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Scene.cpp](#)

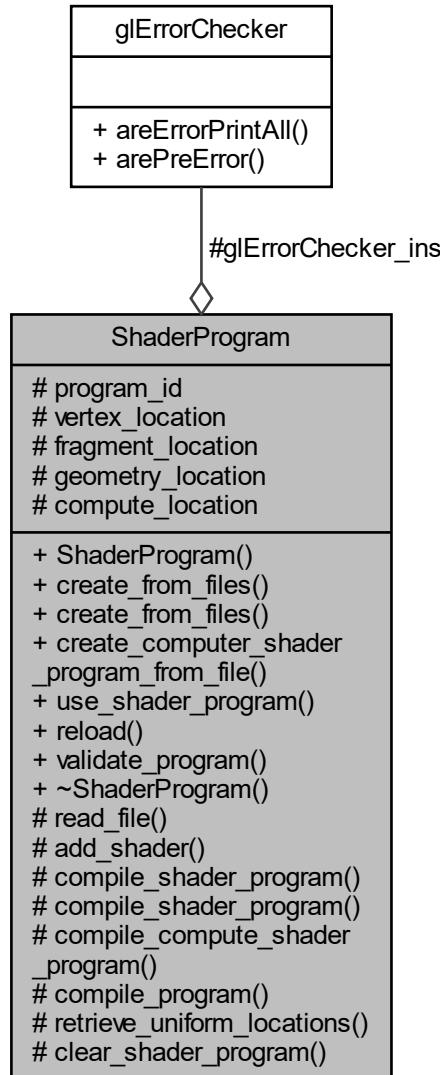
6.36 ShaderProgram Class Reference

```
#include <ShaderProgram.h>
```

Inheritance diagram for ShaderProgram:



Collaboration diagram for ShaderProgram:



Public Member Functions

- `ShaderProgram ()`
- `void create_from_files (const char *vertex_location, const char *fragment_location)`
- `void create_from_files (const char *vertex_location, const char *geometry_location, const char *fragment_location)`
- `void create_computer_shader_program_from_file (const char *compute_location)`
- `void use_shader_program ()`
- `void reload ()`
- `void validate_program ()`
- `~ShaderProgram ()`

Protected Member Functions

- std::string `read_file` (const char *file_location)
- void `add_shader` (GLuint program, const char *shader_code, GLenum shader_type)
- void `compile_shader_program` (const char *vertex_code, const char *fragment_code)
- void `compile_shader_program` (const char *vertex_code, const char *geometry_code, const char *fragment_code)
- void `compile_compute_shader_program` (const char *compute_code)
- void `compile_program` ()
- virtual void `retrieve_uniform_locations` ()=0
- void `clear_shader_program` ()

Protected Attributes

- GLuint `program_id`
- const char * `vertex_location`
- const char * `fragment_location`
- const char * `geometry_location`
- const char * `compute_location`
- glErrorChecker `glErrorChecker_ins`

6.36.1 Constructor & Destructor Documentation

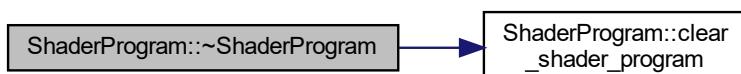
6.36.1.1 `ShaderProgram()`

```
ShaderProgram::ShaderProgram ( )
```

6.36.1.2 `~ShaderProgram()`

```
ShaderProgram::~ShaderProgram ( )
```

Here is the call graph for this function:

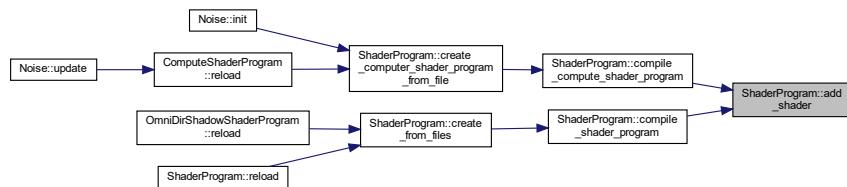


6.36.2 Member Function Documentation

6.36.2.1 add_shader()

```
void ShaderProgram::add_shader (
    GLuint program,
    const char * shader_code,
    GLenum shader_type ) [protected]
```

Here is the caller graph for this function:



6.36.2.2 clear_shader_program()

```
void ShaderProgram::clear_shader_program () [protected]
```

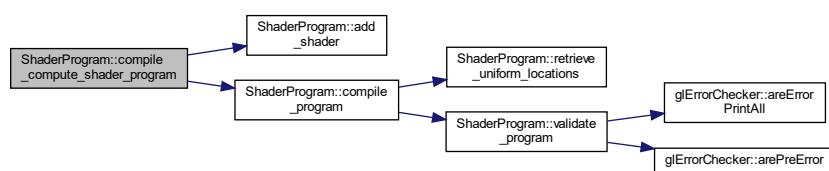
Here is the caller graph for this function:



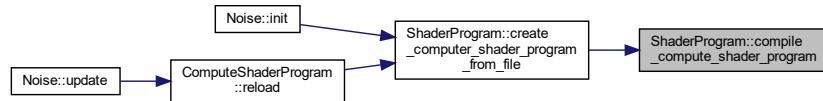
6.36.2.3 compile_compute_shader_program()

```
void ShaderProgram::compile_compute_shader_program (
    const char * compute_code ) [protected]
```

Here is the call graph for this function:



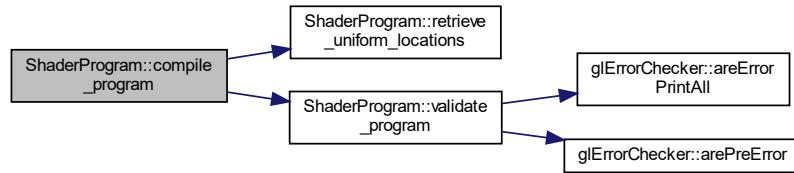
Here is the caller graph for this function:



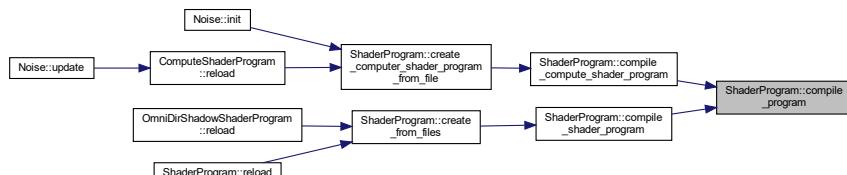
6.36.2.4 compile_program()

```
void ShaderProgram::compile_program () [protected]
```

Here is the call graph for this function:



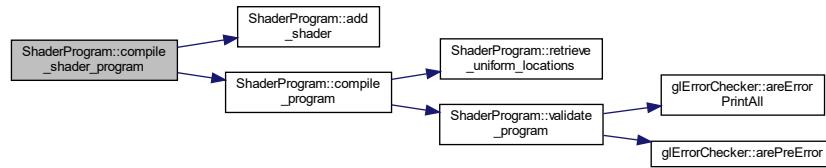
Here is the caller graph for this function:



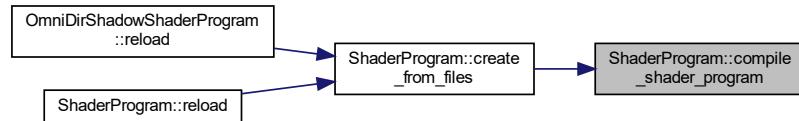
6.36.2.5 compile_shader_program() [1/2]

```
void ShaderProgram::compile_shader_program (
    const char * vertex_code,
    const char * fragment_code ) [protected]
```

Here is the call graph for this function:



Here is the caller graph for this function:

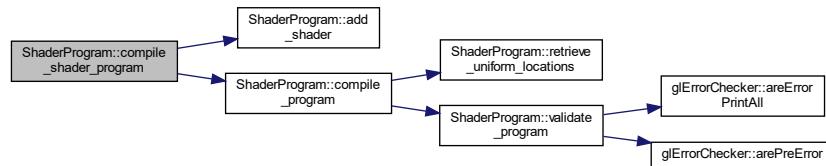


6.36.2.6 compile_shader_program() [2/2]

```

void ShaderProgram::compile_shader_program (
    const char * vertex_code,
    const char * geometry_code,
    const char * fragment_code )  [protected]
  
```

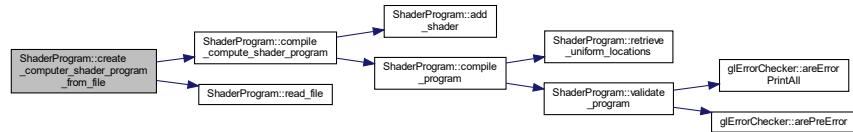
Here is the call graph for this function:



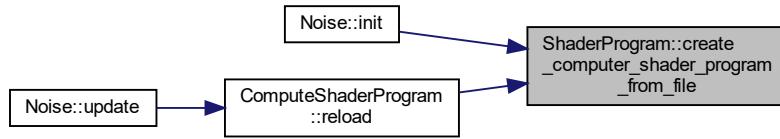
6.36.2.7 create_computer_shader_program_from_file()

```
void ShaderProgram::create_computer_shader_program_from_file (
    const char * compute_location )
```

Here is the call graph for this function:



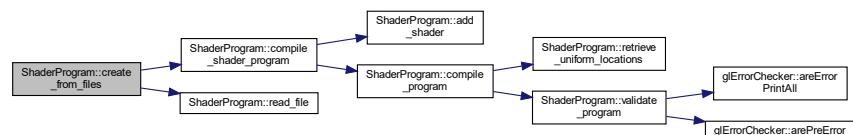
Here is the caller graph for this function:



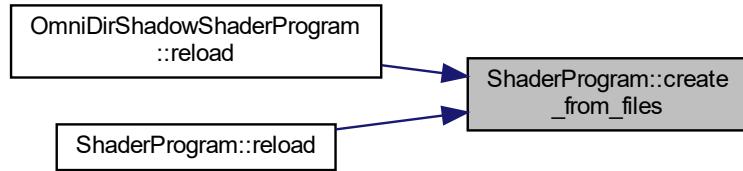
6.36.2.8 create_from_files() [1/2]

```
void ShaderProgram::create_from_files (
    const char * vertex_location,
    const char * fragment_location )
```

Here is the call graph for this function:



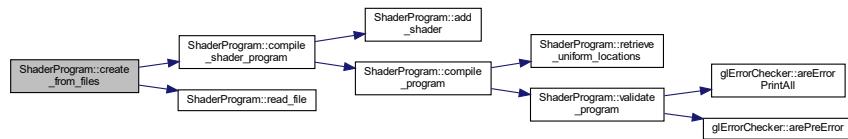
Here is the caller graph for this function:



6.36.2.9 create_from_files() [2/2]

```
void ShaderProgram::create_from_files (
    const char * vertex_location,
    const char * geometry_location,
    const char * fragment_location )
```

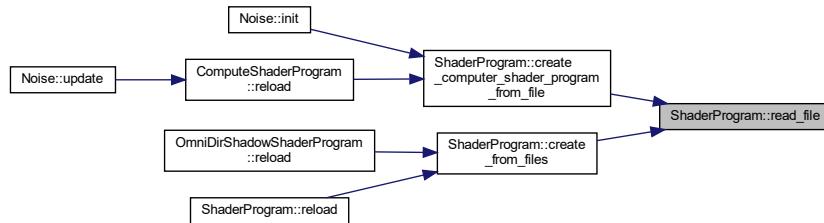
Here is the call graph for this function:



6.36.2.10 read_file()

```
std::string ShaderProgram::read_file (
    const char * file_location ) [protected]
```

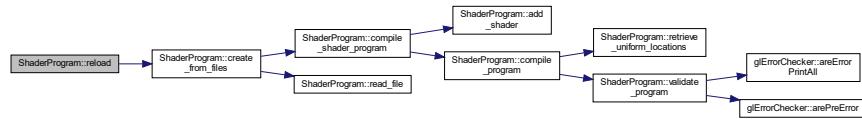
Here is the caller graph for this function:



6.36.2.11 reload()

```
void ShaderProgram::reload ( )
```

Here is the call graph for this function:

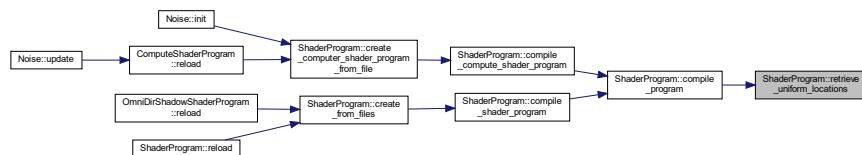


6.36.2.12 retrieve_uniform_locations()

```
virtual void ShaderProgram::retrieve_uniform_locations ( ) [protected], [pure virtual]
```

Implemented in [CloudsShaderProgram](#), [ComputeShaderProgram](#), and [GeometryPassShaderProgram](#).

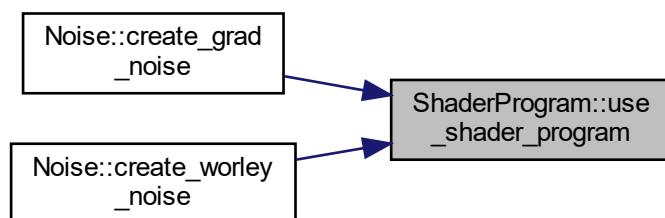
Here is the caller graph for this function:



6.36.2.13 use_shader_program()

```
void ShaderProgram::use_shader_program ( )
```

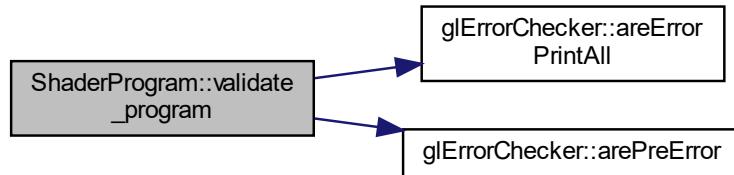
Here is the caller graph for this function:



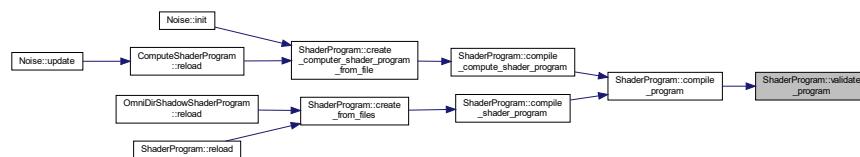
6.36.2.14 validate_program()

```
void ShaderProgram::validate_program( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



6.36.3 Member Data Documentation

6.36.3.1 compute_location

```
const char* ShaderProgram::compute_location [protected]
```

6.36.3.2 fragment_location

```
const char* ShaderProgram::fragment_location [protected]
```

6.36.3.3 geometry_location

```
const char* ShaderProgram::geometry_location [protected]
```

6.36.3.4 glErrorChecker_ins

```
glErrorChecker ShaderProgram::glErrorChecker_ins [protected]
```

6.36.3.5 program_id

```
GLuint ShaderProgram::program_id [protected]
```

6.36.3.6 vertex_location

```
const char* ShaderProgram::vertex_location [protected]
```

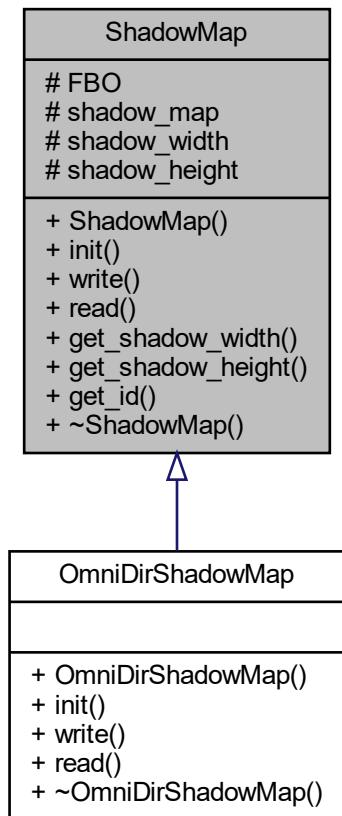
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShaderProgram.cpp](#)

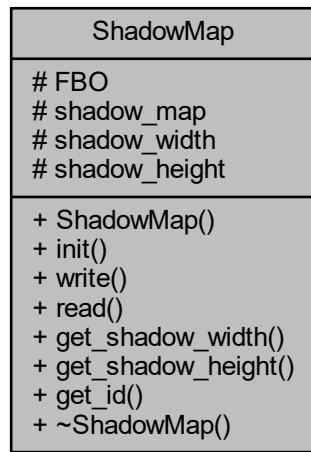
6.37 ShadowMap Class Reference

```
#include <ShadowMap.h>
```

Inheritance diagram for ShadowMap:



Collaboration diagram for ShadowMap:



Public Member Functions

- [ShadowMap \(\)](#)
- virtual bool [init \(GLuint width, GLuint height\)](#)
- virtual void [write \(\)](#)
- virtual void [read \(GLenum texture_unit\)](#)
- GLuint [get_shadow_width \(\)](#)
- GLuint [get_shadow_height \(\)](#)
- GLuint [get_id \(\)](#)
- [~ShadowMap \(\)](#)

Protected Attributes

- GLuint [FBO](#)
- GLuint [shadow_map](#)
- GLuint [shadow_width](#)
- GLuint [shadow_height](#)

6.37.1 Constructor & Destructor Documentation

6.37.1.1 ShadowMap()

```
ShadowMap::ShadowMap ( )
```

6.37.1.2 ~ShadowMap()

```
ShadowMap::~ShadowMap ( )
```

6.37.2 Member Function Documentation

6.37.2.1 get_id()

```
GLuint ShadowMap::get_id ( )
```

6.37.2.2 get_shadow_height()

```
GLuint ShadowMap::get_shadow_height ( ) [inline]
```

6.37.2.3 get_shadow_width()

```
GLuint ShadowMap::get_shadow_width ( ) [inline]
```

6.37.2.4 init()

```
bool ShadowMap::init (
    GLuint width,
    GLuint height ) [virtual]
```

Reimplemented in [OmniDirShadowMap](#).

6.37.2.5 read()

```
void ShadowMap::read (
    GLenum texture_unit ) [virtual]
```

Reimplemented in [OmniDirShadowMap](#).

6.37.2.6 write()

```
void ShadowMap::write ( ) [virtual]
```

Reimplemented in [OmniDirShadowMap](#).

6.37.3 Member Data Documentation

6.37.3.1 FBO

```
GLuint ShadowMap::FBO [protected]
```

6.37.3.2 shadow_height

```
GLuint ShadowMap::shadow_height [protected]
```

6.37.3.3 shadow_map

```
GLuint ShadowMap::shadow_map [protected]
```

6.37.3.4 shadow_width

```
GLuint ShadowMap::shadow_width [protected]
```

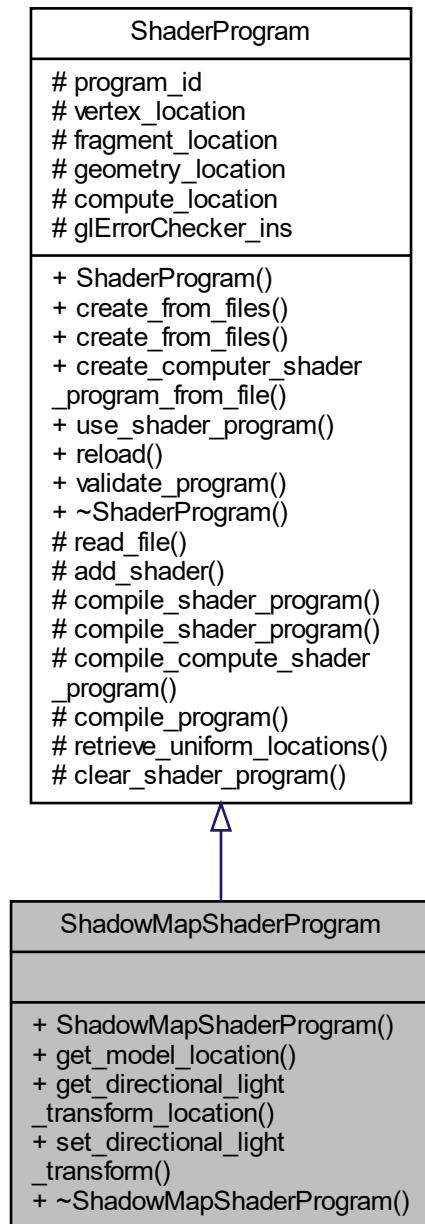
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShadowMap.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShadowMap.cpp](#)

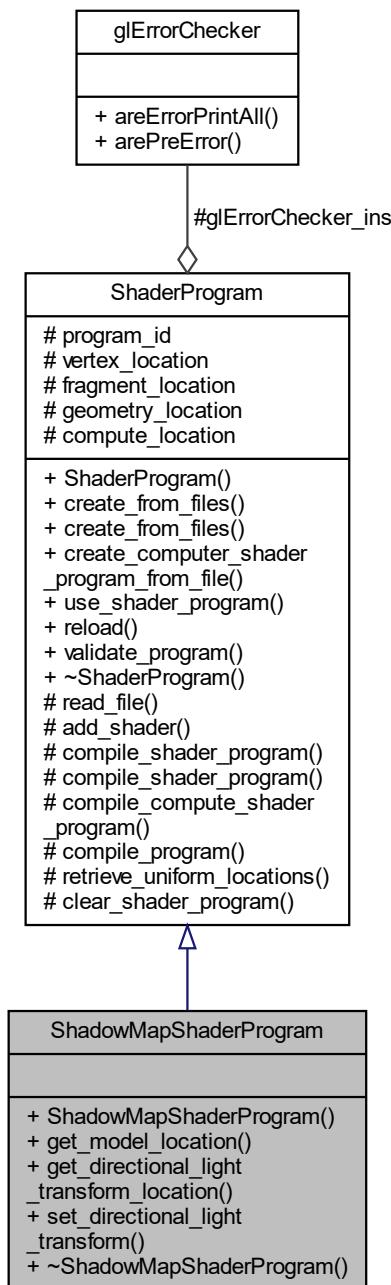
6.38 ShadowMapShaderProgram Class Reference

```
#include <ShadowMapShaderProgram.h>
```

Inheritance diagram for ShadowMapShaderProgram:



Collaboration diagram for ShadowMapShaderProgram:



Public Member Functions

- `ShadowMapShaderProgram ()`
- `GLuint get_model_location ()`
- `GLuint get_directional_light_transform_location ()`
- `void set_directional_light_transform (glm::mat4 &l_traf)`
- `~ShadowMapShaderProgram ()`

Additional Inherited Members

6.38.1 Constructor & Destructor Documentation

6.38.1.1 `ShadowMapShaderProgram()`

```
ShadowMapShaderProgram::ShadowMapShaderProgram ( )
```

6.38.1.2 `~ShadowMapShaderProgram()`

```
ShadowMapShaderProgram::~ShadowMapShaderProgram ( )
```

6.38.2 Member Function Documentation

6.38.2.1 `get_directional_light_transform_location()`

```
GLuint ShadowMapShaderProgram::get_directional_light_transform_location ( )
```

6.38.2.2 `get_model_location()`

```
GLuint ShadowMapShaderProgram::get_model_location ( )
```

6.38.2.3 `set_directional_light_transform()`

```
void ShadowMapShaderProgram::set_directional_light_transform (
    glm::mat4 & l_traf )
```

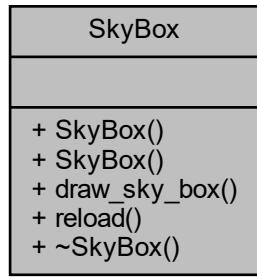
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShadowMapShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ShadowMapShaderProgram.cpp](#)

6.39 SkyBox Class Reference

```
#include <SkyBox.h>
```

Collaboration diagram for SkyBox:



Public Member Functions

- [SkyBox \(\)](#)
- [SkyBox \(std::vector< std::string > face_locations\)](#)
- void [draw_sky_box \(glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat window_height, GLfloat delta_time\)](#)
- void [reload \(\)](#)
- [~SkyBox \(\)](#)

6.39.1 Constructor & Destructor Documentation

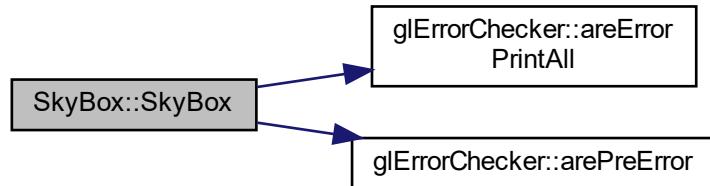
6.39.1.1 SkyBox() [1/2]

```
SkyBox::SkyBox ( )
```

6.39.1.2 SkyBox() [2/2]

```
SkyBox::SkyBox ( std::vector< std::string > face_locations )
```

Here is the call graph for this function:



6.39.1.3 ~SkyBox()

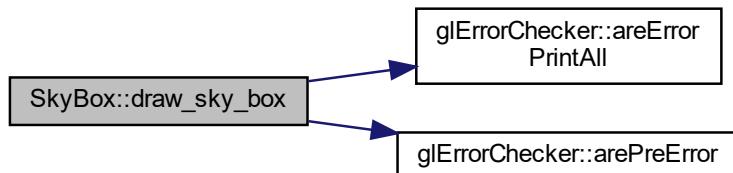
```
SkyBox::~SkyBox ( )
```

6.39.2 Member Function Documentation

6.39.2.1 draw_sky_box()

```
void SkyBox::draw_sky_box ( glm::mat4 projection_matrix,
                            glm::mat4 view_matrix,
                            GLfloat window_width,
                            GLfloat window_height,
                            GLfloat delta_time )
```

Here is the call graph for this function:



6.39.2.2 reload()

```
void SkyBox::reload ( )
```

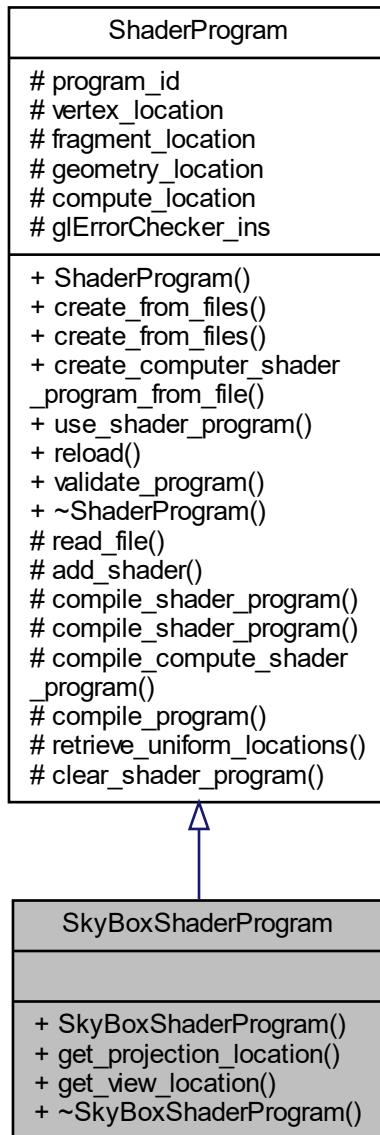
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.h
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.cpp

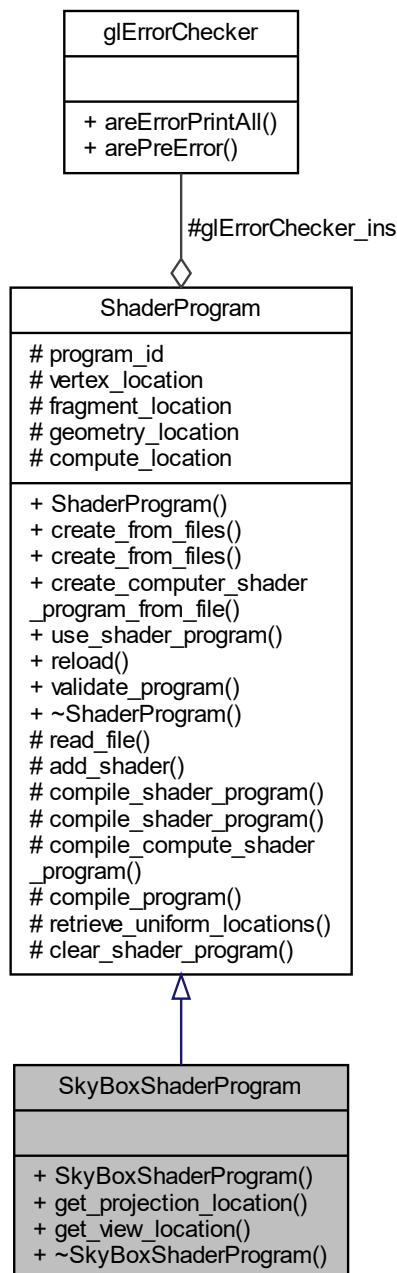
6.40 SkyBoxShaderProgram Class Reference

```
#include <SkyBoxShaderProgram.h>
```

Inheritance diagram for SkyBoxShaderProgram:



Collaboration diagram for SkyBoxShaderProgram:



Public Member Functions

- [SkyBoxShaderProgram \(\)](#)
- [GLuint get_projection_location \(\)](#)
- [GLuint get_view_location \(\)](#)
- [~SkyBoxShaderProgram \(\)](#)

Additional Inherited Members

6.40.1 Constructor & Destructor Documentation

6.40.1.1 SkyBoxShaderProgram()

```
SkyBoxShaderProgram::SkyBoxShaderProgram ( )
```

6.40.1.2 ~SkyBoxShaderProgram()

```
SkyBoxShaderProgram::~SkyBoxShaderProgram ( )
```

6.40.2 Member Function Documentation

6.40.2.1 get_projection_location()

```
GLuint SkyBoxShaderProgram::get_projection_location ( )
```

6.40.2.2 get_view_location()

```
GLuint SkyBoxShaderProgram::get_view_location ( )
```

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

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[SkyBoxShaderProgram.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[SkyBoxShaderProgram.cpp](#)

6.41 Terrain_Generator Class Reference

```
#include <Terrain_generator.h>
```

Collaboration diagram for Terrain_Generator:



Public Member Functions

- [Terrain_Generator \(\)](#)
- void [init \(\)](#)
- void [retreive_uniform_locations \(std::shared_ptr< GeometryPassShaderProgram > shader_program\)](#)
- [~Terrain_Generator \(\)](#)
- void [set_world_trafo \(glm::mat4 model_to_world\)](#)
- glm::mat4 [get_world_trafo \(\)](#)
- glm::mat4 [get_normal_world_trafo \(\)](#)
- void [set_material_id \(GLuint material_id\)](#)

- GLuint [get_material_id \(\)](#)
- std::shared_ptr< [Terrain_Texture](#) > [get_textures \(\)](#)
- void [generateMesh \(GLfloat &progress\)](#)
- void [load_plants \(\)](#)
- void [expandTerrain \(\)](#)
- std::vector< GLuint > [generate_indices \(\)](#)
- std::vector< float > [generate_noise_map \(int xOffset, int yOffset\)](#)
- std::vector< glm::vec3 > [generate_vertices \(const std::vector< float > &noise_map\)](#)

```
return vertices = {x0, noise0, y0, x1, noise1, y1, x2, noise2, y2, ....}
```
- std::vector< glm::vec3 > [calc_smooth_normals \(std::vector< glm::vec3 > verticesPos, std::vector< glm::vec3 > triangleNormals, std::vector< unsigned int > indices\)](#)
- std::vector< glm::vec3 > [generate_normals \(const std::vector< GLuint > &indices, const std::vector< glm::vec3 > &vertices\)](#)
- void [generate_map_chunk \(int xOffset, int yOffset, std::vector< std::vector< \[Terrain_Model\]\(#\) > > &chunk_models\)](#)
- std::vector< glm::vec2 > [generate_biome \(const std::vector< glm::vec3 > &vertices, const std::vector< GLuint > &indices, int xOffset, int yOffset, std::vector< std::vector< \[Terrain_Model\]\(#\) > > &chunk_models\)](#)
- void [generate_render_context \(\)](#)
- std::vector< bool > [render \(GLfloat ratio, std::shared_ptr< \[Camera\]\(#\) > camera, std::shared_ptr< \[ViewFrustumCulling\]\(#\) > view_cull, \[RenderPassSceneDependend\]\(#\) *shader_pass\)](#)

```
dynamic render Terrain TODO parallel loading?
```
- void [render_plants \(std::vector< bool > is_chunk_rendered_flags, \[RenderPassSceneDependend\]\(#\) *shader_pass\)](#)
- void [transform \(glm::vec3 translate_vector, glm::vec3 scale\)](#)
- void [changeMaxHeight \(float newMaxHeight\)](#)
- void [regenerate \(\)](#)
- std::tuple< int, float, float, float, float > [getNoiseParameters \(\)](#)
- void [setNoiseParameters \(std::tuple< int, float, float, float, float > noiseParam\)](#)

Public Attributes

- std::vector< std::shared_ptr< [Mesh](#) > > [meshes](#)
- std::vector< std::shared_ptr< [AABB](#) > > [aabbs](#)

6.41.1 Constructor & Destructor Documentation

6.41.1.1 [Terrain_Generator\(\)](#)

```
Terrain_Generator::Terrain_Generator ( )
```

6.41.1.2 [~Terrain_Generator\(\)](#)

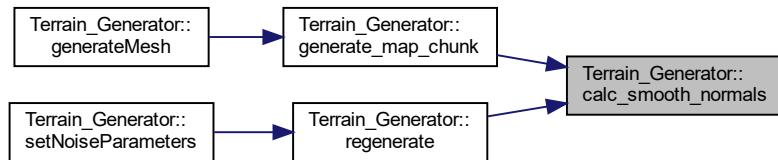
```
Terrain_Generator::~Terrain_Generator ( )
```

6.41.2 Member Function Documentation

6.41.2.1 calc_smooth_normals()

```
std::vector< glm::vec3 > Terrain_Generator::calc_smooth_normals (
    std::vector< glm::vec3 > verticesPos,
    std::vector< glm::vec3 > triangleNormals,
    std::vector< unsigned int > indices )
```

Here is the caller graph for this function:



6.41.2.2 changeMaxHeight()

```
void Terrain_Generator::changeMaxHeight (
    float newMaxHeight )
```

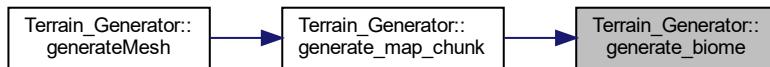
6.41.2.3 expandTerrain()

```
void Terrain_Generator::expandTerrain ( )
```

6.41.2.4 generate_biome()

```
std::vector< glm::vec2 > Terrain_Generator::generate_biome (
    const std::vector< glm::vec3 > & vertices,
    const std::vector< GLuint > & indices,
    int xOffset,
    int yOffset,
    std::vector< std::vector< Terrain_Model > > & chunk_models )
```

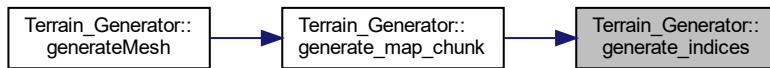
Here is the caller graph for this function:



6.41.2.5 generate_indices()

```
std::vector< GLuint > Terrain_Generator::generate_indices ( )
```

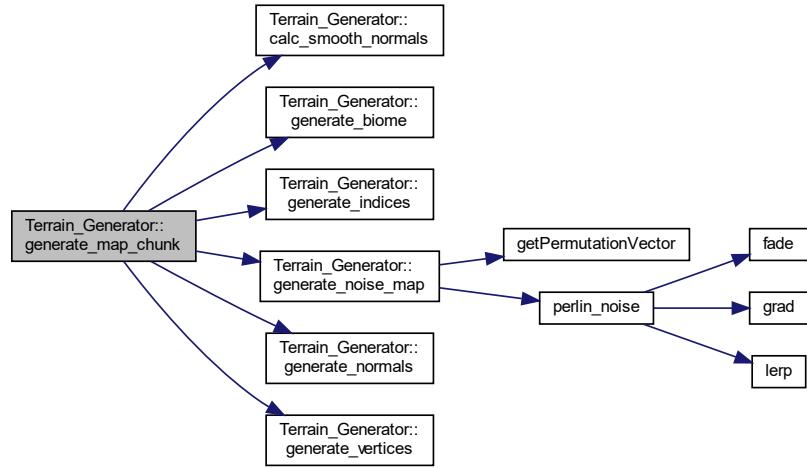
Here is the caller graph for this function:



6.41.2.6 generate_map_chunk()

```
void Terrain_Generator::generate_map_chunk (
    int xOffset,
    int yOffset,
    std::vector< std::vector< Terrain_Model > > & chunk_models )
```

Here is the call graph for this function:



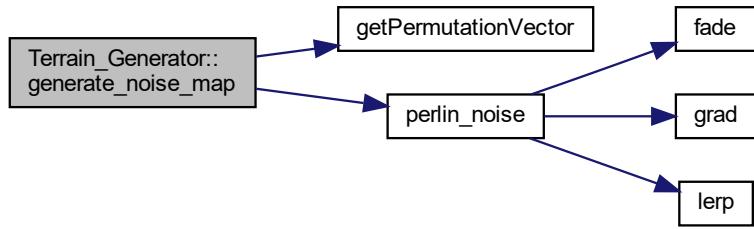
Here is the caller graph for this function:



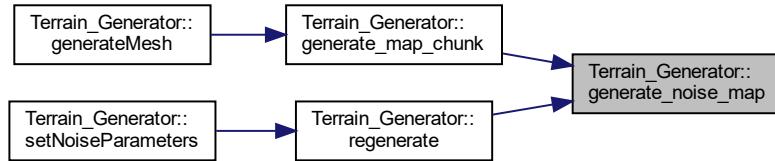
6.41.2.7 `generate_noise_map()`

```
std::vector< float > Terrain_Generator::generate_noise_map ( int xOffset, int yOffset )
```

Here is the call graph for this function:



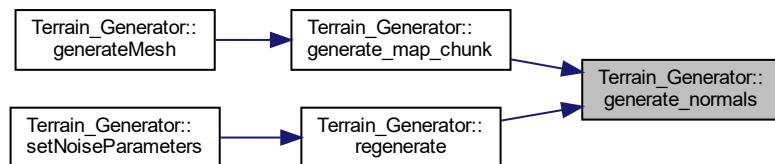
Here is the caller graph for this function:



6.41.2.8 generate_normals()

```
std::vector< glm::vec3 > Terrain_Generator::generate_normals (
    const std::vector< GLuint > & indices,
    const std::vector< glm::vec3 > & vertices )
```

Here is the caller graph for this function:



6.41.2.9 generate_render_context()

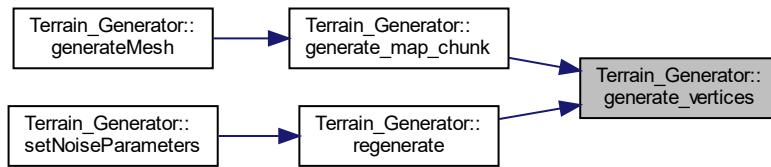
```
void Terrain_Generator::generate_render_context ( )
```

6.41.2.10 generate_vertices()

```
std::vector< glm::vec3 > Terrain_Generator::generate_vertices (
    const std::vector< float > & noise_map )

return vertices = {x0, noise0, y0, noise1, y1, x2, noise2, y2, ...}
```

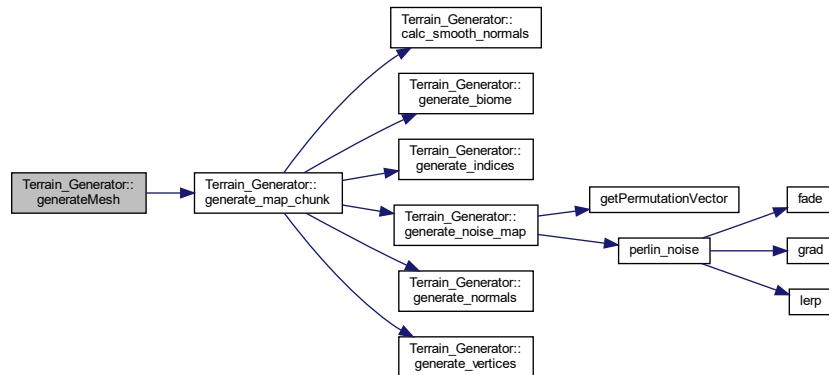
Here is the caller graph for this function:



6.41.2.11 generateMesh()

```
void Terrain_Generator::generateMesh (
    GLfloat & progress )
```

Here is the call graph for this function:



6.41.2.12 get_material_id()

```
GLuint Terrain_Generator::get_material_id ( )
```

6.41.2.13 get_normal_world_trafo()

```
glm::mat4 Terrain_Generator::get_normal_world_trafo ( )
```

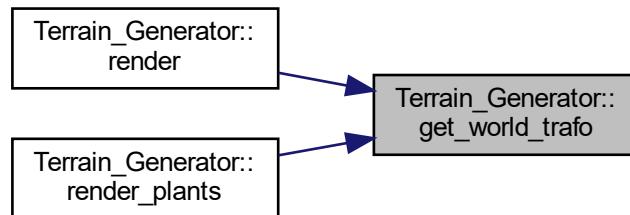
6.41.2.14 get_textures()

```
std::shared_ptr< Terrain_Texture > Terrain_Generator::get_textures ( )
```

6.41.2.15 get_world_trafo()

```
glm::mat4 Terrain_Generator::get_world_trafo ( )
```

Here is the caller graph for this function:

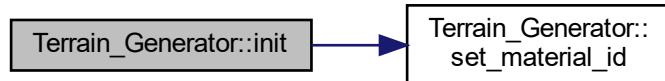
**6.41.2.16 getNoiseParameters()**

```
std::tuple< int, float, float, float, float > Terrain_Generator::getNoiseParameters ( ) [inline]
```

6.41.2.17 init()

```
void Terrain_Generator::init ( )
```

Here is the call graph for this function:



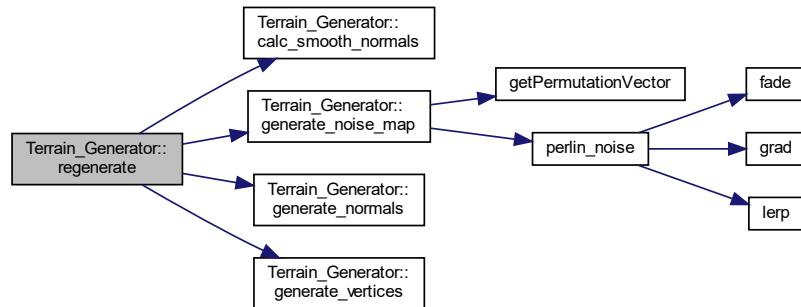
6.41.2.18 load_plants()

```
void Terrain_Generator::load_plants ( )
```

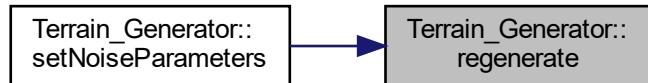
6.41.2.19 regenerate()

```
void Terrain_Generator::regenerate ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:

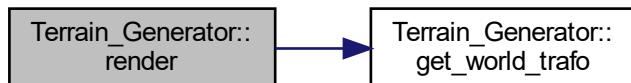


6.41.2.20 render()

```
std::vector< bool > Terrain_Generator::render (  
    GLfloat ratio,  
    std::shared_ptr< Camera > camera,  
    std::shared_ptr< ViewFrustumCulling > view_cull,  
    RenderPassSceneDependend * shader_pass )
```

dynamic render Terrain TODO paralell loading?

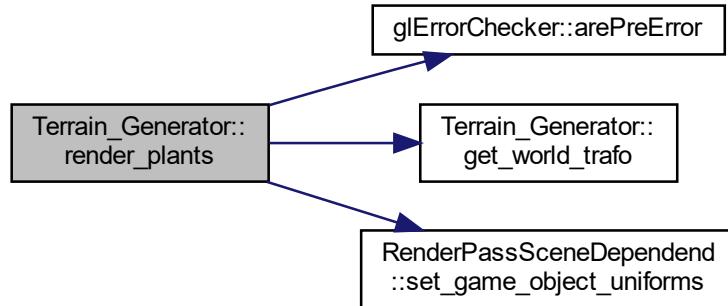
Here is the call graph for this function:



6.41.2.21 render_plants()

```
void Terrain_Generator::render_plants (   
    std::vector< bool > is_chunk_rendered_flags,  
    RenderPassSceneDependend * shader_pass )
```

Here is the call graph for this function:



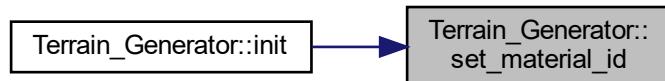
6.41.2.22 `retrive_uniform_locations()`

```
void Terrain_Generator::retrive_uniform_locations ( std::shared_ptr< GeometryPassShaderProgram > shader_program )
```

6.41.2.23 `set_material_id()`

```
void Terrain_Generator::set_material_id ( GLuint material_id )
```

Here is the caller graph for this function:



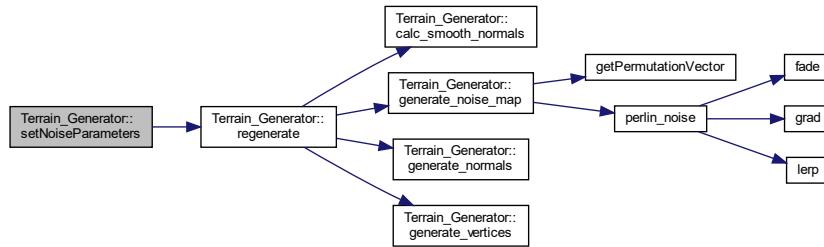
6.41.2.24 set_world_trafo()

```
void Terrain_Generator::set_world_trafo (
    glm::mat4 model_to_world )
```

6.41.2.25 setNoiseParameters()

```
void Terrain_Generator::setNoiseParameters (
    std::tuple< int, float, float, float, float > noiseParam ) [inline]
```

Here is the call graph for this function:



6.41.2.26 transform()

```
void Terrain_Generator::transform (
    glm::vec3 translate_vector,
    glm::vec3 scale )
```

6.41.3 Member Data Documentation

6.41.3.1 aabbs

```
std::vector<std::shared_ptr<AABB>> Terrain_Generator::aabbs
```

6.41.3.2 meshes

```
std::vector<std::shared_ptr<Mesh> > Terrain_Generator::meshes
```

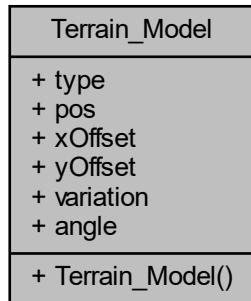
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.h
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp

6.42 Terrain_Model Class Reference

```
#include <Terrain_Model.h>
```

Collaboration diagram for Terrain_Model:



Public Types

- enum [M_TYPE](#) { [BUSH](#) , [TREE](#) , [STONE](#) }

Public Member Functions

- [Terrain_Model \(M_TYPE type, glm::vec3 pos, int xOffset, int yOffset, int variation=0\)](#)

Public Attributes

- [M_TYPE type](#)
- [glm::vec3 pos](#)
- [int xOffset](#)
- [int yOffset](#)
- [int variation](#)
- [float angle](#)

6.42.1 Member Enumeration Documentation

6.42.1.1 M_TYPE

```
enum Terrain_Model::M_TYPE
```

Enumerator

BUSH	
TREE	
STONE	

6.42.2 Constructor & Destructor Documentation

6.42.2.1 Terrain_Model()

```
Terrain_Model::Terrain_Model (
    M_TYPE type,
    glm::vec3 pos,
    int xOffset,
    int yOffset,
    int variation = 0 ) [inline]
```

6.42.3 Member Data Documentation

6.42.3.1 angle

```
float Terrain_Model::angle
```

6.42.3.2 pos

```
glm::vec3 Terrain_Model::pos
```

6.42.3.3 type

```
M_TYPE Terrain_Model::type
```

6.42.3.4 variation

```
int Terrain_Model::variation
```

6.42.3.5 xOffset

```
int Terrain_Model::xOffset
```

6.42.3.6 yOffset

```
int Terrain_Model::yOffset
```

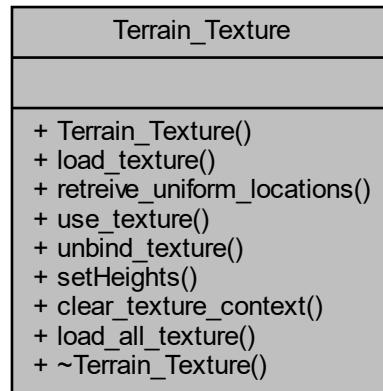
The documentation for this class was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_Model.h

6.43 Terrain_Texture Class Reference

```
#include <Terrain_texture.h>
```

Collaboration diagram for Terrain_Texture:



Public Types

- enum `BIOMETYPE` {
 `DEEP_WATER` , `SHALLOW_WATER` , `SAND` , `LIGHT_GRASS` ,
 `DARK_GRASS` , `LIGHT_ROCK` , `DARK_ROCK` , `SNOW` }

Public Member Functions

- `Terrain_Texture ()`
- `bool load_texture (char *file_location, BIOMETYPE type)`
- `void retrieve_uniform_locations (std::shared_ptr< GeometryPassShaderProgram > shader_program)`
- `void use_texture ()`
- `void unbind_texture ()`
- `void setHeights (std::vector< float > heights, float max_height)`
- `void clear_texture_context ()`
- `bool load_all_texture ()`
- `~Terrain_Texture ()`

6.43.1 Member Enumeration Documentation

6.43.1.1 BIOMETYPE

```
enum Terrain_Texture::BIOMETYPE
```

Enumerator

DEEP_WATER
SHALLOW_WATER
SAND
LIGHT_GRASS
DARK_GRASS
LIGHT_ROCK
DARK_ROCK
SNOW

6.43.2 Constructor & Destructor Documentation

6.43.2.1 Terrain_Texture()

```
Terrain_Texture::Terrain_Texture ( )
```

6.43.2.2 ~Terrain_Texture()

```
Terrain_Texture::~Terrain_Texture ( )
```

6.43.3 Member Function Documentation

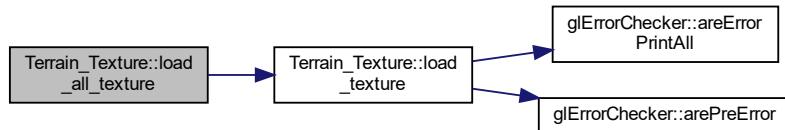
6.43.3.1 clear_texture_context()

```
void Terrain_Texture::clear_texture_context ( )
```

6.43.3.2 load_all_texture()

```
bool Terrain_Texture::load_all_texture ( )
```

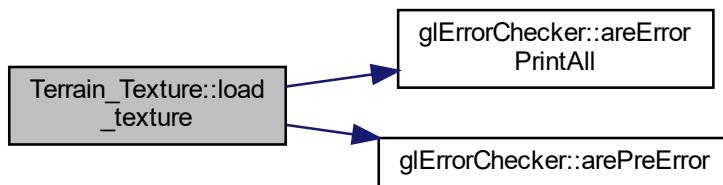
Here is the call graph for this function:



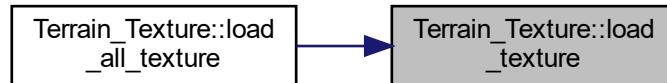
6.43.3.3 load_texture()

```
bool Terrain_Texture::load_texture (
    char * file_location,
    BIOMETYPE type )
```

Here is the call graph for this function:



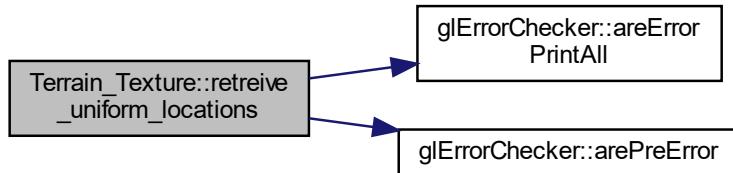
Here is the caller graph for this function:



6.43.3.4 `retreive_uniform_locations()`

```
void Terrain_Texture::retreive_uniform_locations (
    std::shared_ptr< GeometryPassShaderProgram > shader_program )
```

Here is the call graph for this function:



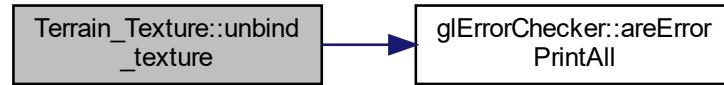
6.43.3.5 `setHeights()`

```
void Terrain_Texture::setHeights (
    std::vector< float > heights,
    float max_height )
```

6.43.3.6 unbind_texture()

```
void Terrain_Texture::unbind_texture ( )
```

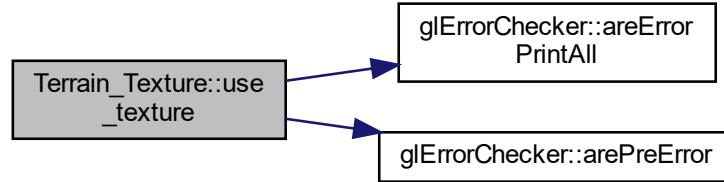
Here is the call graph for this function:



6.43.3.7 use_texture()

```
void Terrain_Texture::use_texture ( )
```

Here is the call graph for this function:

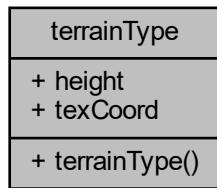


The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.h
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.cpp

6.44 terrainType Struct Reference

Collaboration diagram for terrainType:



Public Member Functions

- [terrainType](#) (float _height, glm::vec2 _texCoord)

Public Attributes

- float [height](#)
- glm::vec2 [texCoord](#)

6.44.1 Constructor & Destructor Documentation

6.44.1.1 [terrainType\(\)](#)

```
terrainType::terrainType (
    float _height,
    glm::vec2 _texCoord ) [inline]
```

6.44.2 Member Data Documentation

6.44.2.1 [height](#)

```
float terrainType::height
```

6.44.2.2 texCoord

```
glm::vec2 terrainType::texCoord
```

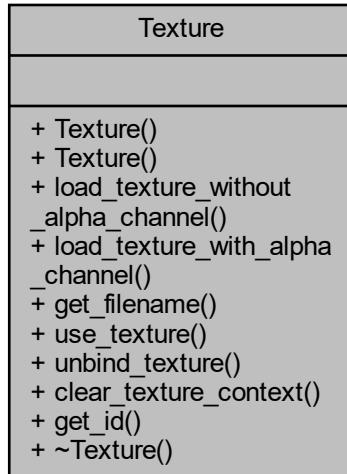
The documentation for this struct was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp

6.45 Texture Class Reference

```
#include <Texture.h>
```

Collaboration diagram for Texture:



Public Member Functions

- `Texture ()`
- `Texture (const char *file_loc, std::shared_ptr< TextureWrappingMode > wrapping_mode)`
- `bool load_texture_without_alpha_channel ()`
- `bool load_texture_with_alpha_channel ()`
- `std::string get_filename ()`
- `void use_texture ()`
- `void unbind_texture ()`
- `void clear_texture_context ()`
- `GLuint get_id ()`
- `~Texture ()`

6.45.1 Constructor & Destructor Documentation

6.45.1.1 Texture() [1/2]

```
Texture::Texture ( )
```

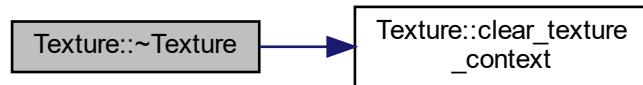
6.45.1.2 Texture() [2/2]

```
Texture::Texture (
    const char * file_loc,
    std::shared_ptr< TextureWrappingMode > wrapping_mode )
```

6.45.1.3 ~Texture()

```
Texture::~Texture ( )
```

Here is the call graph for this function:

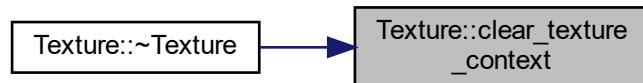


6.45.2 Member Function Documentation

6.45.2.1 clear_texture_context()

```
void Texture::clear_texture_context ( )
```

Here is the caller graph for this function:



6.45.2.2 get_filename()

```
std::string Texture::get_filename ( )
```

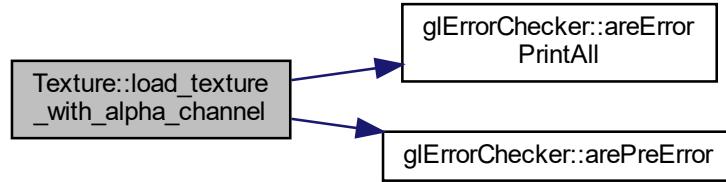
6.45.2.3 get_id()

```
GLuint Texture::get_id ( )
```

6.45.2.4 load_texture_with_alpha_channel()

```
bool Texture::load_texture_with_alpha_channel ( )
```

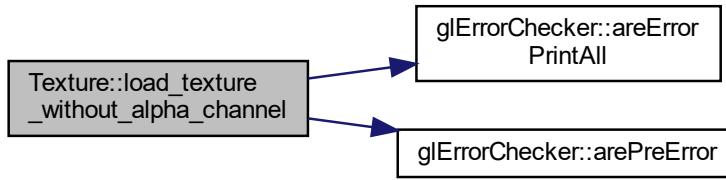
Here is the call graph for this function:



6.45.2.5 load_texture_without_alpha_channel()

```
bool Texture::load_texture_without_alpha_channel ( )
```

Here is the call graph for this function:



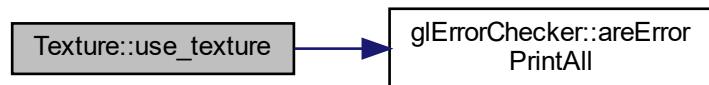
6.45.2.6 unbind_texture()

```
void Texture::unbind_texture ( )
```

6.45.2.7 use_texture()

```
void Texture::use_texture ( )
```

Here is the call graph for this function:



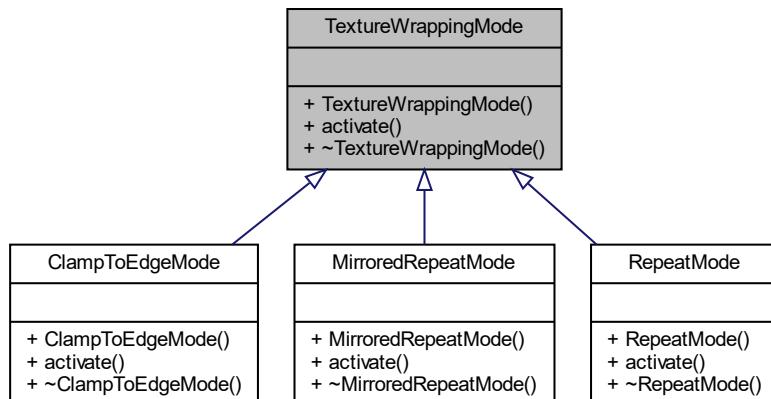
The documentation for this class was generated from the following files:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.h
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.cpp

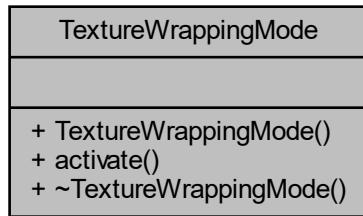
6.46 TextureWrappingMode Class Reference

```
#include <TextureWrappingMode.h>
```

Inheritance diagram for TextureWrappingMode:



Collaboration diagram for TextureWrappingMode:



Public Member Functions

- [TextureWrappingMode \(\)](#)
- virtual void [activate \(\)=0](#)
- [~TextureWrappingMode \(\)](#)

6.46.1 Constructor & Destructor Documentation

6.46.1.1 [TextureWrappingMode\(\)](#)

```
TextureWrappingMode::TextureWrappingMode ( )
```

6.46.1.2 [~TextureWrappingMode\(\)](#)

```
TextureWrappingMode::~TextureWrappingMode ( )
```

6.46.2 Member Function Documentation

6.46.2.1 [activate\(\)](#)

```
virtual void TextureWrappingMode::activate ( ) [pure virtual]
```

Implemented in [ClampToEdgeMode](#), [MirroredRepeatMode](#), and [RepeatMode](#).

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

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[TextureWrappingMode.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[TextureWrappingMode.cpp](#)

6.47 Vertex Struct Reference

```
#include <Vertex.h>
```

Collaboration diagram for Vertex:

Vertex
+ position + normal + texture_coords
+ Vertex() + Vertex() + get_position() + get_normal() + get_tex_coors() + operator==()

Public Member Functions

- `Vertex ()`
- `Vertex (glm::vec3 pos, glm::vec3 normal, glm::vec2 texture_coords)`
- `glm::vec3 get_position ()`
- `glm::vec3 get_normal ()`
- `glm::vec2 get_tex_coors ()`
- `bool operator== (const Vertex &other) const`

Public Attributes

- `glm::vec3 position`
- `glm::vec3 normal`
- `glm::vec2 texture_coords`

6.47.1 Constructor & Destructor Documentation

6.47.1.1 Vertex() [1/2]

```
Vertex::Vertex ( ) [inline]
```

6.47.1.2 Vertex() [2/2]

```
Vertex::Vertex (
    glm::vec3 pos,
    glm::vec3 normal,
    glm::vec2 texture_coords ) [inline]
```

6.47.2 Member Function Documentation

6.47.2.1 get_normal()

```
glm::vec3 Vertex::get_normal () [inline]
```

6.47.2.2 get_position()

```
glm::vec3 Vertex::get_position () [inline]
```

6.47.2.3 get_tex_coors()

```
glm::vec2 Vertex::get_tex_coors () [inline]
```

6.47.2.4 operator==()

```
bool Vertex::operator== (
    const Vertex & other ) const [inline]
```

6.47.3 Member Data Documentation

6.47.3.1 normal

```
glm::vec3 Vertex::normal
```

6.47.3.2 position

```
glm::vec3 Vertex::position
```

6.47.3.3 texture_coords

```
glm::vec2 Vertex::texture_coords
```

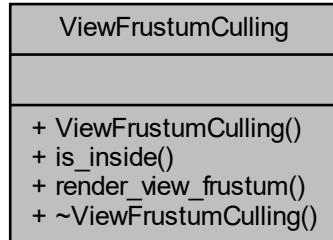
The documentation for this struct was generated from the following file:

- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[Vertex.h](#)

6.48 ViewFrustumCulling Class Reference

```
#include <ViewFrustumCulling.h>
```

Collaboration diagram for ViewFrustumCulling:



Public Member Functions

- [ViewFrustumCulling \(\)](#)
- bool [is_inside \(GLfloat ratio, std::shared_ptr< Camera > main_camera, std::shared_ptr< AABB > bounding_box, glm::mat4 model\)](#)
- void [render_view_frustum \(\)](#)
- [~ViewFrustumCulling \(\)](#)

6.48.1 Constructor & Destructor Documentation

6.48.1.1 ViewFrustumCulling()

```
ViewFrustumCulling::ViewFrustumCulling ( )
```

6.48.1.2 ~ViewFrustumCulling()

```
ViewFrustumCulling::~ViewFrustumCulling ( )
```

6.48.2 Member Function Documentation

6.48.2.1 is_inside()

```
bool ViewFrustumCulling::is_inside (
    GLfloat ratio,
    std::shared_ptr< Camera > main_camera,
    std::shared_ptr< AABB > bounding_box,
    glm::mat4 model )
```

6.48.2.2 render_view_frustum()

```
void ViewFrustumCulling::render_view_frustum ( )
```

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

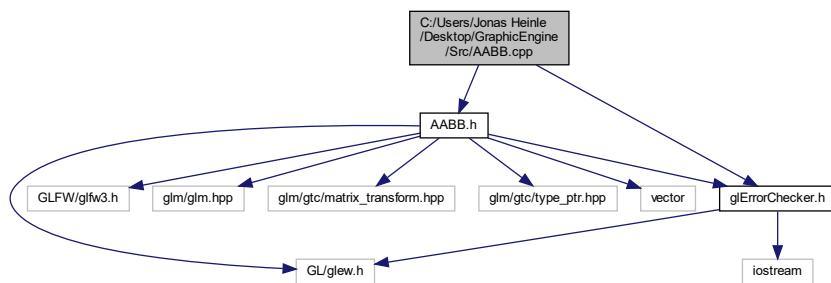
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ViewFrustumCulling.h](#)
- C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/[ViewFrustumCulling.cpp](#)

Chapter 7

File Documentation

7.1 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.cpp File Reference

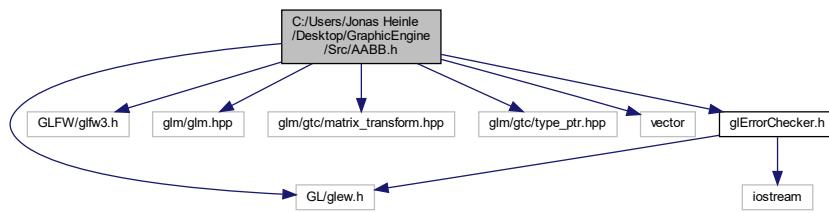
```
#include "AABB.h"
#include "glErrorChecker.h"
Include dependency graph for AABB.cpp:
```



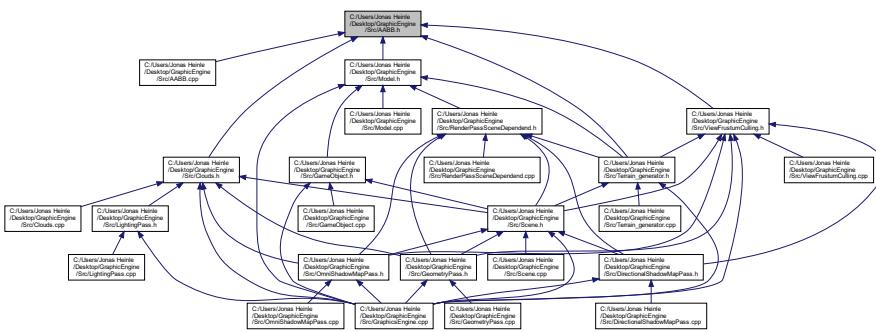
7.2 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.h File Reference

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include <vector>
```

```
#include "glErrorChecker.h"
Include dependency graph for AABB.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [AABB](#)

7.3 AABB.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <GL/glew.h>
3 #include <GLFW/glfw3.h>
4 #include <glm/glm.hpp>
5 #include <glm/gtc/matrix_transform.hpp>
6 #include <glm/gtc/type_ptr.hpp>
7 #include <vector>
8
9 #include "glErrorChecker.h"
10
11 class AABB
12 {
13 public:
14
15     AABB();
16
17     std::vector<glm::vec3> get_corners(glm::mat4 model);
18     void init(GLfloat minX, GLfloat maxX, GLfloat minY, GLfloat maxY, GLfloat minZ, GLfloat maxZ);
19     void render();
20
21     ~AABB();
22
23 private:
24 }
```

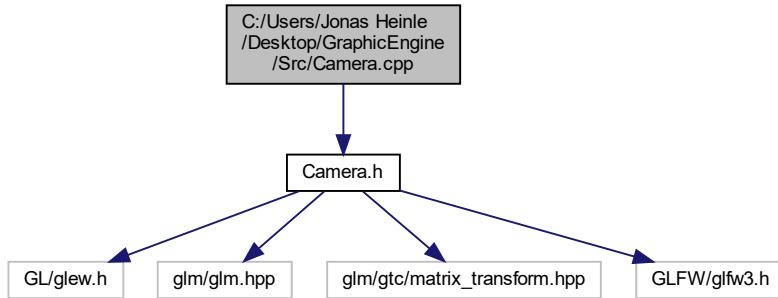
```

25     unsigned int VBO, VAO, EBO;
26
27     unsigned int m_drawCount;
28
29     std::vector<glm::vec3> corners;
30
31
32     // This to test gl operators if they went correctly.
33     glErrorChecker glErrorChecker_ins;
34
35
36     GLfloat minX, maxX, minY, maxY, minZ, maxZ;
37 }
38

```

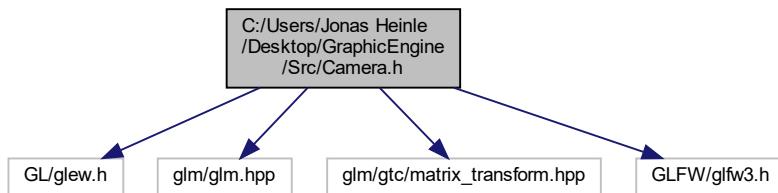
7.4 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.cpp File Reference

#include "Camera.h"
Include dependency graph for Camera.cpp:

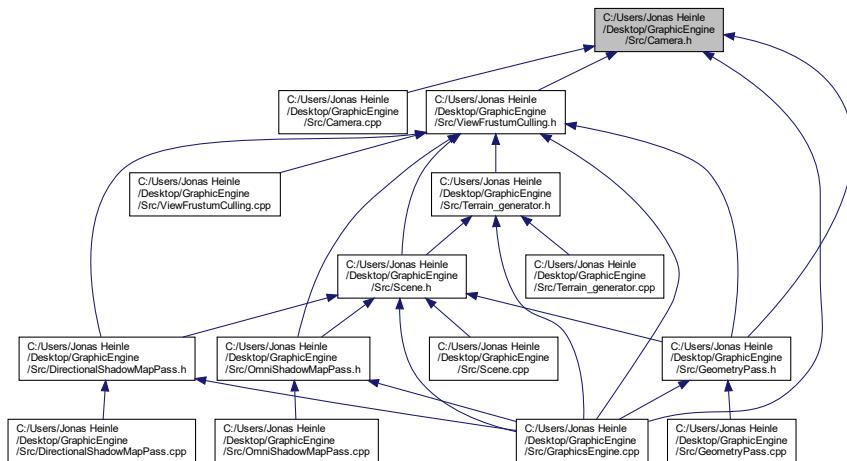


7.5 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.h File Reference

#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <GLFW/glfw3.h>
Include dependency graph for Camera.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [Camera](#)

7.6 Camera.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <GL/glew.h>
3 #include <glm/glm.hpp>
4 #include <glm/gtc/matrix_transform.hpp>
5
6 #include <GLFW/glfw3.h>
7
8 class Camera
9 {
10
11 public:
12
13     Camera();
14
15     Camera(glm::vec3 start_position, glm::vec3 start_up, GLfloat start_yaw, GLfloat start_pitch,
16             GLfloat start_move_speed, GLfloat start_turn_speed,
17             GLfloat near_plane, GLfloat far_plane, GLfloat fov);
18
19     void key_control(bool* keys, GLfloat delta_time);
20     void mouse_control(GLfloat x_change, GLfloat y_change);
21
22     glm::vec3 get_camera_position();
23     glm::vec3 get_camera_direction();
24     glm::vec3 get_up_axis();
25     glm::vec3 get_right_axis();
26     GLfloat get_near_plane();
27     GLfloat get_far_plane();
28     GLfloat get_fov();
29     GLfloat get_yaw();
30     void set_near_plane(GLfloat near_plane);
31     void set_far_plane(GLfloat far_plane);
32     void set_fov(GLfloat fov);
33
34     void set_camera_position(glm::vec3 new_camera_position);
35
36     glm::mat4 calculate_viewmatrix();
37
38     ~Camera();
39
40 private:
  
```

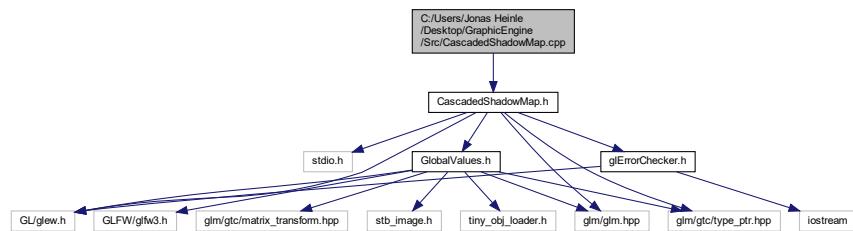
```

41     glm::vec3 position;
42     glm::vec3 front;
43     glm::vec3 up;
44     glm::vec3 right;
45     glm::vec3 world_up;
46
47
48     GLfloat yaw;
49     GLfloat pitch;
50
51     GLfloat movement_speed;
52     GLfloat turn_speed;
53
54     GLfloat near_plane, far_plane, fov;
55
56     void update();
57     float to_radians(float angle_in_degrees);
58 }
59

```

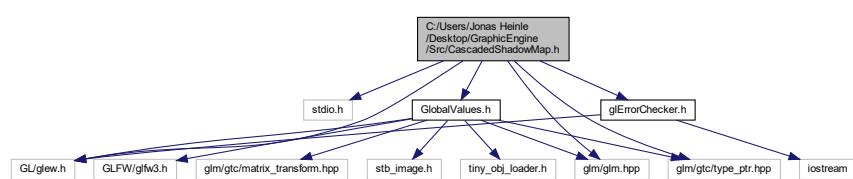
7.7 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.cpp File Reference

#include "CascadedShadowMap.h"
Include dependency graph for CascadedShadowMap.cpp:

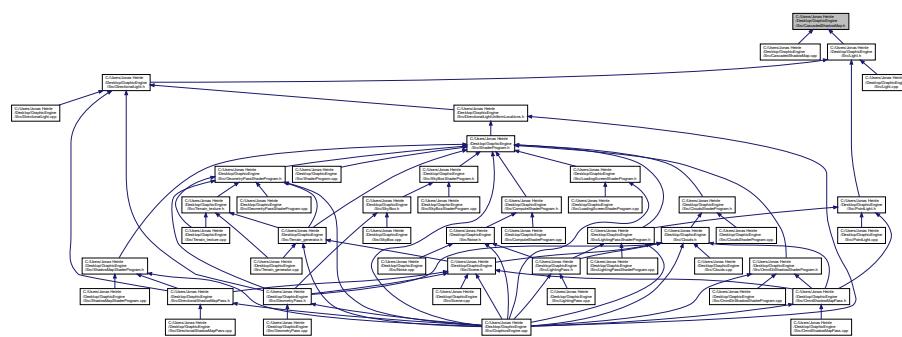


7.8 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.h File Reference

#include <stdio.h>
#include <GL/glew.h>
#include "GlobalValues.h"
#include <glm/glm.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "glErrorChecker.h"
Include dependency graph for CascadedShadowMap.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [CascadedShadowMap](#)

7.9 CascadedShadowMap.h

[Go to the documentation of this file.](#)

```

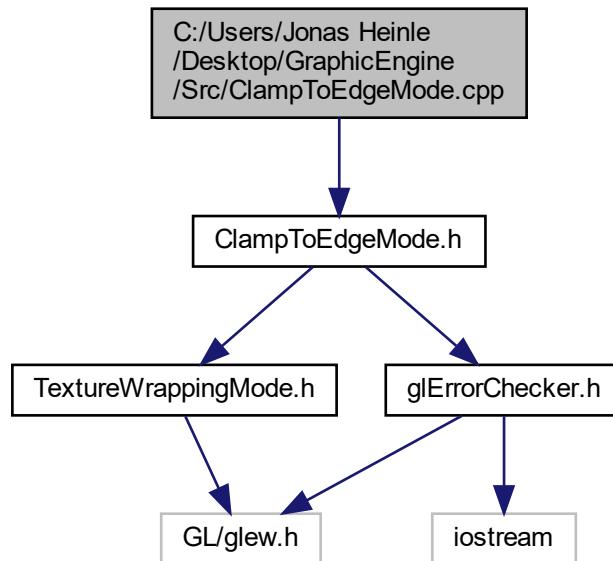
1 #pragma once
2 #include <stdio.h>
3 #include <GL/glew.h>
4 #include "GlobalValues.h"
5 #include <glm/glm.hpp>
6 #include <glm/gtc/type_ptr.hpp>
7
8 #include "glErrorChecker.h"
9
10 class CascadedShadowMap
11 {
12 public:
13
14     CascadedShadowMap();
15
16     virtual bool init(GLuint width, GLuint height, GLuint num_cascades);
17     virtual void write(GLuint cascade_index);
18     virtual void read(GLenum texture_unit);
19     void set_pcf_radius(GLuint radius);
20     void set_intensity(GLfloat intensity);
21
22     GLfloat get_intensity();
23     GLuint get_shadow_width() { return shadow_width; }
24     GLuint get_shadow_height() { return shadow_height; }
25     GLuint get_id(GLuint cascade_index);
26     GLuint get_num_active_cascades();
27     GLuint get_pcf_radius();
28
29     ~CascadedShadowMap();
30
31 protected:
32
33     GLuint FBO, shadow_map[NUM_MAX_CASCADES];
34     GLuint shadow_width, shadow_height;
35
36     GLuint num_active_cascades;
37
38     GLuint pcf_radius;
39
40     GLfloat intensity;
41
42     glErrorChecker glErrorChecker_ins;
43 };
44

```

7.10 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.cpp File Reference

```
#include "ClampToEdgeMode.h"  
Include dependency graph for ClampToEdgeMode.cpp:
```

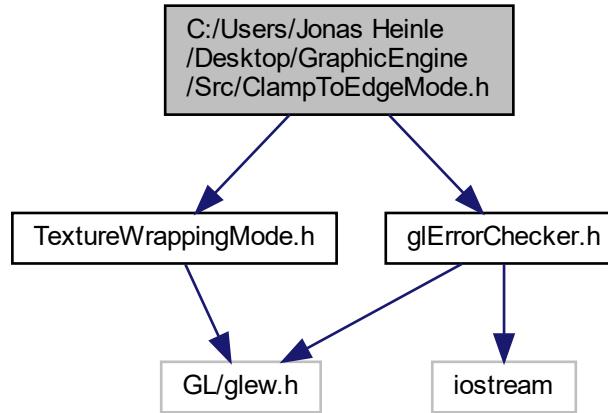


7.11 C:/Users/Jonas

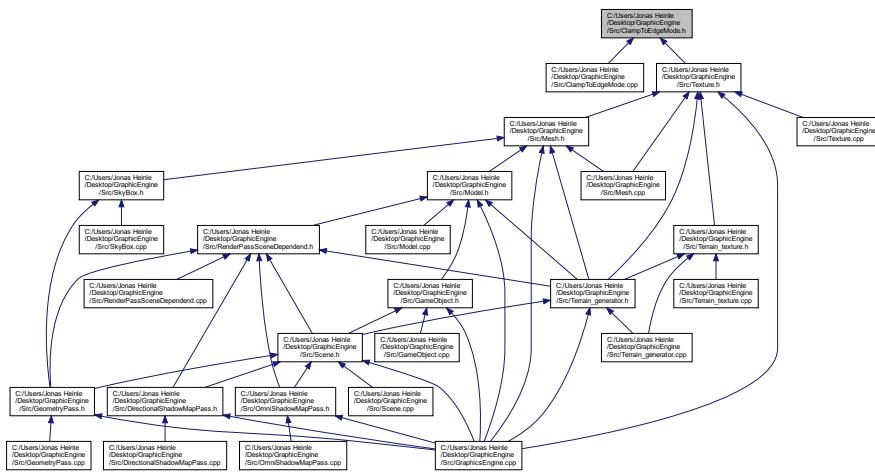
Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.h File Reference

```
#include "TextureWrappingMode.h"  
#include "glErrorChecker.h"
```

Include dependency graph for ClampToEdgeMode.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [ClampToEdgeMode](#)

7.12 ClampToEdgeMode.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "TextureWrappingMode.h"
3 #include "glErrorChecker.h"
4
  
```

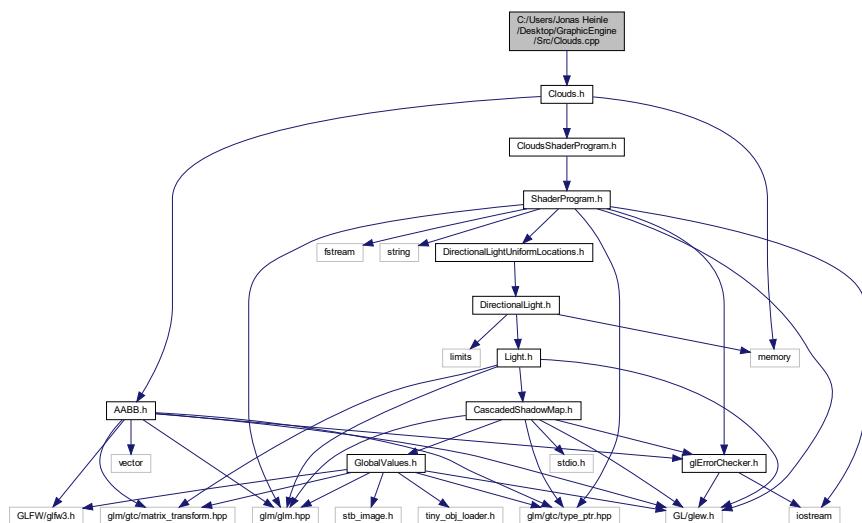
```

5 class ClampToEdgeMode :
6     public TextureWrappingMode
7 {
8 public:
9
10    ClampToEdgeMode();
11
12    void activate();
13
14    ~ClampToEdgeMode();
15
16 private:
17     gLErrorChecker glErrorChecker_ins;
18 };
19

```

7.13 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.cpp File Reference

#include "Clouds.h"
Include dependency graph for Clouds.cpp:



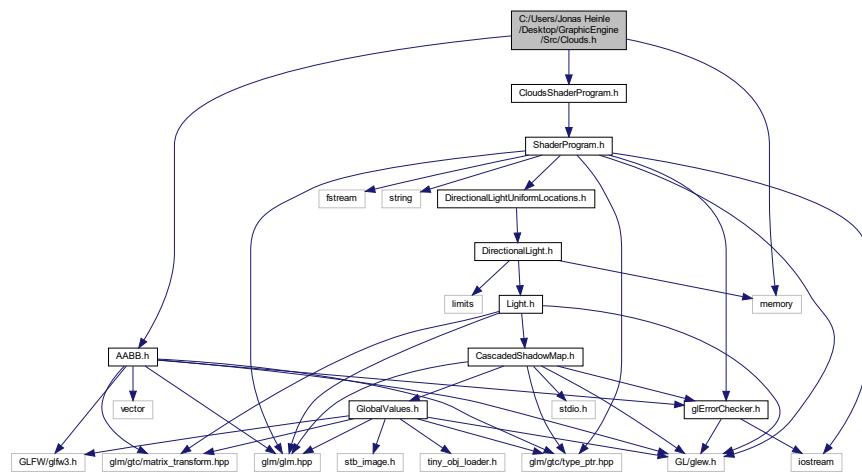
7.14 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.h File Reference

```

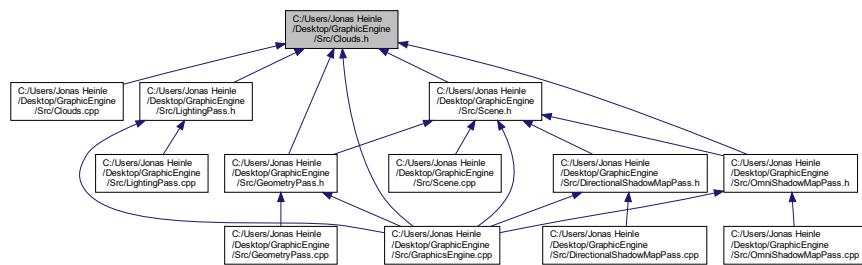
#include "AABB.h"
#include "CloudsShaderProgram.h"
#include <memory>

```

Include dependency graph for Clouds.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [Clouds](#)

7.15 Clouds.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "AABB.h"
3 #include "CloudsShaderProgram.h"
4 #include <memory>
5
6 class Clouds
7 {
8 public:
9
10     Clouds();
11
12     void init(GLfloat window_width, GLfloat window_height, GLuint movement_speed);
13     void render(glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat
window_height);
14     void read(GLuint index);
15     void update_window_params(GLfloat window_width, GLfloat window_height);
16

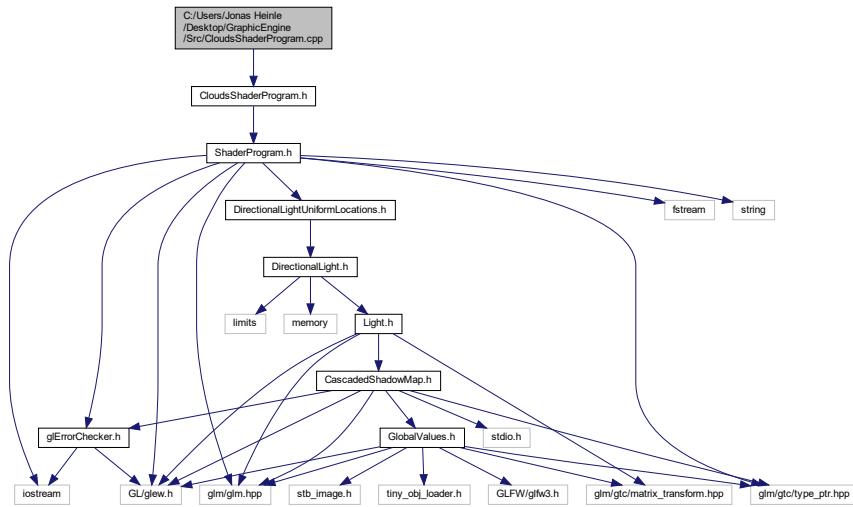
```

```
17     glm::mat4 get_model();
18     glm::vec3 get_movement_direction();
19     glm::vec3 get_rad();
20     glm::mat4 get_normal_model();
21     glm::vec3 get_mesh_scale();
22     GLfloat get_movement_speed();
23     GLfloat get_density();
24     GLfloat get_scale();
25     GLfloat get_pillowness();
26     GLfloat get_cirrus_effect();
27     bool get_powder_effect();
28
29     void set_powder_effect(bool cloud_powder_effect);
30     void set_cirrus_effect(GLfloat cirrus_effect);
31     void set_pillowness(GLfloat cloud_pillowness);
32     void set_scale(GLfloat scale);
33     void set_density(GLfloat density);
34     void set_movement_speed(GLfloat speed);
35     void set_scale(glm::vec3 scale);
36     void set_translation(glm::vec3 translation);
37     void set_movement_direction(glm::vec3 movement_dir);
38
39     std::shared_ptr<CloudsShaderProgram> get_shader_program();
40
41     ~Clouds();
42
43 private:
44
45     void retrieve_and_set_uniform_locations(glm::mat4 projection_matrix, glm::mat4 view_matrix);
46
47     GLuint FBO, cloud_id;
48
49     glm::mat4 model;
50     AABB aabb;
51     GLfloat minX, maxX, minY, maxY, minZ, maxZ;
52
53     glm::vec3 movement_direction;
54     glm::vec3 scale_factor, translation;
55
56     GLfloat movement_speed, density, scale, pillowness, cirrus_effect;
57
58     bool powder_effect;
59
60     std::shared_ptr<CloudsShaderProgram> shader_program;
61
62     GLuint attachments[1] = {GL_COLOR_ATTACHMENT0};
63 };
64
```

7.16 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.cpp File Reference

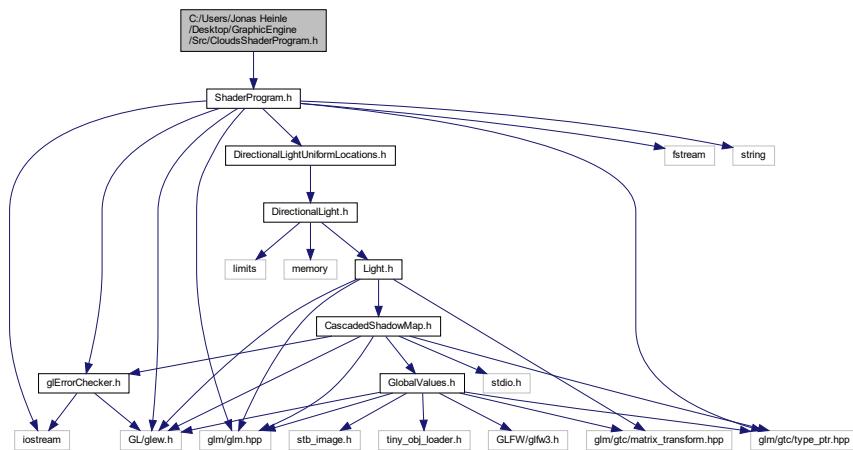
```
#include "CloudsShaderProgram.h"
Include dependency graph for CloudsShaderProgram.cpp:
```



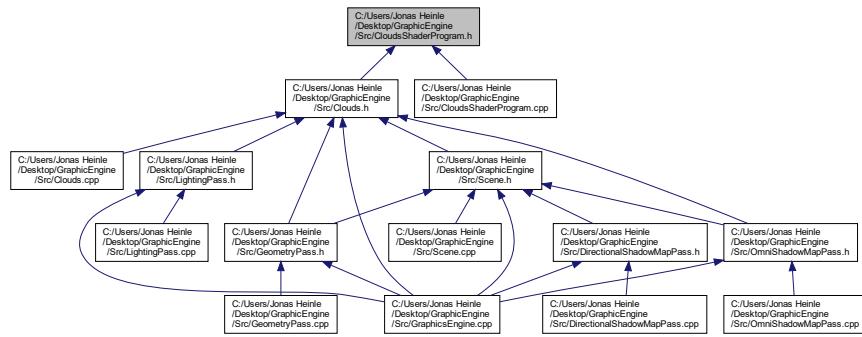
7.17 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for CloudsShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [CloudsShaderProgram](#)

7.18 CloudsShaderProgram.h

[Go to the documentation of this file.](#)

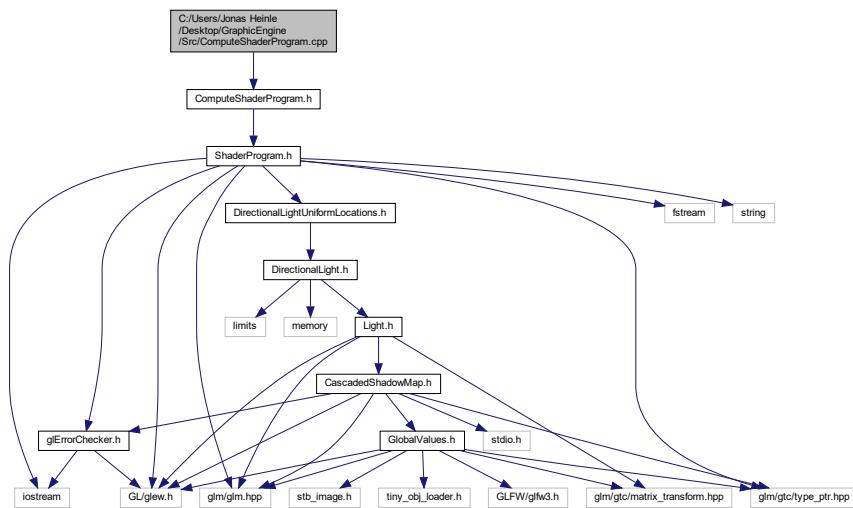
```

1 #pragma once
2 #include "ShaderProgram.h"
3 class CloudsShaderProgram :
4     public ShaderProgram
5 {
6 public:
7     CloudsShaderProgram();
8
9     GLuint get_projection_location();
10    GLuint get_view_location();
11    GLuint get_model_location();
12
13    ~CloudsShaderProgram();
14
15 protected:
16
17     GLuint uniform_model_location,
18         uniform_view_location, uniform_projection_location;
19
20     void retrieve_uniform_locations();
21
22 };
23
24
  
```

7.19 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.cpp File Reference

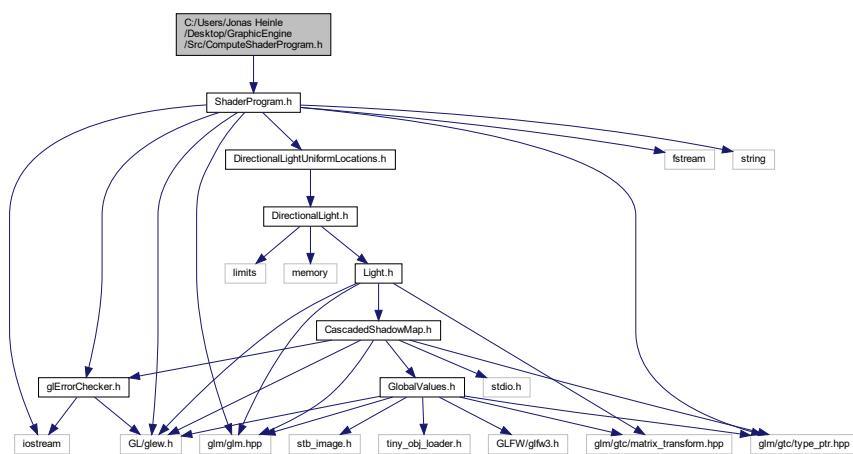
```
#include "ComputeShaderProgram.h"
Include dependency graph for ComputeShaderProgram.cpp:
```



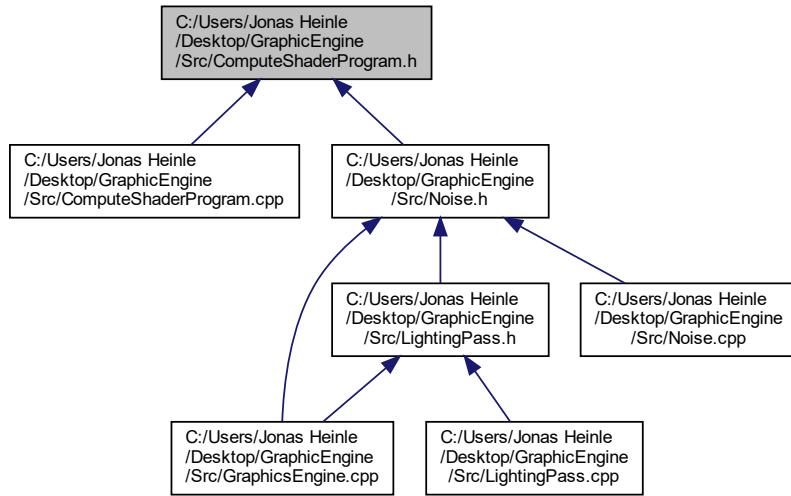
7.20 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for ComputeShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [ComputeShaderProgram](#)

7.21 ComputeShaderProgram.h

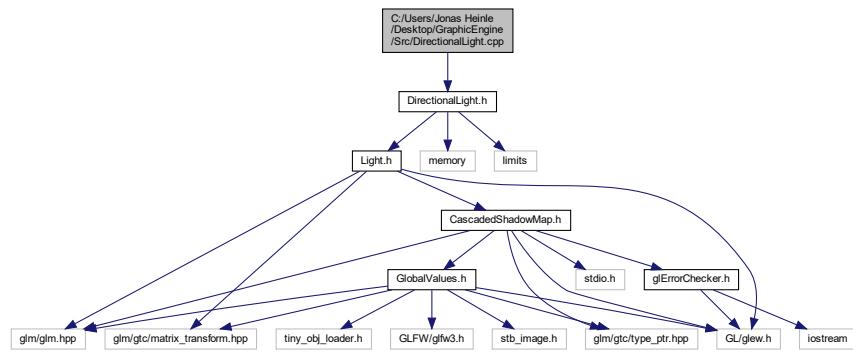
[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include "ShaderProgram.h"
3 class ComputeShaderProgram :
4     public ShaderProgram
5 {
6 public:
7
8     ComputeShaderProgram();
9
10    void set_noise(GLuint loc);
11
12    GLuint get_cell_location(GLuint index);
13    GLuint get_num_cell_location(GLuint index);
14
15    void retrieve_uniform_locations();
16    void reload();
17
18    ~ComputeShaderProgram();
19
20 private:
21
22    GLuint uniform_noise_image_location;
23    GLuint uniform_cell_locations[NUM_CELLS];
24    GLuint uniform_num_cell_locations[NUM_CELLS];
25
26 };
27
```

7.22 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/DirectionalLight.cpp File Reference

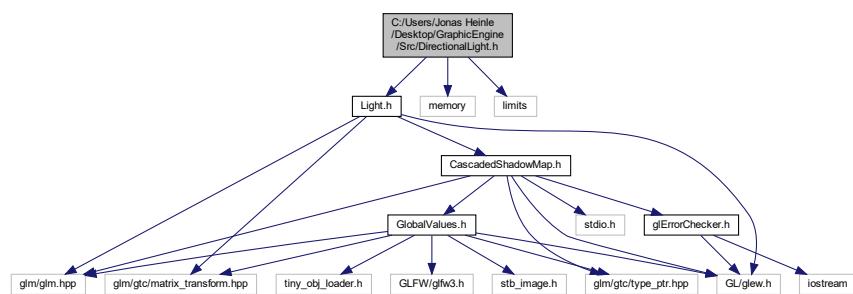
```
#include "DirectionalLight.h"
Include dependency graph for DirectionalLight.cpp:
```



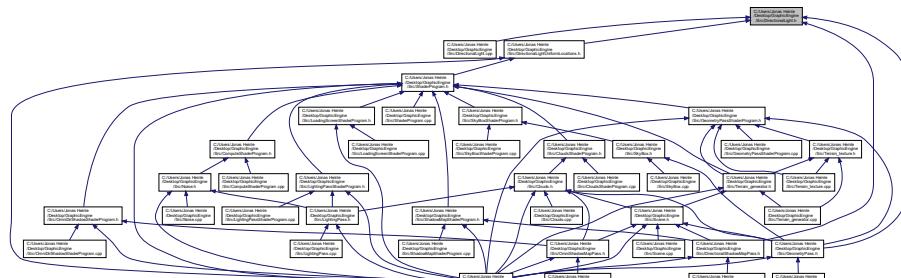
7.23 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/DirectionalLight.h File Reference

```
#include "Light.h"
#include <memory>
#include <limits>
Include dependency graph for DirectionalLight.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [DirectionalLight](#)

7.24 DirectionalLight.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "Light.h"
3 #include <memory>
4 #include <limits>
5
6 class DirectionalLight : public Light
7 {
8 public:
9
10    DirectionalLight();
11
12    DirectionalLight(GLuint shadow_width, GLuint shadow_height,
13                      GLfloat red, GLfloat green, GLfloat blue,
14                      GLfloat a_intensity, GLfloat d_intensity,
15                      GLfloat x_dir, GLfloat y_dir, GLfloat z_dir,
16                      GLfloat near_plane, GLfloat far_plane,
17                      GLfloat shadow_far_plane, int num_cascades);
18
19    glm::mat4 calculate_light_transform();
20
21    std::shared_ptr<CascadedShadowMap> get_shadow_map() { return shadow_map; }
22
23    glm::vec3 get_direction();
24    glm::vec3 get_color();
25    float get_diffuse_intensity();
26    float get_ambient_intensity();
27    glm::mat4 get_light_view_matrix();
28    std::vector<GLfloat> get_cascaded_slots();
29    std::vector<glm::mat4>& get_cascaded_light_matrices();
30
31    void update_shadow_map(GLfloat shadow_width, GLfloat shadow_height, GLuint num_cascades);
32
33    void calc_orthogonal_projections(glm::mat4 camera_view_matrix, GLfloat window_width, GLfloat
34                                     window_height,
35                                     GLfloat fov, GLuint
36                                     current_num_cascades);
37
38    void set_direction(glm::vec3 direction);
39    void set_ambient_intensity(float ambient_intensity);
40    void set_diffuse_intensity(float diffuse_intensity);
41    void set_color(glm::vec3 color);
42
43 private:
44
45    void calc_cascaded_slots();
46
47    std::shared_ptr<CascadedShadowMap> shadow_map;
48    //CascadedShadowMap* shadow_map;

```

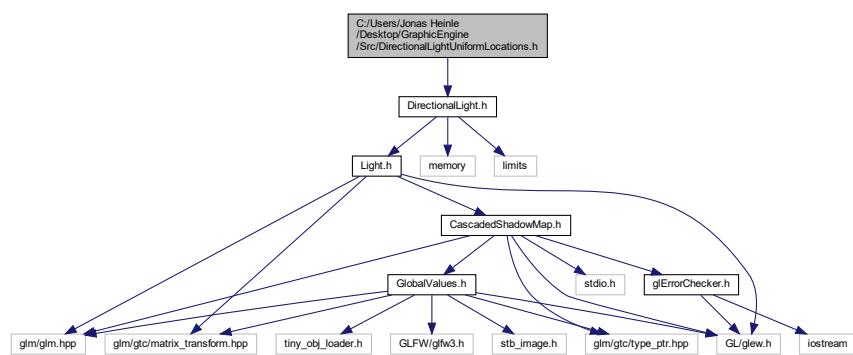
```

49     glm::vec3 direction;
50     GLfloat shadow_near_plane, shadow_far_plane;
51
52     GLfloat cascade_slots[NUM_MAX_CASCADES + 1];
53     std::vector<glm::mat4> cascade_light_matrices;
54 };
55

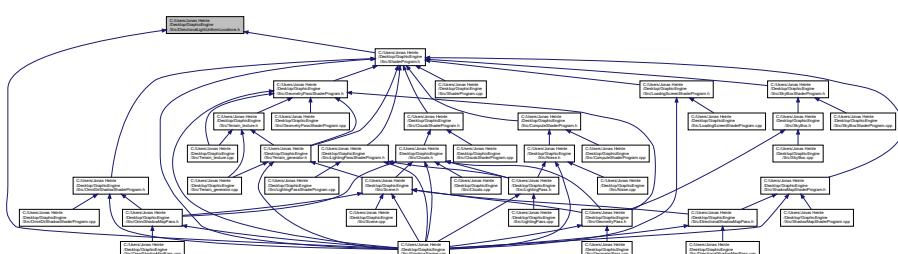
```

7.25 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLightUniformLocations.h File Reference

#include "DirectionalLight.h"
Include dependency graph for DirectionalLightUniformLocations.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [DirectionalLightUniformLocations](#)

7.26 DirectionalLightUniformLocations.h

[Go to the documentation of this file.](#)

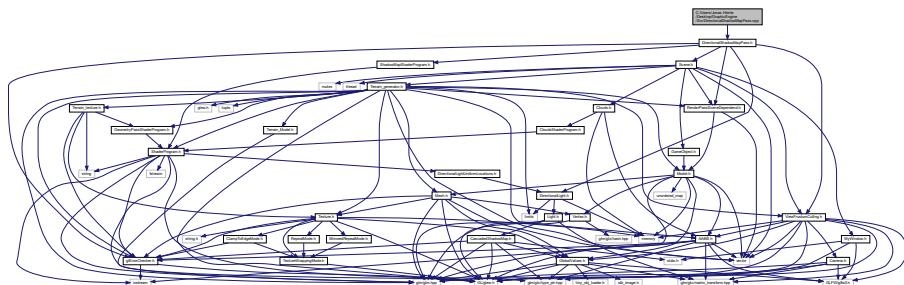
```

1 #pragma once
2 #include "DirectionalLight.h"
3
4 struct DirectionalLightUniformLocations {
5
6     GLuint uniform_color_location;
7     GLuint uniform_ambient_intensity_location;
8     GLuint uniform_diffuse_intensity_location;
9
10    GLuint uniform_direction_location;
11    GLuint uniform_shadow_intensity_location;
12 };

```

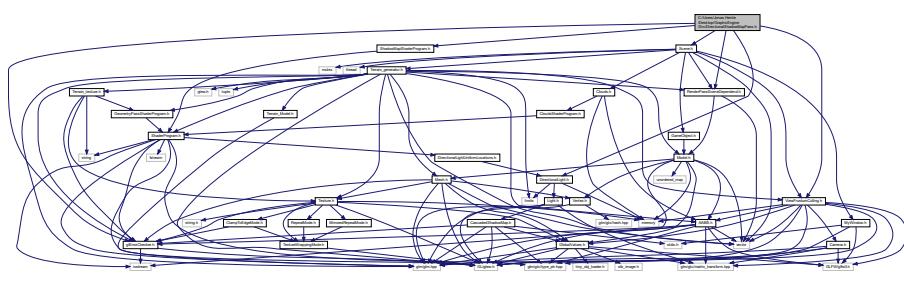
7.27 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.cpp File Reference

```
#include "DirectionalShadowMapPass.h"  
Include dependency graph for DirectionalShadowMapPass.cpp:
```

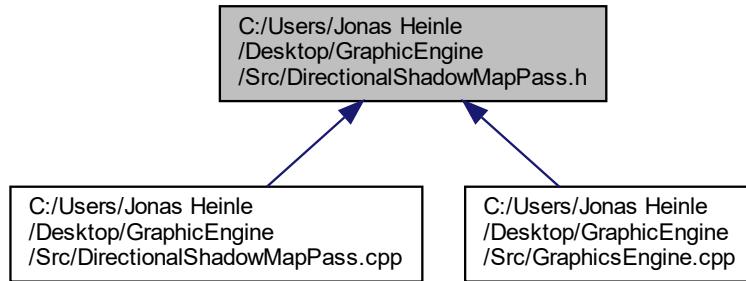


7.28 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.h File Reference

```
#include "RenderPassSceneDependend.h"  
#include "DirectionalLight.h"  
#include "ShadowMapShaderProgram.h"  
#include "ViewFrustumCulling.h"  
#include "Scene.h"  
#include "glErrorChecker.h"  
Include dependency graph for DirectionalShadowMapPass.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [DirectionalShadowMapPass](#)

7.29 DirectionalShadowMapPass.h

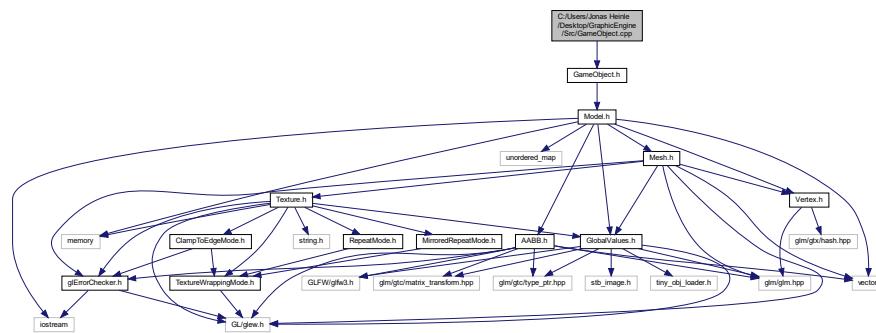
[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "RenderPassSceneDependend.h"
3 #include "DirectionalLight.h"
4 #include "ShadowMapShaderProgram.h"
5 #include "ViewFrustumCulling.h"
6 #include "Scene.h"
7
8 #include "glErrorChecker.h"
9
10 class DirectionalShadowMapPass :
11     public RenderPassSceneDependend
12 {
13 public:
14     DirectionalShadowMapPass();
15     DirectionalShadowMapPass(std::shared_ptr<ShadowMapShaderProgram> shader_program);
16
17     void execute(std::shared_ptr<DirectionalLight> d_light, glm::mat4 viewmatrix,
18                  bool first_person_mode, Scene* scene);
19
20     void set_game_object_uniforms(glm::mat4 model, glm::mat4 normal_model, GLuint material_id);
21
22     bool use_terrain_textures();
23
24     ~DirectionalShadowMapPass();
25
26 private:
27
28     std::shared_ptr<ShadowMapShaderProgram> shader_program;
29
30     glErrorChecker glErrorChecker_ins;
31 };
32
  
```

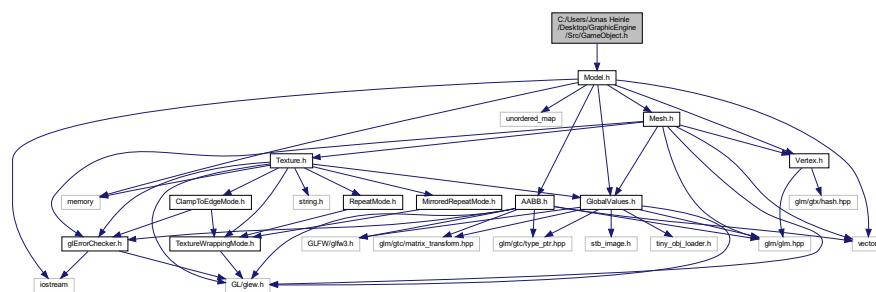
7.30 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.cpp File Reference

```
#include "GameObject.h"
Include dependency graph for GameObject.cpp:
```

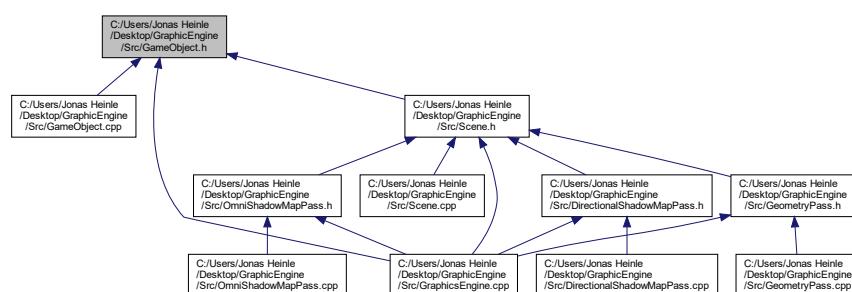


7.31 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.h File Reference

```
#include "Model.h"
Include dependency graph for GameObject.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [GameObject](#)

7.32 GameObject.h

[Go to the documentation of this file.](#)

```

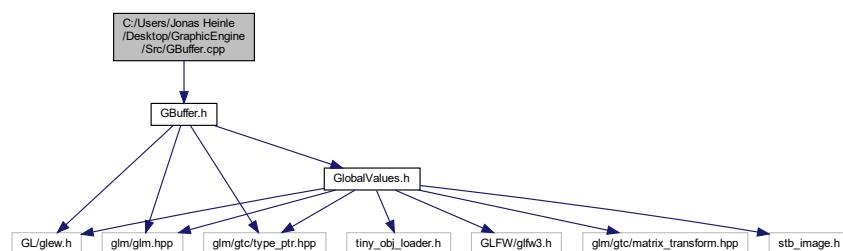
1 #pragma once
2 #include "Model.h"
3
4 class GameObject
5 {
6 public:
7     GameObject();
8
9     GameObject(std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint
10    material_id);
11
12    void init(std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot, GLuint
13    material_id);
14    void set_material_id(GLuint material_id);
15
16    glm::mat4 get_world_trafo();
17    glm::mat4 get_normal_world_trafo();
18    GLuint get_material_id();
19    std::shared_ptr<AABB> get_aabb();
20    std::shared_ptr<Model> get_model();
21
22    void translate(glm::vec3 translate);
23    void scale(GLfloat scale_factor);
24    void rotate(Rotation rot);
25
26    void render();
27
28    ~GameObject();
29
30
31 private:
32
33    GLuint material_id;
34    std::shared_ptr<Model> model;
35
36    GLfloat scale_factor;
37    Rotation rot;
38    glm::vec3 translation;
39
40 };
41

```

7.33 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.cpp File Reference

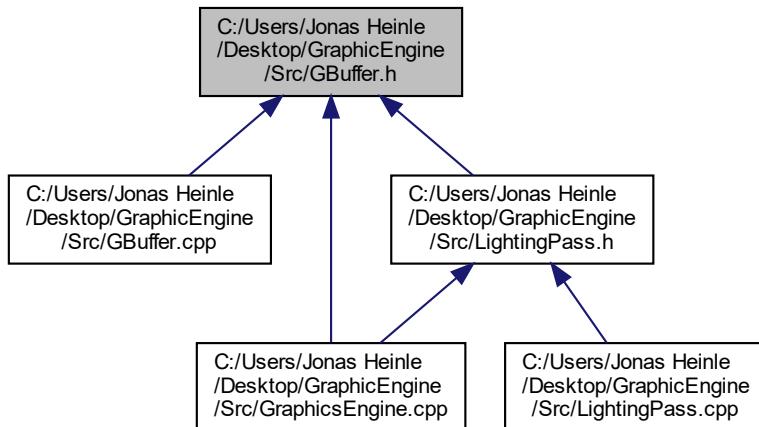
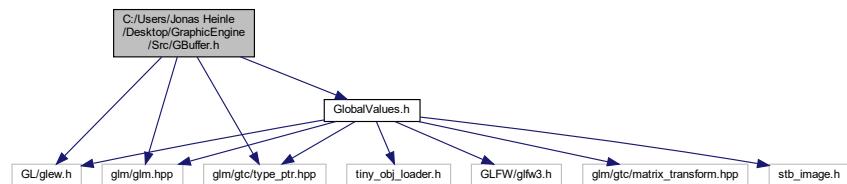
#include "GBuffer.h"

Include dependency graph for GBuffer.cpp:



7.34 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.h File Reference

```
#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "GlobalValues.h"
Include dependency graph for GBuffer.h:
```



Classes

- class [GBuffer](#)

7.35 GBuffer.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <GL/glew.h>
3 #include <glm/glm.hpp>
4 #include <glm/gtc/type_ptr.hpp>
```

```

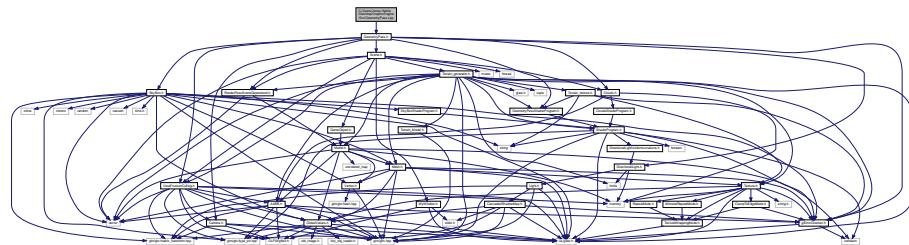
5 #include "GlobalValues.h"
6
7 class GBuffer
8 {
9 public:
10    GBuffer();
11    GBuffer(GLint window_width, GLint window_height);
12
13    void create();
14
15    void read(GLint start_buffer_index);
16
17    void use_gbuffer(GLuint g_buffer_lighting_uniform_position_location,
18                     GLuint g_buffer_lighting_uniform_normal_location,
19                     GLuint g_buffer_lighting_uniform_albedo_location,
20                     GLuint g_buffer_frag_depth_location,
21                     GLuint g_buffer_material_id_location);
22
23    void update_window_params(GLfloat window_width, GLfloat window_height);
24
25    GLint get_id();
26
27    ~GBuffer();
28
29 private:
30    GLuint g_buffer;
31
32    GLuint g_position, g_normal, g_albedo, g_depth, g_fragment_depth,
33    g_material_id;
34
35    GLuint g_buffer_attachment[G_BUFFER_SIZE] = {GL_COLOR_ATTACHMENT0, GL_COLOR_ATTACHMENT1,
36                                              GL_COLOR_ATTACHMENT2, GL_COLOR_ATTACHMENT3,
37                                              GL_COLOR_ATTACHMENT4};
38
39    GLint window_width, window_height;
40
41 };
42
43
44

```

7.36 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/GeometryPass.cpp File Reference

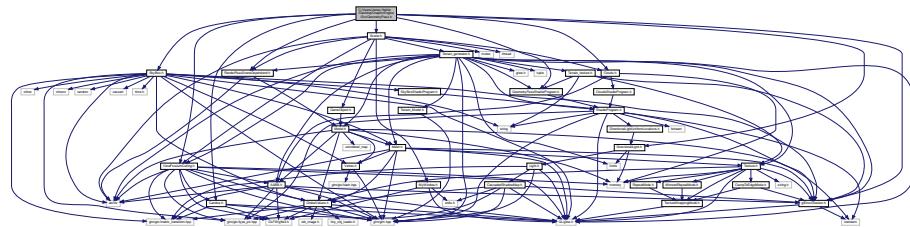
```
#include "GeometryPass.h"
Include dependency graph for GeometryPass.cpp:
```



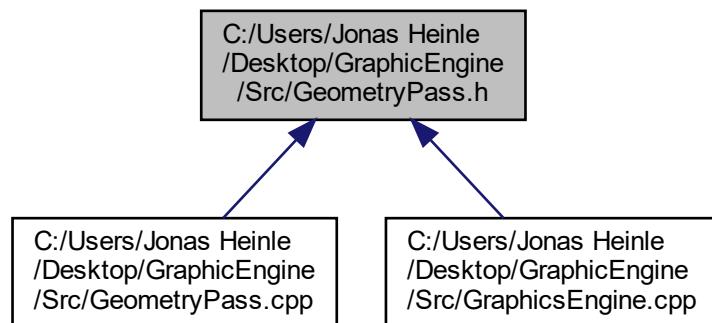
7.37 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.h File Reference

```
#include "RenderPassSceneDependend.h"
#include "DirectionalLight.h"
#include "GeometryPassShaderProgram.h"
```

```
#include "SkyBox.h"
#include "ViewFrustumCulling.h"
#include "Camera.h"
#include "Clouds.h"
#include "Scene.h"
#include "glErrorChecker.h"
Include dependency graph for GeometryPass.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [GeometryPass](#)

7.38 GeometryPass.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include "RenderPassSceneDependend.h"
3 #include "DirectionalLight.h"
4 #include "GeometryPassShaderProgram.h"
5 #include "SkyBox.h"
6 #include "ViewFrustumCulling.h"
7 #include "Camera.h"
8 #include "Clouds.h"
9 #include "Scene.h"
10 #include "SkyBox.h"
11
12 #include "glErrorChecker.h"
```

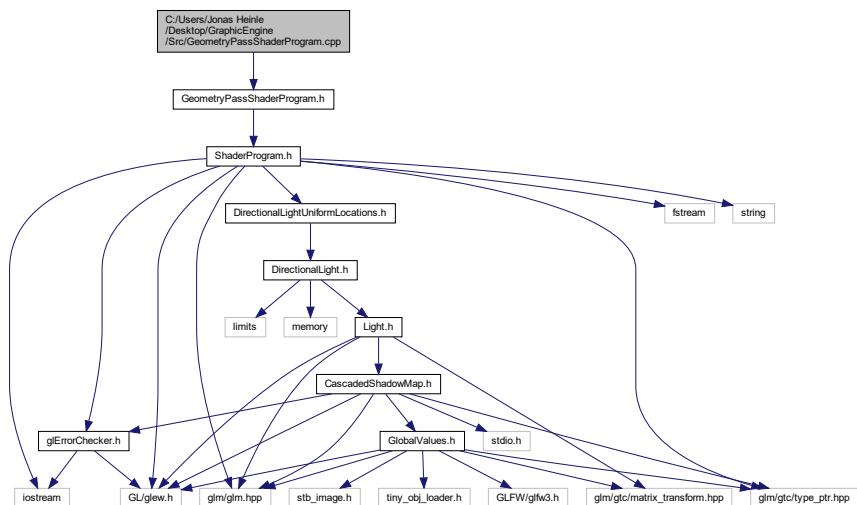
```

13
14 class GeometryPass :
15     public RenderPassSceneDependend
16 {
17 public:
18
19     GeometryPass();
20     GeometryPass(std::shared_ptr<GeometryPassShaderProgram> shader_program);
21
22     void execute(glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat
23                             window_height,
24                                         GLuint gbuffer_id, bool first_person_mode, GLfloat delta_time, Scene* scene);
25
26     void set_game_object_uniforms(glm::mat4 model, glm::mat4 normal_model, GLuint material_id);
27     bool use_terrain_textures();
28
29     ~GeometryPass();
30
31 private:
32     void retrieve_geometry_pass_locations(glm::mat4 projection_matrix, glm::mat4 view_matrix,
33                                         std::shared_ptr<Terrain_Generator> terrain_generator);
34
35     std::shared_ptr<GeometryPassShaderProgram> shader_program;
36
37     SkyBox skybox;
38
39     // TO check if there are any gl error
40     glErrorChecker glErrorChecker_ins;
41 };
42

```

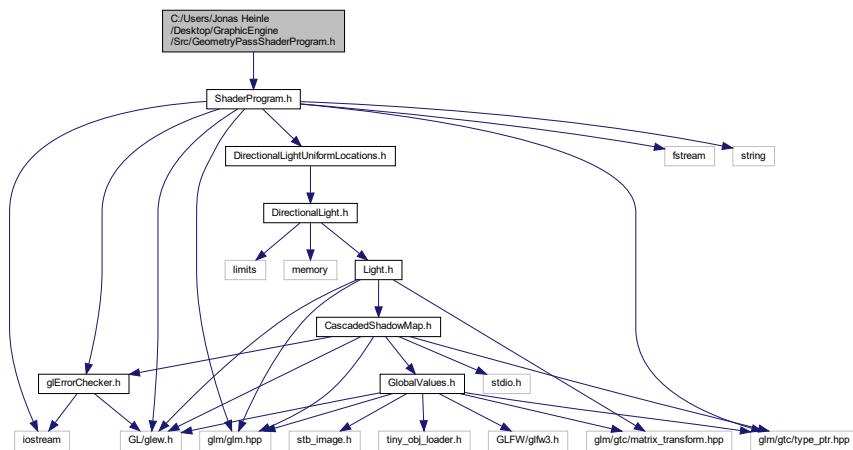
7.39 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass ShaderProgram.cpp File Reference

#include "GeometryPassShaderProgram.h"
Include dependency graph for GeometryPassShaderProgram.cpp:

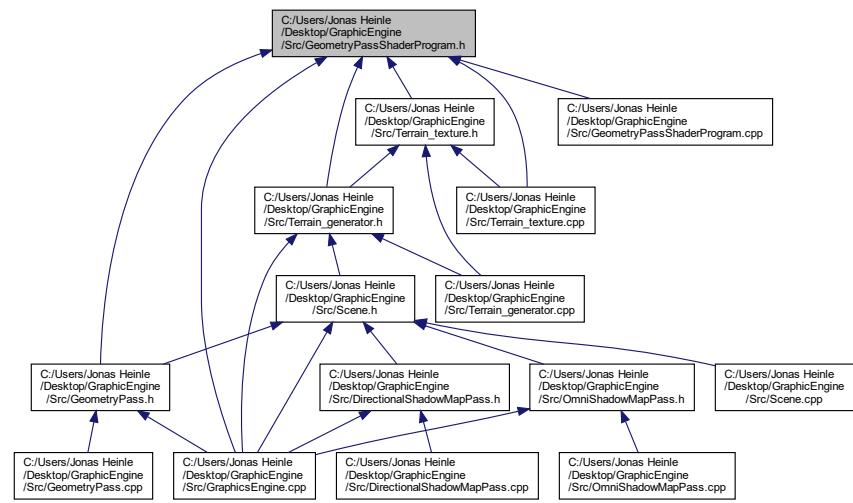


7.40 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for GeometryPassShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [GeometryPassShaderProgram](#)

7.41 GeometryPassShaderProgram.h

[Go to the documentation of this file.](#)

```

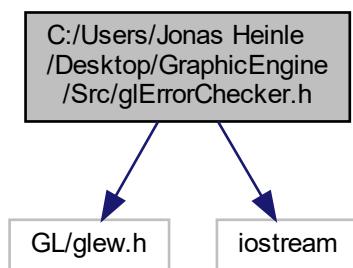
1 #pragma once
2 #include "ShaderProgram.h"
3
4 class GeometryPassShaderProgram:
5     public ShaderProgram
6 {
7
8 public:
9
10    GeometryPassShaderProgram();
11
12    GLuint get_projection_location();
13    GLuint get_view_location();
14    GLuint get_model_location();
15    GLuint get_normal_modal_location();
16    GLuint get_material_id_location();
17    GLuint get_biom_texture_location(GLuint id);
18    GLuint get_terrain_id();
19    GLuint get_biom_height_id();
20    GLuint get_max_height_id();
21
22    // change this by kansei is it okay?
23    GLuint get_program_id() { return program_id; }
24
25 ~GeometryPassShaderProgram();
26
27 protected:
28    // the unique id of our program(program is set of ShaderPrograms ...)
29    GLuint uniform_model_location,
30        uniform_view_location, uniform_projection_location,
31        uniform_material_id_location, uniform_normal_model_location;
32
33    GLuint uniform_biom_texture_locations[NUM_BIOM_TEXTURES];
34
35    GLuint uniform_isTerrainValue_id, uniform_biomHeight_id,
36        uniform_max_height_id;
37
38    void retrieve_uniform_locations();
39
40 };
41

```

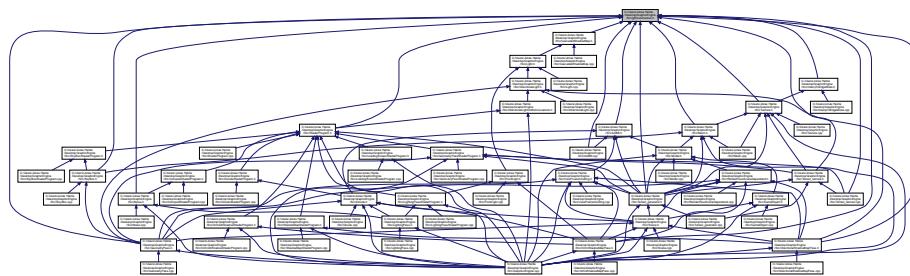
7.42 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/glErrorChecker.h File Reference

```
#include <GL/glew.h>
#include <iostream>
Include dependency graph for glErrorChecker.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [glErrorChecker](#)

7.43 glErrorChecker.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include <GL/glew.h>
4
5 #include <iostream>
6
7 class glErrorChecker {
8
9 public:
10
11 bool areErrorPrintAll(std::string AdditionalArrayMessage = "Empty") {
12     GLenum err;
13     bool isError = false;
14     while ((err = glGetError()) != GL_NO_ERROR)
15     {
16         std::string errorCode;
17         switch (err)
18         {
19             case GL_INVALID_ENUM: errorCode = "INVALID_ENUM"; break;
20             case GL_INVALID_VALUE: errorCode = "INVALID_VALUE"; break;
21             case GL_INVALID_OPERATION: errorCode = "INVALID_OPERATION"; break;
22             case GL_STACK_OVERFLOW: errorCode = "STACK_OVERFLOW"; break;
23             case GL_STACK_UNDERFLOW: errorCode = "STACK_UNDERFLOW"; break;
24             case GL_OUT_OF_MEMORY: errorCode = "OUT_OF_MEMORY"; break;
25             case GL_INVALID_FRAMEBUFFER_OPERATION: errorCode = "INVALID_FRAMEBUFFER_OPERATION"; break;
26         }
27         std::cout << "Error, " << errorCode << " | " << "\nAdditional Error Message: " <<
28         AdditionalArrayMessage << std::endl;
29         isError = true;
30     }
31     return isError;
32 }
33
34
35
36 // use this before executing gl functions to check if there uncaptured errors
37 bool arePreError(std::string AdditionalArrayMessage = "Empty") {
38     GLenum err;
39     bool isError = false;
40     while ((err = glGetError()) != GL_NO_ERROR)
41     {
42         std::string errorCode;
43         switch (err)
44         {
45             case GL_INVALID_ENUM: errorCode = "INVALID_ENUM"; break;
46             case GL_INVALID_VALUE: errorCode = "INVALID_VALUE"; break;
47             case GL_INVALID_OPERATION: errorCode = "INVALID_OPERATION"; break;
48             case GL_STACK_OVERFLOW: errorCode = "STACK_OVERFLOW"; break;
49             case GL_STACK_UNDERFLOW: errorCode = "STACK_UNDERFLOW"; break;
50             case GL_OUT_OF_MEMORY: errorCode = "OUT_OF_MEMORY"; break;

```

```

51     case GL_INVALID_FRAMEBUFFER_OPERATION: errorCode = "INVALID_FRAMEBUFFER_OPERATION"; break;
52   }
53   std::cout << errorCode << " Error appears before executing the function, " << " | " << "\nAdditional
54   Error Message: " << AdditionalArrayMessage << std::endl;
55   isError = true;
56 }
57 return isError;
58 }
59
60 };

```

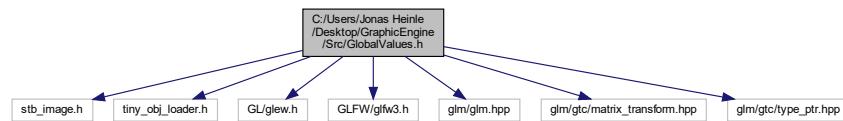
7.44 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GlobalValues.h

File Reference

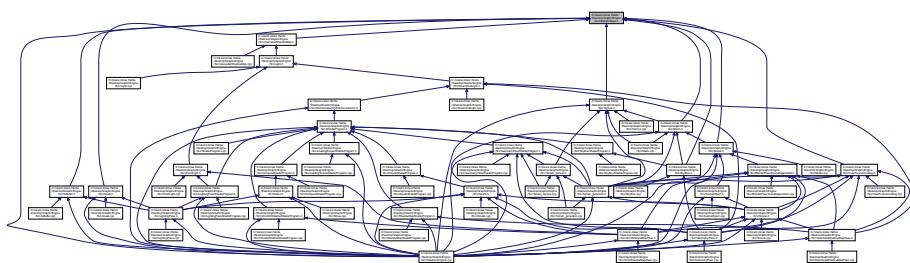
```

#include <stb_image.h>
#include <tiny_obj_loader.h>
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
Include dependency graph for GlobalValues.h:

```



This graph shows which files directly or indirectly include this file:



Classes

- struct [Rotation](#)

Macros

- #define COMMONVALS
- #define NUM_MIN_CASCADES 3
- #define NUM_MAX_CASCADES 10
- #define NUM_BIOM_TEXTURES 8
- #define MAX_RESOLUTION_X 1920
- #define MAX_RESOLUTION_Y 1080

Variables

- const int **MAX_POINT_LIGHTS** = 3
- const int **MAX_MATERIALS** = 2
- const int **G_BUFFER_SIZE** = 5
- const int **NUM_FRUSTUM_PLANES** = 6
- const int **NUM_NOISE_TEXTURES** = 2
- const int **NUM_CELLS** = 5
- const int **NUM_CLOUDS** = 1

7.44.1 Macro Definition Documentation

7.44.1.1 COMMONVALS

```
#define COMMONVALS
```

7.44.1.2 MAX_RESOLUTION_X

```
#define MAX_RESOLUTION_X 1920
```

7.44.1.3 MAX_RESOLUTION_Y

```
#define MAX_RESOLUTION_Y 1080
```

7.44.1.4 NUM BIOM_TEXTURES

```
#define NUM_BIOM_TEXTURES 8
```

7.44.1.5 NUM_MAX CASCADES

```
#define NUM_MAX_CASCADES 10
```

7.44.1.6 NUM_MIN_CASCADES

```
#define NUM_MIN_CASCADES 3
```

7.44.2 Variable Documentation

7.44.2.1 G_BUFFER_SIZE

```
const int G_BUFFER_SIZE = 5
```

7.44.2.2 MAX_MATERIALS

```
const int MAX_MATERIALS = 2
```

7.44.2.3 MAX_POINT_LIGHTS

```
const int MAX_POINT_LIGHTS = 3
```

7.44.2.4 NUM_CELLS

```
const int NUM_CELLS = 5
```

7.44.2.5 NUM_CLOUDS

```
const int NUM_CLOUDS = 1
```

7.44.2.6 NUM_FRUSTUM_PLANES

```
const int NUM_FRUSTUM_PLANES = 6
```

7.44.2.7 NUM_NOISE_TEXTURES

```
const int NUM_NOISE_TEXTURES = 2
```

7.45 GlobalValues.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2
3 #ifndef COMMONVALS
4 #define COMMONVALS
5 #include <stb_image.h>
6 #include <tiny_obj_loader.h>
7
8 #include <GL/glew.h>
9 #include <GLFW/glfw3.h>
10 #include <glm/glm.hpp>
11 #include <glm/gtc/matrix_transform.hpp>
12 #include <glm/gtc/type_ptr.hpp>
13
14 #define NUM_MIN_CASCADES 3
15 #define NUM_MAX_CASCADES 10
16 #define NUM_BIOM_TEXTURES 8
17 #define MAX_RESOLUTION_X 1920
18 #define MAX_RESOLUTION_Y 1080
19
20 const int MAX_POINT_LIGHTS = 3;
21 const int MAX_MATERIALS = 2;
22 const int G_BUFFER_SIZE = 5;
23 const int NUM_FRUSTUM_PLANES = 6;
24 const int NUM_NOISE_TEXTURES = 2;
25 const int NUM_CELLS = 5;
26 const int NUM_CLOUDS = 1;
27
28 struct Rotation {
29
30     GLfloat degrees;
31     glm::vec3 axis;
32 }
33 };
34
35 #endif
```

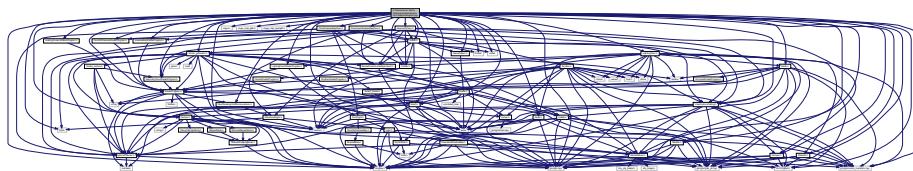
7.46 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/GraphicsEngine.cpp File Reference

```
#include <thread>
#include <mutex>
#include <memory>
#include <limits>
#include <vector>
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include <imgui.h>
#include <imgui_impl_glfw.h>
#include <imgui_impl_opengl3.h>
#include "ShaderProgram.h"
#include "GeometryPassShaderProgram.h"
#include "LightingPassShaderProgram.h"
```

```
#include "ShadowMapShaderProgram.h"
#include "OmniDirShadowShaderProgram.h"
#include "DirectionalLightUniformLocations.h"
#include "LoadingScreenShaderProgram.h"
#include "DirectionalShadowMapPass.h"
#include "OmniShadowMapPass.h"
#include "LightingPass.h"
#include "GeometryPass.h"
#include "GBuffer.h"
#include "Quad.h"
#include "Noise.h"
#include "Clouds.h"
#include "Scene.h"
#include "GameObject.h"
#include "Texture.h"
#include "Material.h"
#include "Mesh.h"
#include "Model.h"
#include "Terrain_generator.h"
#include "ViewFrustumCulling.h"
#include "MyWindow.h"
#include "Camera.h"
#include "GlobalValues.h"
```

Include dependency graph for GraphicsEngine.cpp:



Macros

- #define STB_IMAGE_IMPLEMENTATION
- #define TINYOBJLOADER_IMPLEMENTATION

Functions

- std::vector< std::shared_ptr< PointLight > > point_lights (MAX_POINT_LIGHTS, NULL)
- std::vector< std::shared_ptr< Material > > materials (MAX_MATERIALS, NULL)
- void [create_geometry_pass_shader_program](#) ()
- void [create_lighting_pass_shader_program](#) ()
- void [create_shadow_map_shader_program](#) ()
- void [create_omni_shadow_map_shader_program](#) ()
- void [create_loading_screen_shader_program](#) ()
- void [create_shader_programs](#) ()
- void [reload_shader_programs](#) ()
- void [create_noise_textures](#) ()
- void [reload_noise_programs](#) ()
- int [main](#) ()

Variables

- GLfloat delta_time = 0.0f
- GLfloat last_time = 0.0f
- unsigned int point_light_count = 0
- GLfloat near_plane = 0.1f
- GLfloat far_plane = 500.f
- GLfloat far_plane_shadow = 900.f
- GLuint shadow_map_resolution = 4096
- GLfloat fov = 60.0f
- std::shared_ptr< Camera > main_camera
- std::shared_ptr< DirectionalLight > main_light
- Texture ornament1
- std::shared_ptr< GBuffer > gbuffer
- std::shared_ptr< Noise > noise
- std::shared_ptr< Clouds > clouds
- std::shared_ptr< GeometryPassShaderProgram > g_buffer_geometry_pass_shader_program
- std::shared_ptr< LightingPassShaderProgram > g_buffer_lighting_pass_shader_program
- std::shared_ptr< ShadowMapShaderProgram > shadow_map_shader_program
- std::shared_ptr< OmniDirShadowShaderProgram > omni_dir_shadow_shader_program
- std::shared_ptr< LoadingScreenShaderProgram > loading_screen_shader_program
- OmniShadowMapPass omni_shadow_map_pass
- DirectionalShadowMapPass directional_shadow_map_pass
- GeometryPass geometry_pass
- LightingPass lighting_pass
- std::shared_ptr< Terrain_Generator > tGenerator
- Quad loading_screen
- Texture loading_screen_tex
- Texture logo
- glm::vec3 directional_light_starting_position = glm::vec3(0.0f, -1.0f, 0.1f)
- glm::vec3 directional_light_starting_color = glm::vec3(1.0f)
- unsigned int material_counter = 0
- bool ssao_enabled = false
- bool ssr_enabled = false
- int cloud_speed = 6
- float cloud_scale = 0.63f
- float cloud_density = 0.667f
- float cloud_pillowness = 0.966f
- float cloud_cirrus_effect = 0.0f
- float cloud_mesh_scale [3] = { 1000.f, 20.f, 1000.f }
- bool cloud_powder_effect = false
- float cloud_movement_direction [3] = { 1.f, 1.f, 1.f }
- float sound_volume = 0.0f
- int chosen_space_ship = 1
- bool loading_screen_finished = false
- int terrain_height = 32
- int shadow_map_res_index = 3
- bool shadow_resolution_changed = false
- int num_shadow_cascades = NUM_MIN_CASCADES
- int pcf_radius = 2
- float cascaded_shadow_intensity = 0.65f
- const char * available_shadow_map_resolutions [] = { "512", "1024", "2048", "4096" }

7.46.1 Macro Definition Documentation

7.46.1.1 STB_IMAGE_IMPLEMENTATION

```
#define STB_IMAGE_IMPLEMENTATION
```

7.46.1.2 TINYOBJLOADER_IMPLEMENTATION

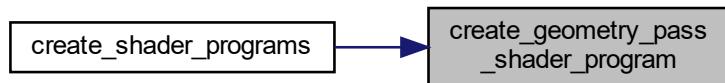
```
#define TINYOBJLOADER_IMPLEMENTATION
```

7.46.2 Function Documentation

7.46.2.1 create_geometry_pass_shader_program()

```
void create_geometry_pass_shader_program ( )
```

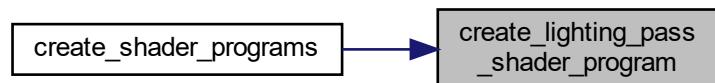
Here is the caller graph for this function:



7.46.2.2 create_lighting_pass_shader_program()

```
void create_lighting_pass_shader_program ( )
```

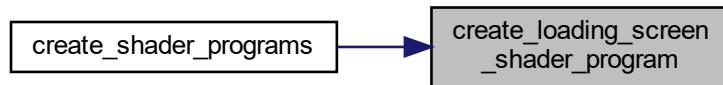
Here is the caller graph for this function:



7.46.2.3 create_loading_screen_shader_program()

```
void create_loading_screen_shader_program ( )
```

Here is the caller graph for this function:



7.46.2.4 create_noise_textures()

```
void create_noise_textures ( )
```

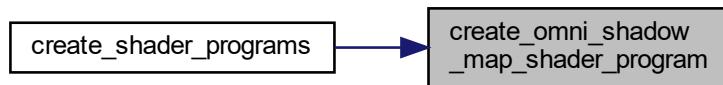
Here is the caller graph for this function:



7.46.2.5 create_omni_shadow_map_shader_program()

```
void create_omni_shadow_map_shader_program ( )
```

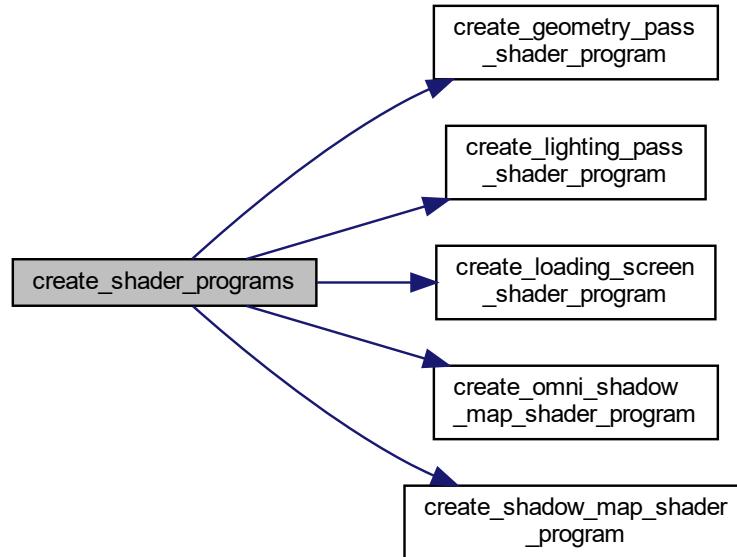
Here is the caller graph for this function:



7.46.2.6 `create_shader_programs()`

```
void create_shader_programs ( )
```

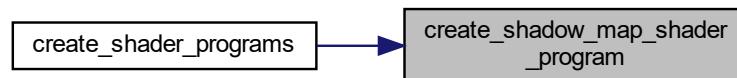
Here is the call graph for this function:



7.46.2.7 `create_shadow_map_shader_program()`

```
void create_shadow_map_shader_program ( )
```

Here is the caller graph for this function:



7.46.2.8 main()

```
int main ( )
```

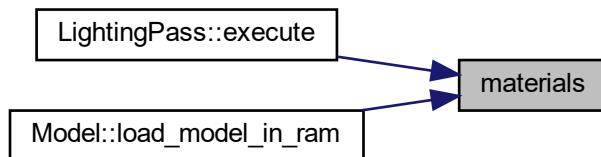
Here is the call graph for this function:



7.46.2.9 materials()

```
std::vector< std::shared_ptr< Material > > materials (
    MAX_MATERIALS ,
    NULL  )
```

Here is the caller graph for this function:



7.46.2.10 point_lights()

```
std::vector< std::shared_ptr< PointLight > > point_lights (
    MAX_POINT_LIGHTS ,
    NULL  )
```

Here is the caller graph for this function:



7.46.2.11 reload_noise_programs()

```
void reload_noise_programs ( )
```

Here is the call graph for this function:



7.46.2.12 reload_shader_programs()

```
void reload_shader_programs ( )
```

7.46.3 Variable Documentation

7.46.3.1 available_shadow_map_resolutions

```
const char* available_shadow_map_resolutions[] = { "512", "1024", "2048", "4096" }
```

7.46.3.2 cascaded_shadow_intensity

```
float cascaded_shadow_intensity = 0.65f
```

7.46.3.3 choosen_space_ship

```
int choosen_space_ship = 1
```

7.46.3.4 cloud_cirrus_effect

```
float cloud_cirrus_effect = 0.0f
```

7.46.3.5 cloud_density

```
float cloud_density = 0.667f
```

7.46.3.6 cloud_mesh_scale

```
float cloud_mesh_scale[3] = { 1000.f, 20.f, 1000.f }
```

7.46.3.7 cloud_movement_direction

```
float cloud_movement_direction[3] = { 1.f, 1.f, 1.f }
```

7.46.3.8 cloud_pillowness

```
float cloud_pillowness = 0.966f
```

7.46.3.9 cloud_powder_effect

```
bool cloud_powder_effect = false
```

7.46.3.10 cloud_scale

```
float cloud_scale = 0.63f
```

7.46.3.11 cloud_speed

```
int cloud_speed = 6
```

7.46.3.12 clouds

```
std::shared_ptr<Clouds> clouds
```

7.46.3.13 delta_time

```
GLfloat delta_time = 0.0f
```

7.46.3.14 directional_light_starting_color

```
glm::vec3 directional_light_starting_color = glm::vec3(1.0f)
```

7.46.3.15 directional_light_starting_position

```
glm::vec3 directional_light_starting_position = glm::vec3(0.0f, -1.0f, 0.1f)
```

7.46.3.16 directional_shadow_map_pass

```
DirectionalShadowMapPass directional_shadow_map_pass
```

7.46.3.17 far_plane

```
GLfloat far_plane = 500.f
```

7.46.3.18 far_plane_shadow

```
GLfloat far_plane_shadow = 900.f
```

7.46.3.19 fov

```
GLfloat fov = 60.0f
```

7.46.3.20 g_buffer_geometry_pass_shader_program

```
std::shared_ptr<GeometryPassShaderProgram> g_buffer_geometry_pass_shader_program
```

7.46.3.21 g_buffer_lighting_pass_shader_program

```
std::shared_ptr<LightingPassShaderProgram> g_buffer_lighting_pass_shader_program
```

7.46.3.22 gbuffer

```
std::shared_ptr<GBuffer> gbuffer
```

7.46.3.23 geometry_pass

```
GeometryPass geometry_pass
```

7.46.3.24 last_time

```
GLfloat last_time = 0.0f
```

7.46.3.25 lighting_pass

```
LightingPass lighting_pass
```

7.46.3.26 loading_screen

```
Quad loading_screen
```

7.46.3.27 loading_screen_finished

```
bool loading_screen_finished = false
```

7.46.3.28 loading_screen_shader_program

```
std::shared_ptr<LoadingScreenShaderProgram> loading_screen_shader_program
```

7.46.3.29 loading_screen_tex

```
Texture loading_screen_tex
```

7.46.3.30 logo

```
Texture logo
```

7.46.3.31 main_camera

```
std::shared_ptr<Camera> main_camera
```

7.46.3.32 main_light

```
std::shared_ptr<DirectionalLight> main_light
```

7.46.3.33 material_counter

```
unsigned int material_counter = 0
```

7.46.3.34 near_plane

```
GLfloat near_plane = 0.1f
```

7.46.3.35 noise

```
std::shared_ptr<Noise> noise
```

7.46.3.36 num_shadow_cascades

```
int num_shadow_cascades = NUM_MIN_CASCADES
```

7.46.3.37 omni_dir_shadow_shader_program

```
std::shared_ptr<OmniDirShadowShaderProgram> omni_dir_shadow_shader_program
```

7.46.3.38 omni_shadow_map_pass

```
OmniShadowMapPass omni_shadow_map_pass
```

7.46.3.39 ornament1

```
Texture ornament1
```

7.46.3.40 pcf_radius

```
int pcf_radius = 2
```

7.46.3.41 point_light_count

```
unsigned int point_light_count = 0
```

7.46.3.42 shadow_map_res_index

```
int shadow_map_res_index = 3
```

7.46.3.43 shadow_map_resolution

```
GLuint shadow_map_resolution = 4096
```

7.46.3.44 shadow_map_shader_program

```
std::shared_ptr<ShadowMapShaderProgram> shadow_map_shader_program
```

7.46.3.45 shadow_resolution_changed

```
bool shadow_resolution_changed = false
```

7.46.3.46 sound_volume

```
float sound_volume = 0.0f
```

7.46.3.47 ssao_enabled

```
bool ssao_enabled = false
```

7.46.3.48 ssr_enabled

```
bool ssr_enabled = false
```

7.46.3.49 terrain_height

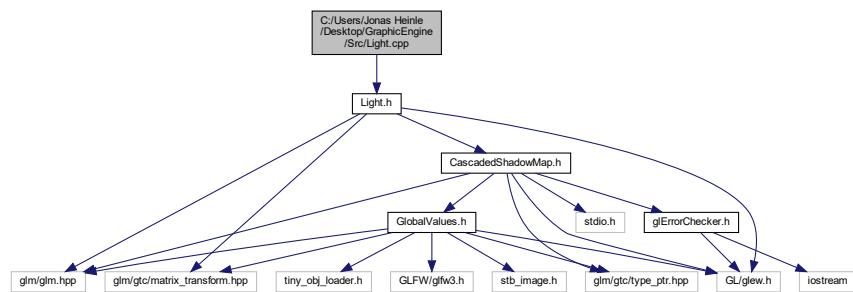
```
int terrain_height = 32
```

7.46.3.50 tGenerator

```
std::shared_ptr<Terrain_Generator> tGenerator
```

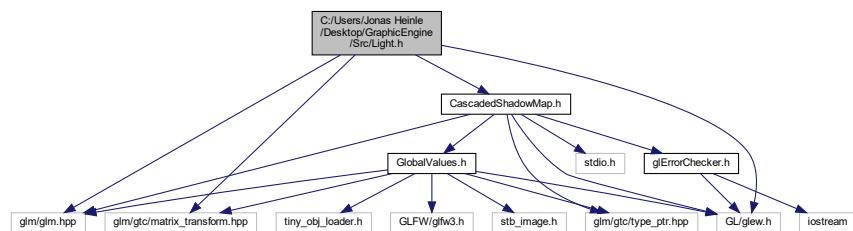
7.47 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.cpp File Reference

```
#include "Light.h"
Include dependency graph for Light.cpp:
```

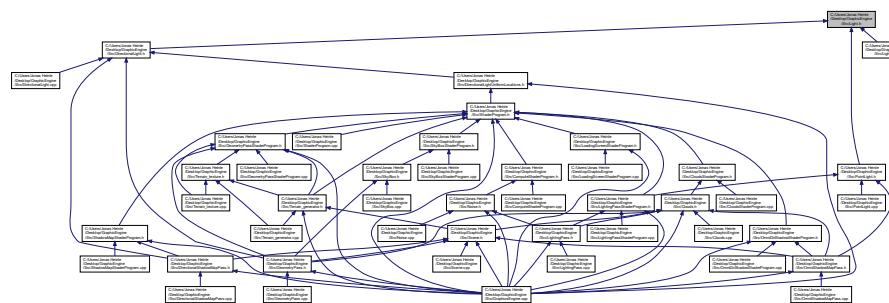


7.48 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.h File Reference

```
#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include "CascadedShadowMap.h"
Include dependency graph for Light.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Light](#)

7.49 Light.h

[Go to the documentation of this file.](#)

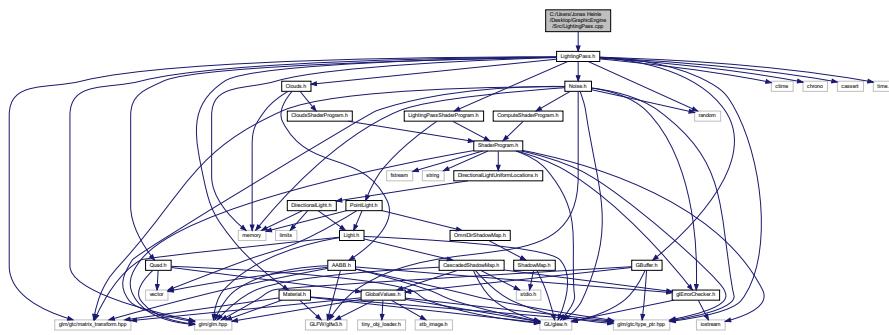
```

1 #pragma once
2 #include <GL/glew.h>
3 #include <glm/glm.hpp>
4 #include <glm/gtc/matrix_transform.hpp>
5
6 #include "CascadedShadowMap.h"
7
8 class Light
9 {
10 public:
11
12     Light();
13     Light(GLfloat shadow_width, GLfloat shadow_height,
14           GLfloat red, GLfloat green, GLfloat blue,
15           GLfloat a_intensity, GLfloat d_intensity);
16
17
18     ~Light();
19
20 protected:
21
22     glm::vec3 color;
23     float ambient_intensity;
24     float diffuse_intensity;
25
26     glm::mat4 light_proj;
27 };
28

```

7.50 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.cpp File Reference

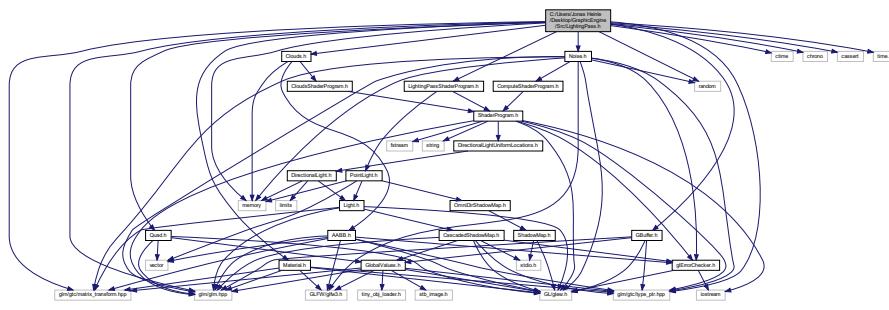
```
#include "LightingPass.h"
Include dependency graph for LightingPass.cpp:
```



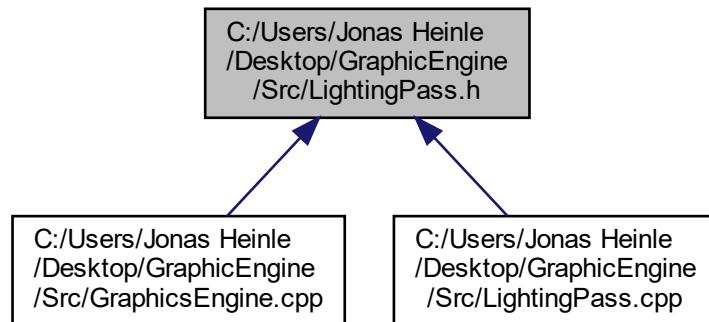
7.51 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.h File Reference

```
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "LightingPassShaderProgram.h"
#include "Quad.h"
#include "GBuffer.h"
#include "Material.h"
#include "Noise.h"
#include "Clouds.h"
#include <ctime>
#include <chrono>
#include <random>
#include <cassert>
#include <time.h>
#include <memory>
```

Include dependency graph for LightingPass.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [LightingPass](#)

7.52 LightingPass.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <glm/glm.hpp>
3 #include <glm/gtc/matrix_transform.hpp>
4 #include <glm/gtc/type_ptr.hpp>
5 #include "LightingPassShaderProgram.h"
6 #include "Quad.h"
7 #include "GBuffer.h"
8 #include "Material.h"
9 #include "Noise.h"
10 #include "Clouds.h"
11 #include "Quad.h"
12
13 #include <ctime>
14 #include <chrono>
15 #include <random>
16 #include <cassert>
17 #include <time.h>
18 #include <memory>
19
20 class LightingPass
21 {
22 public:
23
24     LightingPass();
25
26     void init(std::shared_ptr<LightingPassShaderProgram> shader_program);
27
28     void execute(glm::mat4 projection_matrix, glm::mat4 view_matrix, std::shared_ptr<GBuffer> gbuffer,
29                  std::shared_ptr<DirectionalLight> main_light,
30                  std::vector<std::shared_ptr<PointLight>>& point_lights, GLuint
31                  point_light_count, glm::vec3 camera_position, GLuint material_counter,
32                  std::vector<std::shared_ptr<Material>>& materials, std::shared_ptr<Noise>
33                  noise, std::shared_ptr<Clouds> cloud, float delta_time);
34
35     ~LightingPass();
36
37     glm::vec3 current_offset;
  
```

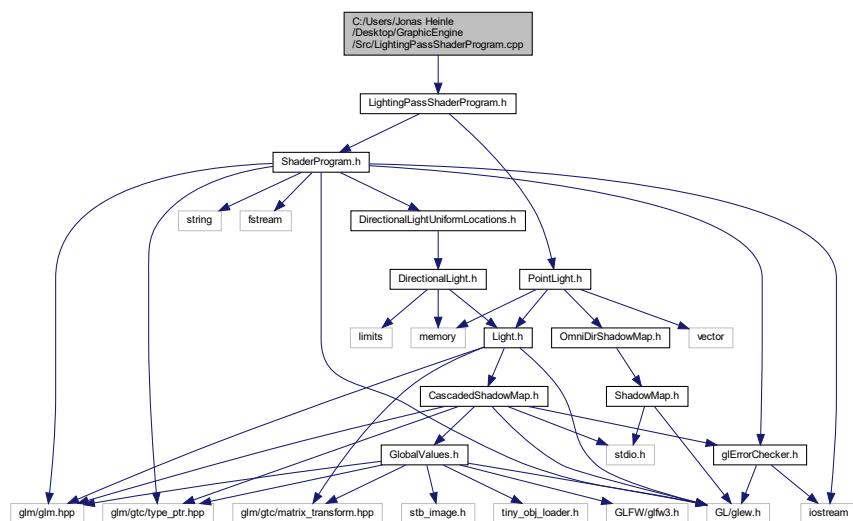
```

38     std::shared_ptr<GLfloat[]> random_number_data;
39
40     GLuint random_number;
41
42     void generate_random_numbers();
43
44     void retrieve_lighting_pass_locations(glm::mat4 projection_matrix, glm::mat4 view_matrix,
45                                         std::shared_ptr<GBuffer> gbuffer,
46                                         std::shared_ptr<DirectionalLight>
47                                         main_light,
48                                         std::vector<std::shared_ptr<PointLight>& point_lights, GLuint point_light_count, glm::vec3
49                                         camera_position, GLuint material_counter,
50                                         std::vector<std::shared_ptr<Material>& materials, std::shared_ptr<Clouds> cloud, float delta_time);
51
52     void bind_buffers_for_lighting(std::shared_ptr<GBuffer> gbuffer, std::shared_ptr<DirectionalLight>
53                                         main_light, std::shared_ptr<Noise> noise, GLuint point_light_count, std::shared_ptr<Clouds> cloud);
54
55     void bind_random_numbers(GLuint texture_unit);
56
57     std::shared_ptr<LightingPassShaderProgram> shader_program;
58
59     Quad quad;
59
59 };

```

7.53 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.cpp File Reference

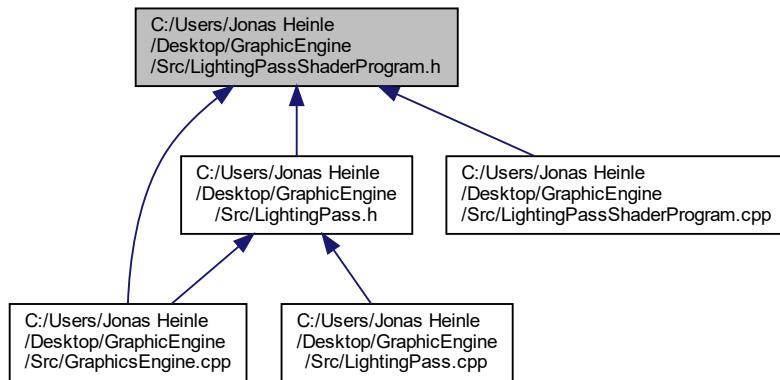
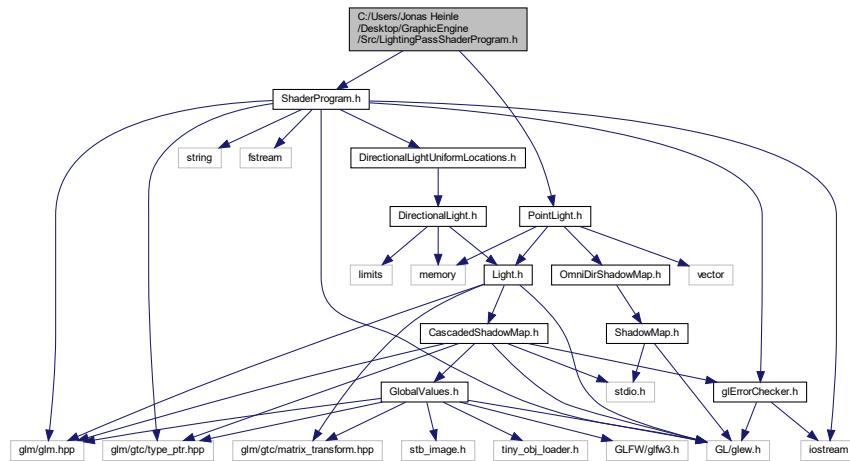
#include "LightingPassShaderProgram.h"
Include dependency graph for LightingPassShaderProgram.cpp:



7.54 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram.h File Reference

```
#include "ShaderProgram.h"
#include "PointLight.h"
```

Include dependency graph for LightingPassShaderProgram.h:



Classes

- class [LightingPassShaderProgram](#)

7.55 LightingPassShaderProgram.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "ShaderProgram.h"
3 #include "PointLight.h"
4
5 class LightingPassShaderProgram :
6     public ShaderProgram
7 {
8 public:
  
```

```

9
10    LightingPassShaderProgram();
11
12    //all the getter for the light locations
13    GLuint get_directional_light_ambient_intensity_location();
14    GLuint get_directional_light_color_location();
15    GLuint get_directional_light_diffuse_intensity_location();
16    GLuint get_directional_light_direction_location();
17    GLuint get_directional_light_shadow_intensity_location();
18    GLuint get_g_position_location();
19    GLuint get_g_normal_location();
20    GLuint get_g_albedo_location();
21    GLuint get_g_directional_light_position_location(GLuint index);
22    GLuint get_g_frag_depth_location();
23    GLuint get_directional_shadow_map_location(GLuint index);
24    GLuint get_eye_position_location();
25    GLuint get_cascade_endpoint_location(GLuint index);
26    GLuint get_uniform_omni_dir_shadow_map_location(GLuint index);
27    GLuint get_uniform_point_light_far_plane_location(GLuint index);
28    GLuint get_uniform_material_roughness_location(GLuint index);
29    GLuint get_uniform_material_metallic_location(GLuint index);
30    GLuint get_uniform_IOR_location(GLuint index);
31    GLuint get_uniform_absorption_location(GLuint index);
32    GLuint get_uniform_material_id_location();
33    GLuint get_uniform_cloud_rad_location();
34    GLuint get_uniform_cloud_offset();
35    GLuint get_uniform_cloud_model();
36    GLuint get_uniform_cloud_scale_location();
37    GLuint get_uniform_cloud_threshold_location();
38    GLuint get_directional_light_transform_location(GLuint index);
39    GLuint get_g_clouds_location();
40    GLuint get_random_number_location();
41    GLuint get_uniform_pillowness_location();
42    GLuint get_cirrus_effect_location();
43    GLuint get_cloud_powderness_effect();
44    GLuint get_uniform_num_active_cascades_location();
45    GLuint get_uniform_pcf_radius_location();
46
47    void set_point_lights(std::vector<std::shared_ptr<PointLight>>& p_light, unsigned int light_count,
48        unsigned int texture_unit, unsigned int offset);
49    void set_noise_textures(GLuint start);
50    void set_cloud_texture(GLuint index);
51
52    ~LightingPassShaderProgram();
53
54 private:
55    int point_light_counter;
56
57    GLuint uniform_g_position_location,
58        uniform_g_normal_position, uniform_g_tex_color_location,
59        uniform_eye_position_location, uniform_g_frag_depth_location,
60        uniform_material_id_location, uniform_g_clouds_location;;
61
62    GLuint uniform_directional_shadow_map_locations[NUM_MAX_CASCADES];
63    GLuint uniform_g_directional_light_position_locations[NUM_MAX_CASCADES];
64
65    GLuint uniform_cascade_endpoints_locations[NUM_MAX_CASCADES];
66
67    DirectionalLightUniformLocations d_light_uniform_locations;
68
69    GLuint uniform_point_light_count;
70
71    struct {
72
73        GLuint uniform_color;
74        GLuint uniform_ambient_intensity;
75        GLuint uniform_diffuse_intensity;
76
77        GLuint uniform_position;
78        GLuint uniform_constant;
79        GLuint uniform_linear;
80        GLuint uniform_exponent;
81
82    } uniform_point_light[MAX_POINT_LIGHTS];
83
84    struct {
85
86        GLuint uniform_shadow_map;
87        GLuint uniform_far_plane;
88
89    } uniform_omni_shadow_map[MAX_POINT_LIGHTS];
90
91    struct {
92
93        GLuint uniform_metallic_location;
94        GLuint uniform_roughness_location;

```

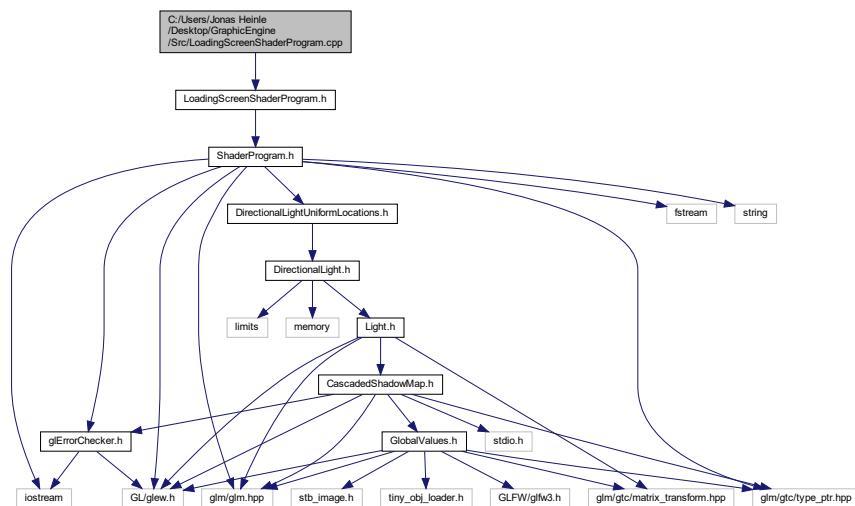
```

95     GLuint uniform_IOR_location;
96     GLuint uniform_absorption_coeff_location;
97
98 } uniform_materials[MAX_MATERIALS];
99
100 GLuint uniform_noise_texture_1_location, uniform_noise_texture_2_location;
101
102 struct uniform_cloud {
103
104     GLuint uniform_cloud_rad_location,
105     uniform_cloud_offset_location, uniform_model_to_world,
106     uniform_scale_location, uniform_threshold_location,
107     uniform_pillowness_location, uniform_cirrus_effect_location,
108     uniform_powder_effect_location;
109
110 };
111
112 uniform_cloud cloud;
113
114 GLuint uniform_cloud_texture_location;
115
116 GLuint uniform_directional_light_transform_locations[NUM_MAX_CASCADES];
117
118 GLuint uniform_random_number_location;
119
120 GLuint uniform_num_active_cascades, uniform_pcf_radius_location;
121
122 void retrieve_uniform_locations();
123 };
124

```

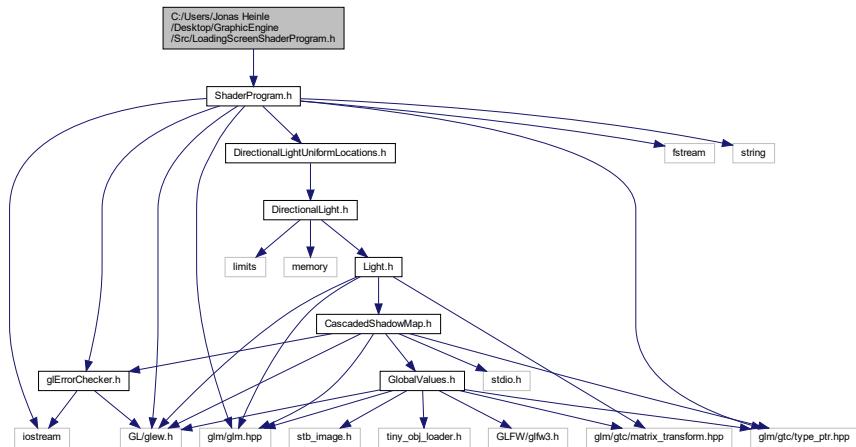
7.56 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.cpp File Reference

#include "LoadingScreenShaderProgram.h"
Include dependency graph for LoadingScreenShaderProgram.cpp:

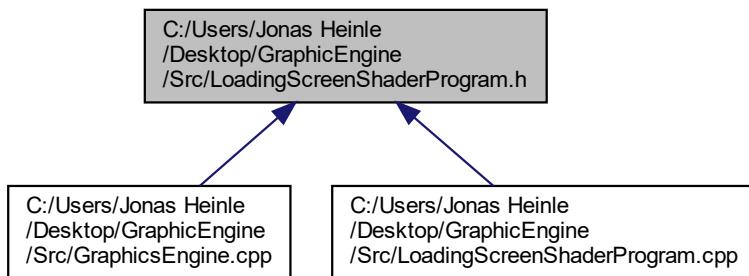


7.57 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for LoadingScreenShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [LoadingScreenShaderProgram](#)

7.58 LoadingScreenShaderProgram.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include "ShaderProgram.h"
3 class LoadingScreenShaderProgram :
4     public ShaderProgram
```

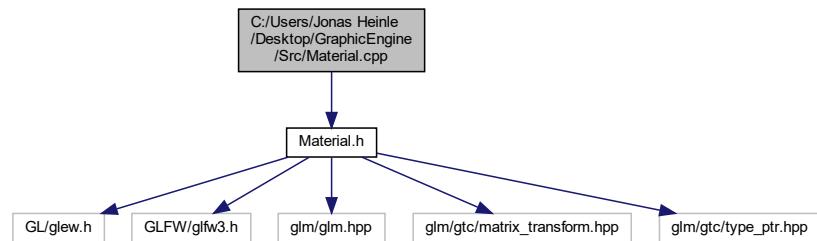
```

5 {
6 public:
7
8     LoadingScreenShaderProgram();
9
10    ~LoadingScreenShaderProgram();
11
12 private:
13
14     void retrieve_uniform_locations();
15
16 };
17

```

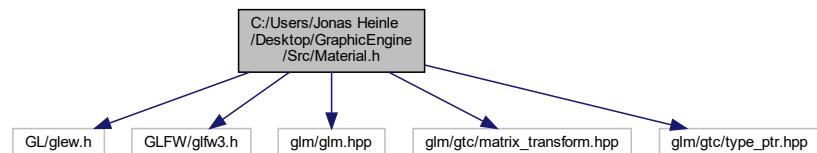
7.59 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.cpp File Reference

```
#include "Material.h"
Include dependency graph for Material.cpp:
```

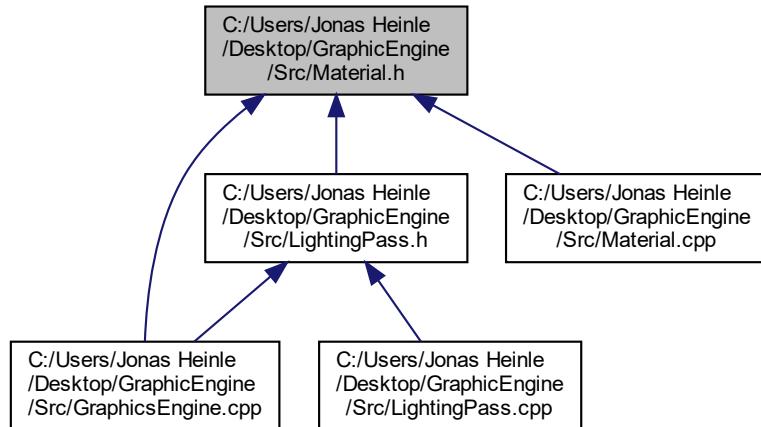


7.60 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.h File Reference

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
Include dependency graph for Material.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Material](#)

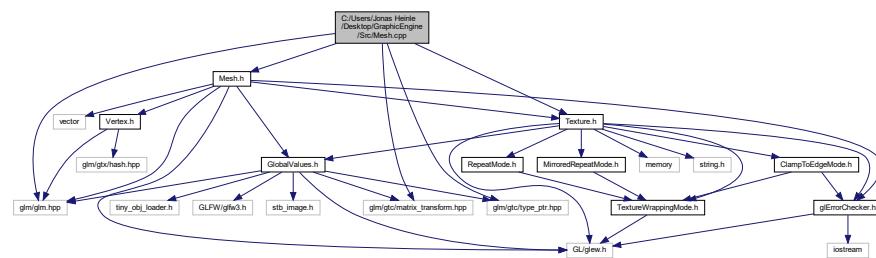
7.61 Material.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <GL/glew.h>
3 #include <GLFW/glfw3.h>
4 #include <glm/glm.hpp>
5 #include <glm/gtc/matrix_transform.hpp>
6 #include <glm/gtc/type_ptr.hpp>
7
8 class Material
9 {
10 public:
11     Material();
12     Material(GLfloat metallic, GLfloat roughness, GLfloat IOR, GLfloat absorption_coef);
13     void use_material(GLuint uniform_metallic_location, GLuint uniform_roughness_location,
14                       GLuint uniform_IOR_location, GLfloat uniform_absorption_coef_location);
15     ~Material();
16
17 private:
18     GLfloat metallic;
19     GLfloat roughness;
20     GLfloat IOR;
21     GLfloat absorption_coef;
22 };
23 }
```

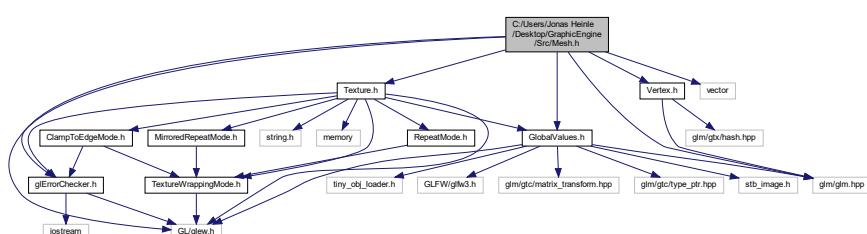
7.62 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.cpp File Reference

```
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "Mesh.h"
#include "Texture.h"
Include dependency graph for Mesh.cpp:
```

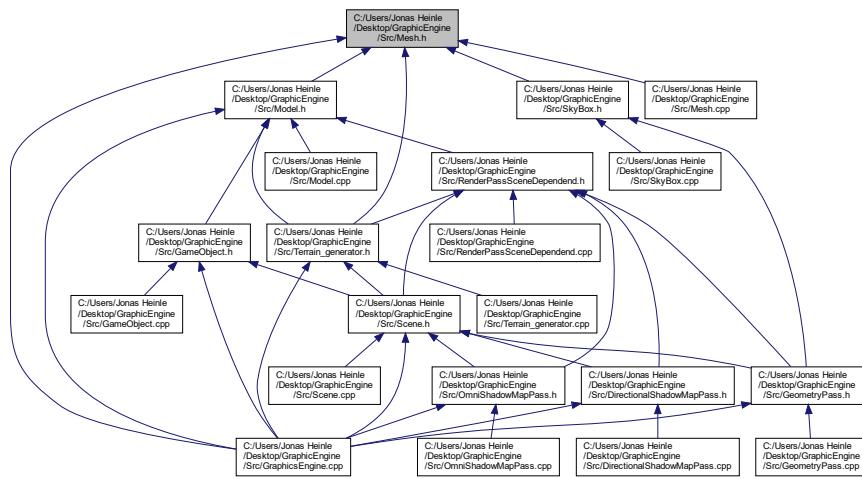


7.63 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.h File Reference

```
#include <glm/glm.hpp>
#include "Texture.h"
#include "Vertex.h"
#include <vector>
#include <GL/glew.h>
#include "GlobalValues.h"
#include "glErrorChecker.h"
Include dependency graph for Mesh.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Mesh](#)

7.64 Mesh.h

[Go to the documentation of this file.](#)

```

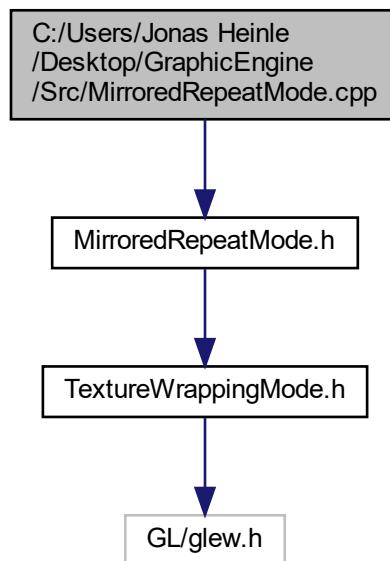
1 #pragma once
2 #include <glm/glm.hpp>
3 #include "Texture.h"
4 #include "Vertex.h"
5 #include <vector>
6 #include <GL/glew.h>
7
8 #include "GlobalValues.h"
9
10 #include "glErrorChecker.h"
11
12
13 using namespace std;
14
15 // this a simple Mesh without mesh generation
16 class Mesh {
17 public:
18
19     Mesh(std::vector<Vertex> vertices, std::vector<unsigned int> indices);
20
21     Mesh();
22
23
24     glm::mat4 transform_Mesh(glm::vec3 translate_vec, glm::vec3 scale = glm::vec3(1.0f), float angle =
25         0.0f, glm::vec3 rotateAxis = glm::vec3(1.0f, 0.0f, 0.0f));
26     void expand(std::vector<Vertex> vertices, std::vector<unsigned int> indices);
27     void changeVertext(std::vector<Vertex> vertices);
28     void render();
29     std::vector<Vertex> getVertices() {
30         return this->vertices;
31     }
32     std::vector<unsigned int> getIndices() {
33         return this->indices;
34     }
35     ~Mesh();
36
37 private:
38     // render data
39     // unsigned int VAO, VBO, EBO;
  
```

```
40
41     enum {
42         POSITION = 0,
43         NORMAL = 1,
44         TEXTURECOORD = 2
45
46     };
47
48     enum
49     {
50         POSITION_VB,
51         NUM_BUFFERS
52     };
53
54     // Vertex Array Object
55     GLuint m_vao, m_ibo;
56     // Vertex array buffer
57     GLuint m_vab[NUM_BUFFERS];
58
59     unsigned int m_drawCount;
60     std::vector<Vertex> vertices;
61     std::vector<unsigned int> indices;
62
63     // for gl error checks
64     glErrorChecker glErrorChecker_ins;
65
66 };
67
68
69
```

7.65 C:/Users/Jonas

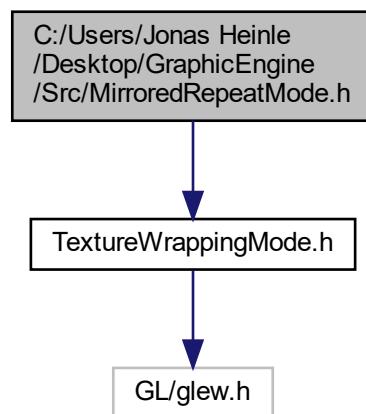
Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.cpp File Reference

```
#include "MirroredRepeatMode.h"
Include dependency graph for MirroredRepeatMode.cpp:
```

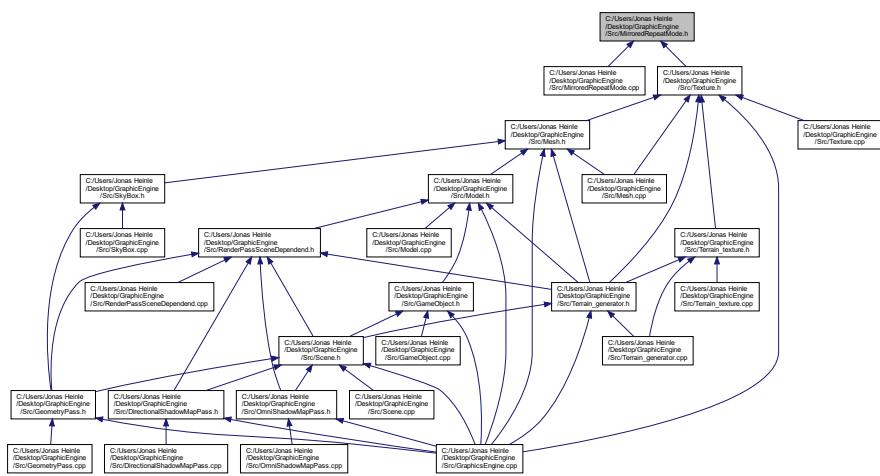


7.66 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.h File Reference

```
#include "TextureWrappingMode.h"
Include dependency graph for MirroredRepeatMode.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [MirroredRepeatMode](#)

7.67 MirroredRepeatMode.h

[Go to the documentation of this file.](#)

```

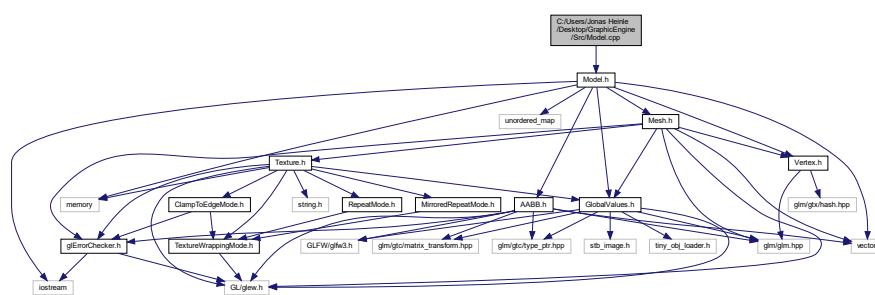
1 #pragma once
2 #include "TextureWrappingMode.h"
3 class MirroredRepeatMode :
4     public TextureWrappingMode
5 {
6 public:
7
8     MirroredRepeatMode();
9     void activate();
10    ~MirroredRepeatMode();
11
12 private:
13
14 };
15

```

7.68 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.cpp File Reference

```
#include "Model.h"
```

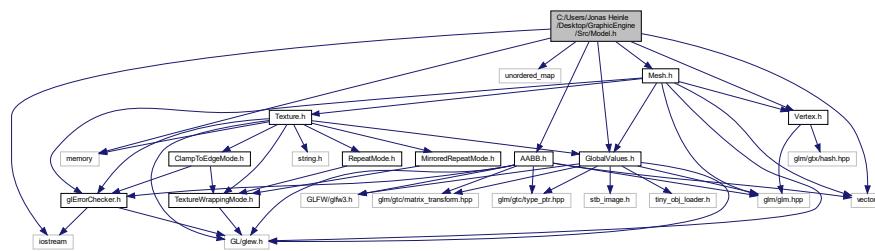
Include dependency graph for Model.cpp:



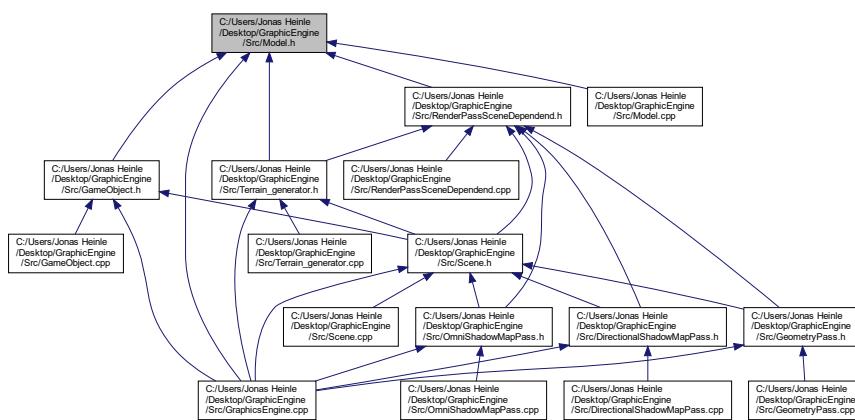
7.69 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.h File Reference

```
#include <iostream>
#include <vector>
#include <unordered_map>
#include "Mesh.h"
#include "Vertex.h"
#include "AABB.h"
#include <memory>
#include "GlobalValues.h"
```

Include dependency graph for Model.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Model

7.70 Model.h

Go to the documentation of this file.

```

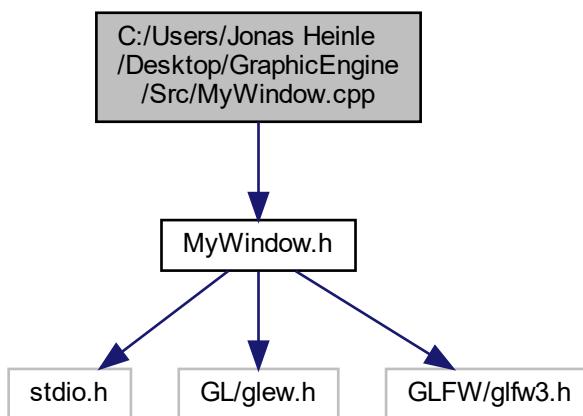
1 #pragma once
2 #include <iostream>
3 #include <vector>
4 #include <unordered_map>
5
6 #include "Mesh.h"
7 #include "Vertex.h"
8 #include "AABB.h"
9
10 #include <memory>
11
12 #include "GlobalValues.h"
13
14 class Model
15 {
16 public:
17     Model();
18
19     void load_model_in_ram(std::string model_path);
20
21
  
```

```

22     void create_render_context();
23     std::string get_base_dir(const std::string& filepath);
24     std::shared_ptr<AABB> get_aabb();
25
26     void render();
27
28     void transform_model(glm::vec3 translate_vec, glm::vec3 scale = glm::vec3(1.0f), float angle = 0.0f,
29                         glm::vec3 rotateAxis = glm::vec3(1.0f, 0.0f, 0.0f));
30
31     ~Model();
32
33     private:
34     std::shared_ptr<AABB> aabb;
35
36     GLuint num_tex;
37
38     std::vector<std::shared_ptr<Mesh>> shapes;
39     //std::vector<Mesh*> mesh_list;
40     std::vector<std::shared_ptr<Texture>> texture_list;
41
42     std::vector<std::vector<Vertex>> vertices_per_shape;
43     std::vector<std::vector<unsigned int>> indices_per_shape;
44
45     std::vector<unsigned int> shapes_to_material;
46     std::vector<int> material_to_tex;
47
48     GLfloat minX = std::numeric_limits<float>::max();
49     GLfloat maxX = std::numeric_limits<float>::min();
50     GLfloat minY = std::numeric_limits<float>::max();
51     GLfloat maxY = std::numeric_limits<float>::min();
52     GLfloat minZ = std::numeric_limits<float>::max();
53     GLfloat maxZ = std::numeric_limits<float>::min();
54 }
55 
```

7.71 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.cpp File Reference

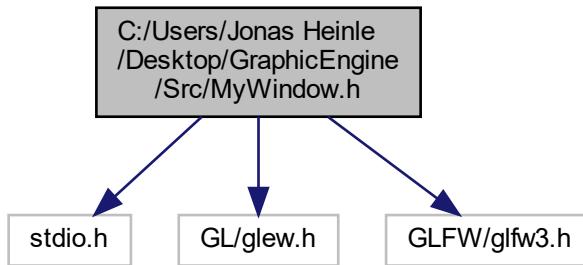
#include "MyWindow.h"
 Include dependency graph for MyWindow.cpp:



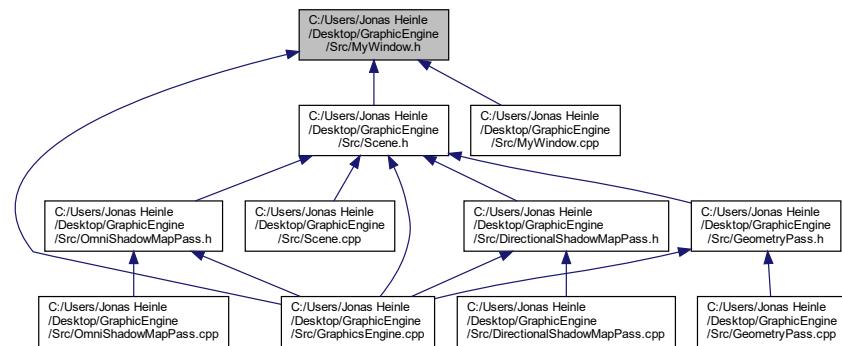
7.72 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.h File Reference

```
#include <stdio.h>
#include <GL/glew.h>
#include <GLFW/glfw3.h>
```

Include dependency graph for MyWindow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [MyWindow](#)

7.73 MyWindow.h

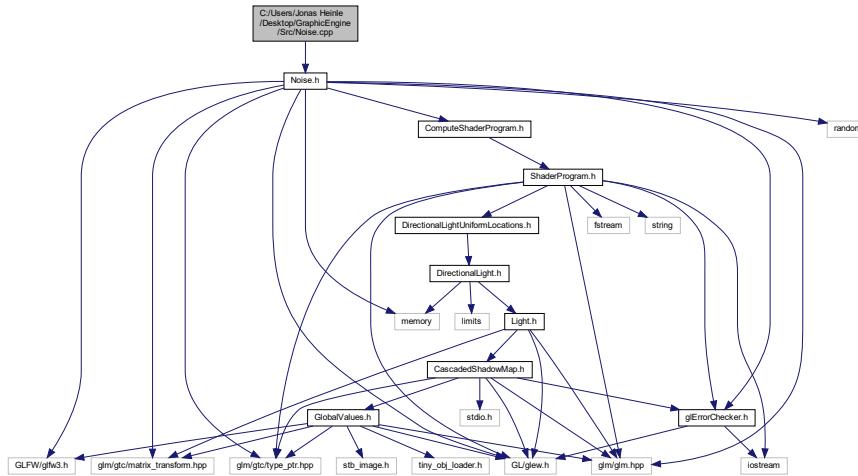
[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <stdio.h>
3
4 #include <GL/glew.h>
```

```
5 #include <GLFW/glfw3.h>
6
7 class MyWindow
8 {
9 public:
10     MyWindow();
11     MyWindow(GLint window_width, GLint window_height);
12
13     bool get_should_close() { return glfwWindowShouldClose(main_window); }
14     void swap_buffers() { glfwSwapBuffers(main_window); }
15
16
17     // init glfw and its context ...
18     int initialize();
19
20     void update_viewport();
21
22     GLfloat get_buffer_width() { return (GLfloat)window_buffer_width; }
23     GLfloat get_buffer_height() { return (GLfloat)window_buffer_height; }
24
25     GLfloat get_x_change();
26     GLfloat get_y_change();
27
28     void set_buffer_size(GLfloat window_buffer_width, GLfloat window_buffer_height);
29
30     GLFWwindow* get_window() { return main_window; }
31
32     bool* get_keys() { return keys; }
33
34     ~MyWindow();
35
36 private:
37
38     GLFWwindow* main_window;
39     GLint window_width, window_height;
40     // what key(-s) was/were pressed
41     bool keys[1024];
42     GLfloat last_x;
43     GLfloat last_y;
44     GLfloat x_change;
45     GLfloat y_change;
46     bool mouse_first_moved;
47
48     //buffers to store our window data to
49     GLint window_buffer_width, window_buffer_height;
50
51     //we need to start our window callbacks for interaction
52     void init_callbacks();
53     static void framebuffer_size_callback(GLFWwindow* window, int width, int height);
54     //after window resizing, update framebuffers
55
56     //need to be static ...
57     static void key_callback(GLFWwindow* window, int key, int code, int action, int mode);
58     static void mouse_callback(GLFWwindow* window, double x_pos, double y_pos);
59     static void mouse_button_callback(GLFWwindow* window, int button, int action, int mods);
60
61 };
62
```

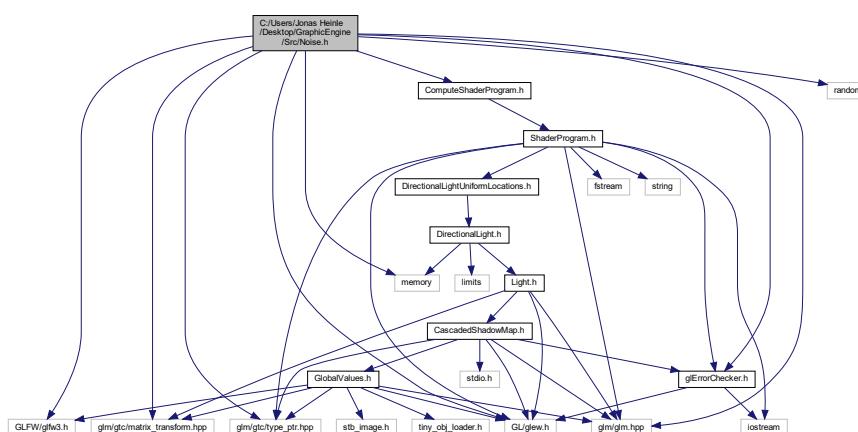
7.74 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.cpp File Reference

```
#include "Noise.h"
Include dependency graph for Noise.cpp:
```

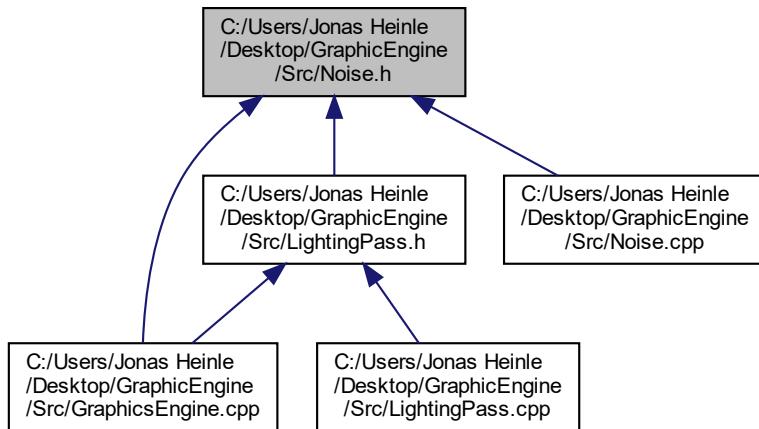


7.75 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.h File Reference

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "ComputeShaderProgram.h"
#include "gLErrorChecker.h"
#include <random>
#include <memory>
Include dependency graph for Noise.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Noise](#)

7.76 Noise.h

[Go to the documentation of this file.](#)

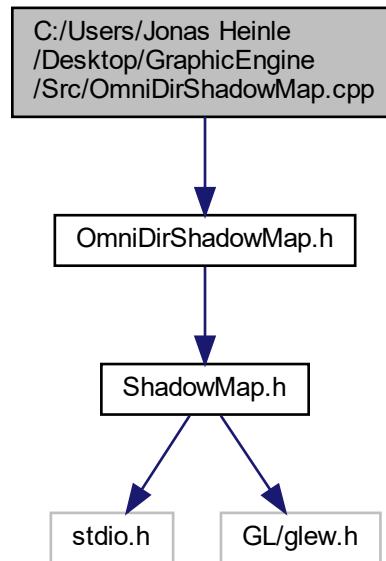
```

1 #pragma once
2
3 #include <GL/glew.h>
4 #include <GLFW/glfw3.h>
5 #include <glm/glm.hpp>
6 #include <glm/gtc/matrix_transform.hpp>
7 #include <glm/gtc/type_ptr.hpp>
8 #include "ComputeShaderProgram.h"
9
10 #include "glErrorChecker.h"
11
12 #include <random>
13 #include <memory>
14
15 class Noise
16 {
17 public:
18     Noise();
19
20     void create_worley_noise();
21     void create_grad_noise();
22
23     void read_worley_noise(GLenum start_buffer_index);
24     void read_grad_noise(GLenum start_buffer_index);
25
26     void init();
27     void update();
28
29     void set_num_cells(GLuint num_cells_per_axis, GLuint index);
30
31     ~Noise();
32
33 private:
34     void generate_cells(GLuint num_cells_per_axis, GLuint cell_index);
35     void generate_textures();
  
```

```
38     void delete_textures();
39
40     GLuint texture_1, texture_2;
41     GLuint cell_ids[NUM_CELLS];
42
43     GLuint texture_dim_1, texture_dim_2;
44     GLuint num_cells_per_axis[NUM_CELLS];
45
46     ComputeShaderProgram texture_1_shader_program;
47     ComputeShaderProgram texture_2_shader_program;
48
49     std::shared_ptr<GLfloat[]> cell_data [NUM_CELLS];
50
51     glErrorChecker glErrorChecker_ins;
52
53 };
54
```

7.77 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.cpp File Reference

#include "OmniDirShadowMap.h"
Include dependency graph for OmniDirShadowMap.cpp:

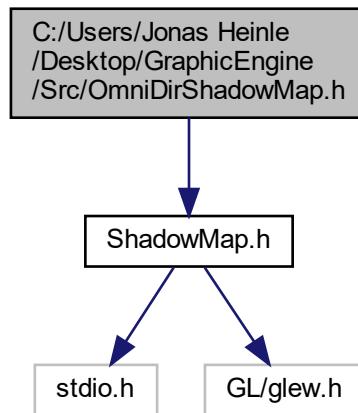


7.78 C:/Users/Jonas

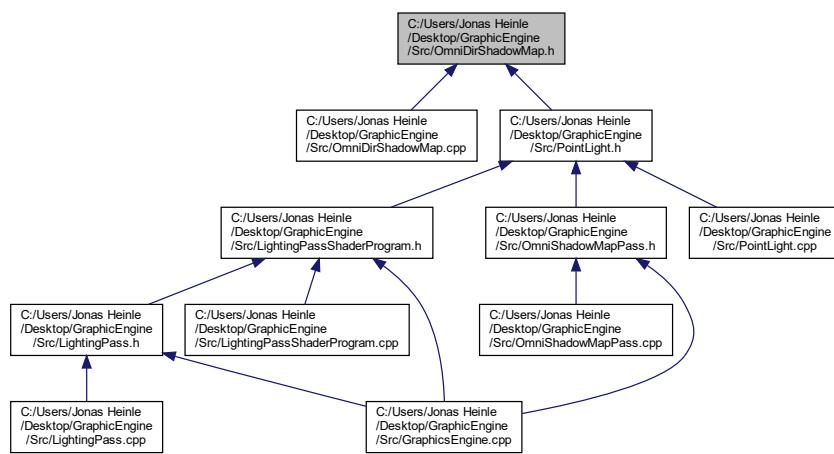
Heinle/Desktop/GraphicEngine/Src/OmniDirShadowMap.h File Reference

```
#include "ShadowMap.h"
```

Include dependency graph for OmniDirShadowMap.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [OmniDirShadowMap](#)

7.79 OmniDirShadowMap.h

[Go to the documentation of this file.](#)

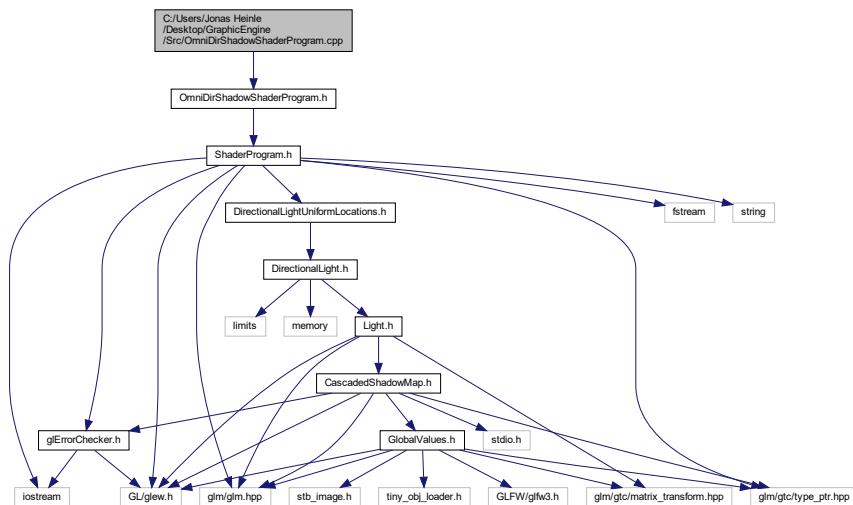
```

1 #pragma once
2 #include "ShadowMap.h"
3
4 class OmniDirShadowMap : public ShadowMap
5 {
6 public:
7     OmniDirShadowMap();
8
9     bool init(GLuint width, GLuint height);
10
11    void write();
12
13    void read(GLenum texture_unit);
14
15    ~OmniDirShadowMap();
16 };
17

```

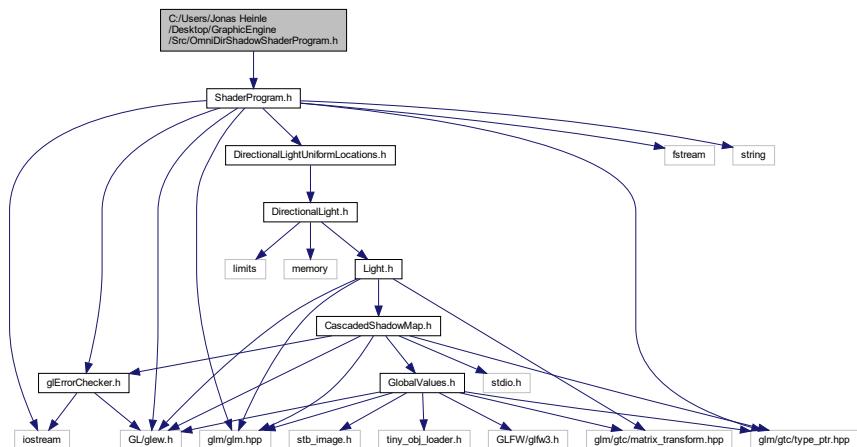
7.80 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.cpp File Reference

#include "OmniDirShadowShaderProgram.h"
 Include dependency graph for OmniDirShadowShaderProgram.cpp:

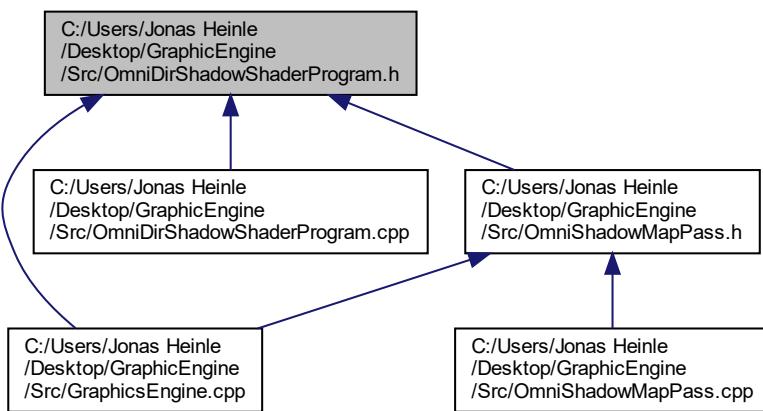


7.81 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDirShadowShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for OmniDirShadowShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [OmniDirShadowShaderProgram](#)

7.82 OmniDirShadowShaderProgram.h

[Go to the documentation of this file.](#)

```

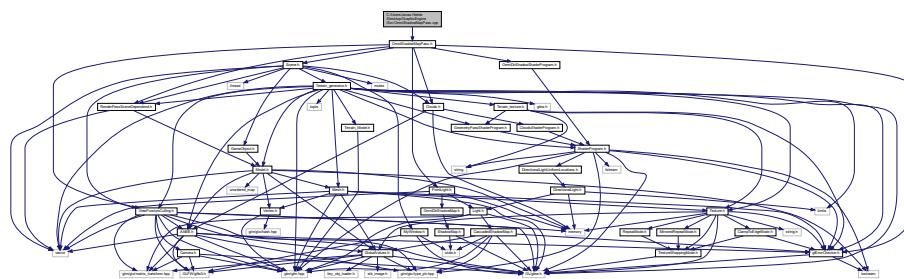
1 #pragma once
2 #include "ShaderProgram.h"
3
4 class OmniDirShadowShaderProgram :
5     public ShaderProgram
6 {
7
8 public:
9
10    OmniDirShadowShaderProgram();
11
12    GLuint get_model_location();
13    GLuint get_omni_light_pos_location();
14    GLuint get_far_plane_location();
15    void reload();
16
17    void set_light_matrices(std::vector<glm::mat4> light_matrices);
18
19 ~OmniDirShadowShaderProgram();
20
21 private:
22
23    GLuint uniform_model_location, uniform_omni_light_pos_location,
24        uniform_far_plane_location;
25
26    GLuint uniform_light_matrices_locations[6];
27
28    void retrieve_uniform_locations();
29
30
31 };
32

```

7.83 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.cpp File Reference

```
#include "OmniShadowMapPass.h"
Include dependency graph for OmniShadowMapPass.cpp:
```

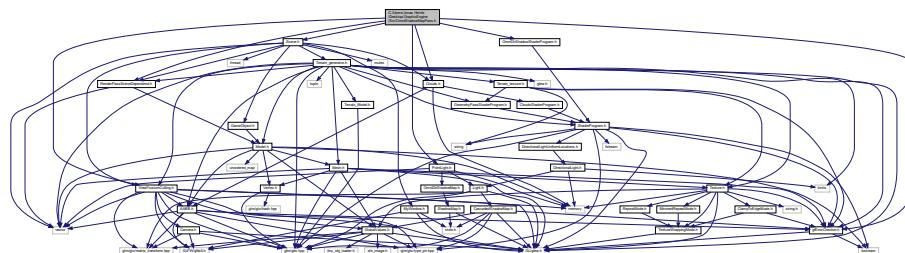


7.84 C:/Users/Jonas

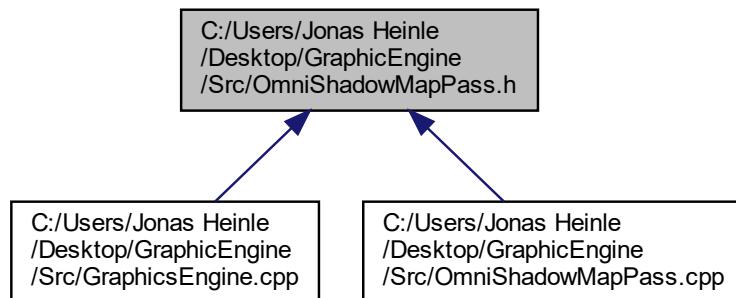
Heinle/Desktop/GraphicEngine/Src/OmniShadowMapPass.h File Reference

```
#include "RenderPassSceneDependend.h"
#include "PointLight.h"
```

```
#include "OmniDirShadowShaderProgram.h"
#include "ViewFrustumCulling.h"
#include "Clouds.h"
#include "Scene.h"
#include "glErrorChecker.h"
Include dependency graph for OmniShadowMapPass.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [OmniShadowMapPass](#)

7.85 OmniShadowMapPass.h

[Go to the documentation of this file.](#)

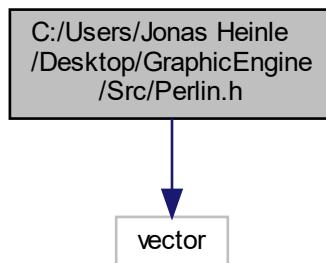
```
1 #pragma once
2 #include "RenderPassSceneDependend.h"
3 #include "PointLight.h"
4 #include "OmniDirShadowShaderProgram.h"
5 #include "ViewFrustumCulling.h"
6 #include "Clouds.h"
7 #include "Scene.h"
8
9 #include "glErrorChecker.h"
10
11 class OmniShadowMapPass :
12     public RenderPassSceneDependend
13 {
14 public:
```

```
15     OmniShadowMapPass();
16     OmniShadowMapPass(std::shared_ptr<OmniDirShadowShaderProgram> shader_program);
17
18     void set_game_object_uniforms(glm::mat4 model, glm::mat4 normal_model, GLuint material_id);
19
20     bool use_terrain_textures();
21
22     void execute(std::shared_ptr<PointLight> p_light, bool first_person_mode, Scene* scene);
23
24     ~OmniShadowMapPass();
25
26 private:
27
28     std::shared_ptr<OmniDirShadowShaderProgram> shader_program;
29
30     // this is for GL error checking
31     glErrorChecker glErrorChecker_ins;
32
33 };
34
```

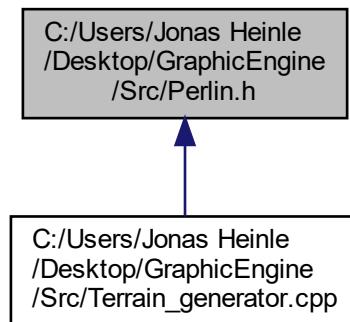
7.86 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.cpp File Reference

7.87 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.h File Reference

```
#include <vector>
Include dependency graph for Perlin.h:
```



This graph shows which files directly or indirectly include this file:



Functions

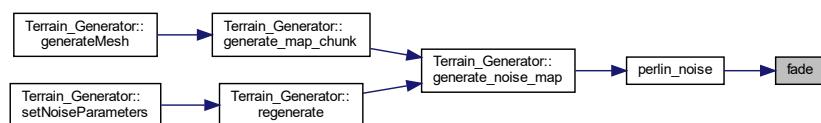
- double **fade** (double t)
- double **lerp** (double t, double a, double b)
- double **grad** (int hash, double x, double y, double z)
- double **perlin_noise** (float x, float y, std::vector< int > &p)
- std::vector< int > **getPermutationVector** ()

7.87.1 Function Documentation

7.87.1.1 fade()

```
double fade (
    double t )
```

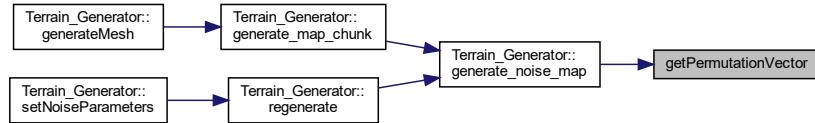
Here is the caller graph for this function:



7.87.1.2 getPermutationVector()

```
std::vector< int > getPermutationVector ( )
```

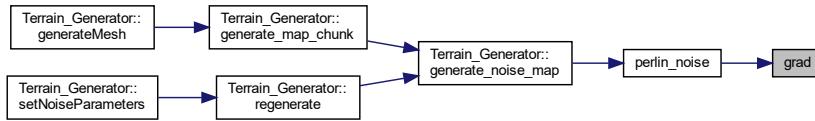
Here is the caller graph for this function:



7.87.1.3 grad()

```
double grad (
    int hash,
    double x,
    double y,
    double z )
```

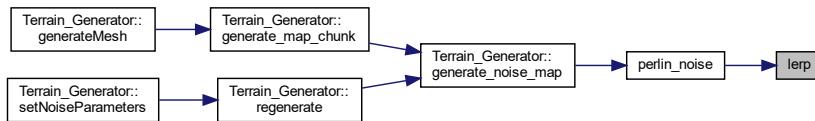
Here is the caller graph for this function:



7.87.1.4 lerp()

```
double lerp (
    double t,
    double a,
    double b )
```

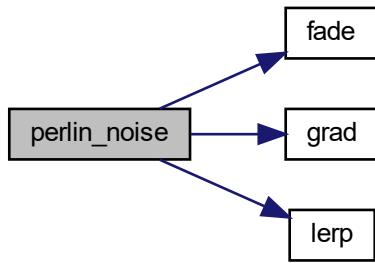
Here is the caller graph for this function:



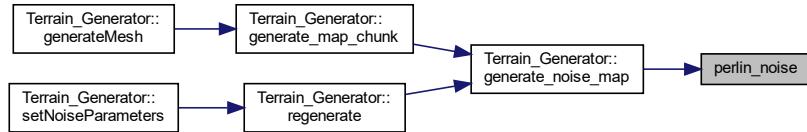
7.87.1.5 perlin_noise()

```
double perlin_noise (
    float x,
    float y,
    std::vector< int > & p )
```

Here is the call graph for this function:



Here is the caller graph for this function:



7.88 Perlin.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <vector>
6 //
7 //double fade(double t);
8 //double lerp(double t, double a, double b);
9 //double grad(int hash, double x, double y, double z);
10 //double perlin_noise(float x, float y, std::vector<int>& p);
11 //
14 //std::vector<int> myFunction(int value, double bullshit) {
15 //  std::vector<int> r{ 2,3,4,4 };
16 //  return r;
17 //}
18 //
19 //void get_Permutation_Vector(std::vector<int>* pVectorPointer);
20 //
21 //
22 //
24
25
26
```

```

27 // fade(t) = t^3 (10 + t * (t * 6 - 15))
28 double fade(double t) {
29
30     return t * t * t * (t * (t * 6 - 15) + 10);
31 }
32
33 // linear interpolation between a and b: For polishing
34 // t = [0,1] is the percentage of the distance between a and b
35 double lerp(double t, double a, double b) {
36     return a + (b - a) * t;
37 }
38
39
40 //
41 double grad(int hash, double x, double y, double z) {
42     int h = hash & 15;                                // CONVERT LO 4 BITS OF HASH CODE
43     double u = h < 8 ? x : y,                         // INTO 12 GRADIENT DIRECTIONS.
44         v = h < 4 ? y : h == 12 || h == 14 ? x : z;
45     return ((h & 1) == 0 ? u : -u) + ((h & 2) == 0 ? v : -v);
46 }
47
48
49 // generate Perlin noise
50 // Parameter:
51 // p is (pseudo) "random" value
52 double perlin_noise(float x, float y, std::vector<int>& p) {
53     int z = 0.5;
54
55     // x,y,z abrunden und mit 1111 1111 verunden == modulo 255
56     int X = (int)floor(x) & 255,                      // FIND UNIT CUBE THAT
57         Y = (int)floor(y) & 255,                      // CONTAINS POINT.
58         Z = (int)floor(z) & 255;
59
60     // Kommastellen
61     x -= floor(x);                                  // FIND RELATIVE X,Y,Z
62     y -= floor(y);                                  // OF POINT IN CUBE.
63     z -= floor(z);
64
65     double u = fade(x),                            // COMPUTE FADE CURVES
66         v = fade(y),                            // FOR EACH OF X,Y,Z.
67         w = fade(z);
68
69     int A = p[X] + Y, AA = p[A] + Z, AB = p[A + 1] + Z,      // HASH COORDINATES OF
70     B = p[X + 1] + Y, BA = p[B] + Z, BB = p[B + 1] + Z;      // THE 8 CUBE CORNERS,
71
72     // return perlin noise of x and y
73     return lerp(w,
74         lerp(v, lerp(u, grad(p[AA], x, y, z), // AND ADD
75             grad(p[BA], x - 1, y, z)), // BLENDED
76             lerp(u, grad(p[AB], x, y - 1, z), // RESULTS
77                 grad(p[BB], x - 1, y - 1, z))), // FROM 8
78         lerp(v, lerp(u, grad(p[AA + 1], x, y, z - 1), // CORNERS
79             grad(p[BA + 1], x - 1, y, z - 1)), // OF CUBE
80             lerp(u, grad(p[AB + 1], x, y - 1, z - 1),
81                 grad(p[BB + 1], x - 1, y - 1, z - 1)))); // OF CUBE
82 }
83 //
84 //void get_Permutation_Vector(std::vector<int>* permVectorPointer) {
85 //    permVectorPointer->clear();
86 //
87 //    std::vector<int> permutation = { 151,160,137,91,90,15,
88 //        131,13,201,95,96,53,194,233,7,225,140,36,103,30,69,142,8,99,37,240,21,10,23,
89 //        190, 6,148,247,120,234,75,0,26,197,62,94,252,219,203,117,35,11,32,57,177,33,
90 //        88,237,149,56,87,174,20,125,136,171,168, 68,175,74,165,71,134,139,48,27,166,
91 //        77,146,158,231,83,111,229,122,60,211,133,230,220,105,92,41,55,46,245,40,244,
92 //        102,143,54, 65,25,63,161, 1,216,80,73,209,76,132,187,208, 89,18,169,200,196,
93 //        135,130,116,188,159,86,164,100,109,198,173,186, 3,64,52,217,226,250,124,123,
94 //        5,202,38,147,118,126,255,82,85,212,207,206,59,227,47,16,58,17,182,189,28,42,
95 //        223,183,170,213,119,248,152, 2,44,154,163, 70,221,153,101,155,167, 43,172,9,
96 //        129,22,39,253, 19,98,108,110,79,113,224,232,178,185, 112,104,218,246,97,228,
97 //        251,34,242,193,238,210,144,12,191,179,162,241, 81,51,145,235,249,14,239,107,
98 //        49,192,214, 31,181,199,106,157,184, 84,204,176,115,121,50,45,127, 4,150,254,
99 //        138,236,205,93,222,114,67,29,24,72,243,141,128,195,78,66,215,61,156,180
100 //    };
101 //
102 //    for (int j = 0; j < 2; j++)
103 //        for (int i = 0; i < 256; i++) {
104 //            permVectorPointer->push_back(permutation[i]);
105 //        }
106 //
107
108
109 //initialise 2x permutation of a hard coded 256 int vector
110 //std::vector<int> PERLIN_H::kansei()
111
112
113 std::vector<int> getPermutationVector() {

```

```

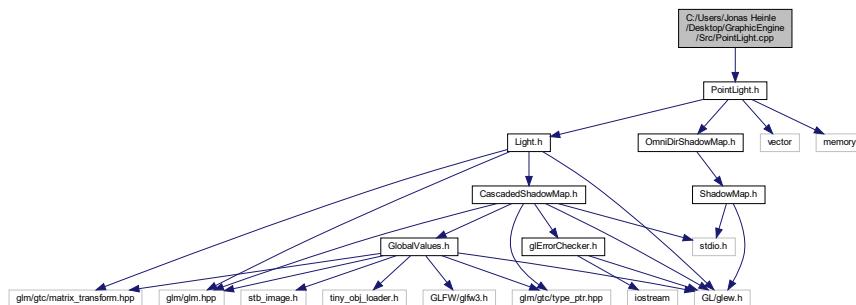
114     std::vector<int> p;
115
116     std::vector<int> permutation = { 151,160,137,91,90,15,
117         131,13,201,95,96,53,194,233,7,225,140,36,103,30,69,142,8,99,37,240,21,10,23,
118         190, 6,148,247,120,234,75,0,26,197,62,94,252,219,203,117,35,11,32,57,177,33,
119         88,237,149,56,87,174,20,125,136,171,168, 68,175,74,165,71,134,139,48,27,166,
120         77,146,158,231,83,111,229,122,60,211,133,230,220,105,92,41,55,46,245,40,244,
121         102,143,54, 65,25,63,161, 1,216,80,73,209,76,132,187,208, 89,18,169,200,196,
122         135,130,116,188,159,86,164,100,109,198,173,186, 3,64,52,217,226,250,124,123,
123         5,202,38,147,118,126,255,82,85,212,207,206,59,227,47,16,58,17,182,189,28,42,
124         223,183,170,213,119,248,152, 2,44,154,163, 70,221,153,101,155,167, 43,172,9,
125         129,22,39,253, 19,98,108,110,79,113,224,232,178,185, 112,104,218,246,97,228,
126         251,34,242,193,238,210,144,12,191,179,162,241, 81,51,145,235,249,14,239,107,
127         49,192,214, 31,181,199,106,157,184, 84,204,176,115,121,50,45,127, 4,150,254,
128         138,236,205,93,222,114,67,29,24,72,243,141,128,195,78,66,215,61,156,180
129     };
130
131     for (int j = 0; j < 2; j++)
132         for (int i = 0; i < 256; i++) {
133             p.push_back(permutation[i]);
134         }
135
136     return p;
137 }
138
139

```

7.89 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.cpp File Reference

```
#include "PointLight.h"
```

Include dependency graph for PointLight.cpp:

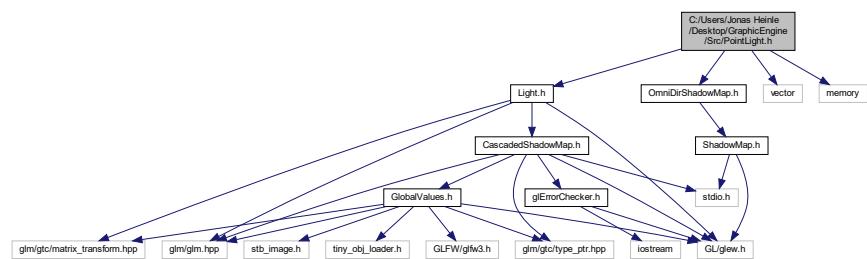


7.90 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.h File Reference

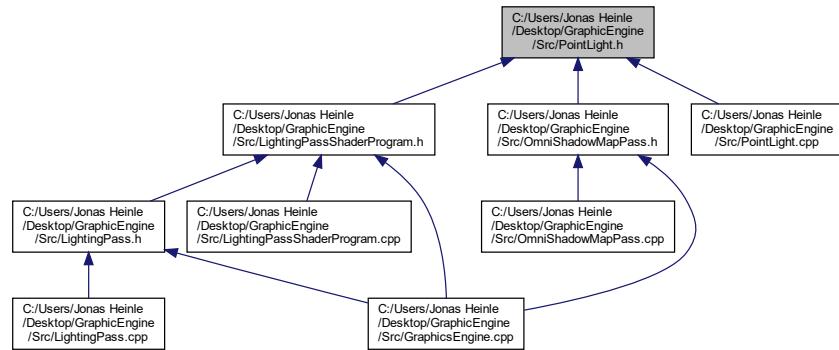
```

#include "Light.h"
#include <vector>
#include "OmniDirShadowMap.h"
#include <memory>
```

Include dependency graph for PointLight.h:



This graph shows which files directly or indirectly include this file:



Classes

- class PointLight

7.91 PointLight.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include "Light.h"
3 #include <vector>
4 #include "OmniDirShadowMap.h"
5 #include <memory>
6
7 class PointLight :
8     public Light
9 {
10
11 public:
12     PointLight();
13     PointLight(GLfloat shadow_width, GLfloat shadow_height,
14                 GLfloat near, GLfloat far,
15                 GLfloat red, GLfloat green, GLfloat blue,
16                 GLfloat a_intensity, GLfloat d_intensity,
17                 GLfloat x_pos, GLfloat y_pos, GLfloat z_pos,
18                 GLfloat con, GLfloat lin, GLfloat exp);
19
20     void use_light(GLuint ambient_intensity_location, GLuint ambient_color_location,
21                   GLuint diffuse_intensity_location, GLuint position_location,
22                   GLuint constant_location, GLuint linear_location, GLuint exponent_location);
```

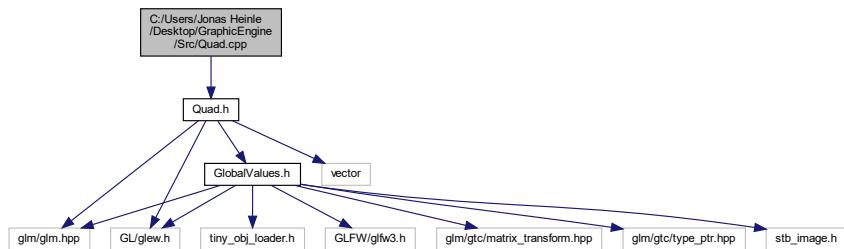
```

23     std::vector<glm::mat4> calculate_light_transform();
24
25     void set_position(glm::vec3 position);
26
27     std::shared_ptr<OmniDirShadowMap> get_omni_shadow_map() { return omni_dir_shadow_map; }
28
29     GLfloat get_far_plane();
30     glm::vec3 get_position();
31
32     ~PointLight();
33
34     protected:
35
36     std::shared_ptr<OmniDirShadowMap> omni_dir_shadow_map;
37
38     glm::vec3 position;
39
40     GLfloat constant, linear, exponent;
41
42     GLfloat far_plane;
43
44 };
45
46

```

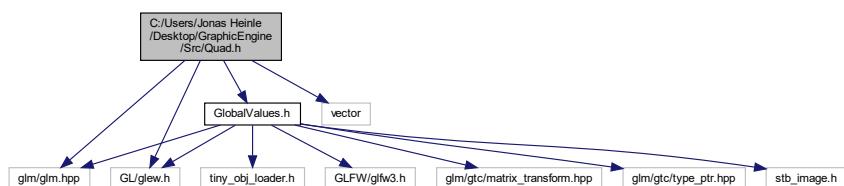
7.92 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.cpp File Reference

#include "Quad.h"
Include dependency graph for Quad.cpp:

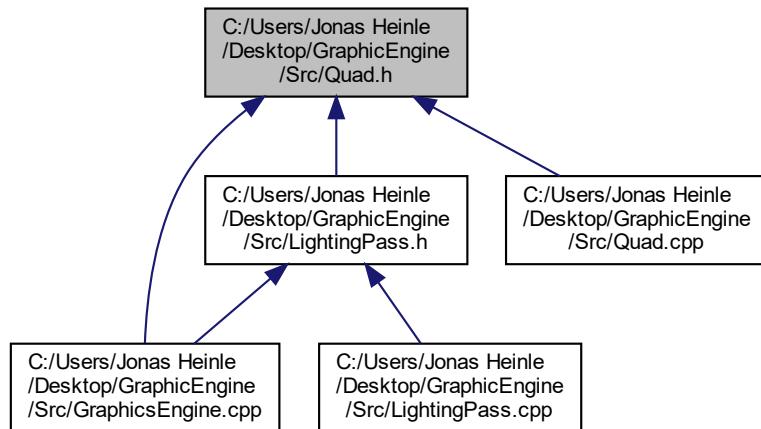


7.93 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.h File Reference

#include <glm/glm.hpp>
#include <vector>
#include <GL/glew.h>
#include "GlobalValues.h"
Include dependency graph for Quad.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [Quad](#)

7.94 Quad.h

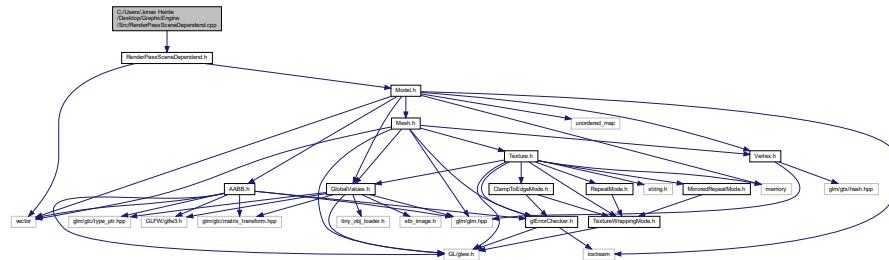
[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <glm/glm.hpp>
3
4 #include <vector>
5 #include <GL/glew.h>
6
7 #include "GlobalValues.h"
8 class Quad
9 {
10 public:
11     Quad();
13     void init();
15     void render();
16
17     ~Quad();
18
19 private:
20     GLuint q_vao, q_vbo;
22
23     float vertices[20] = {
24         //positions           //tex coords
25         -1.0f, 1.0f, 0.0f, 0.0f, 1.0f,
26         -1.0f, -1.0f, 0.0f, 0.0f, 0.0f,
27         1.0f, 1.0f, 0.0f, 1.0f, 1.0f,
28         1.0f, -1.0f, 0.0f, 1.0f, 0.0f
29     };
30
31 };
32
33 };
34
  
```

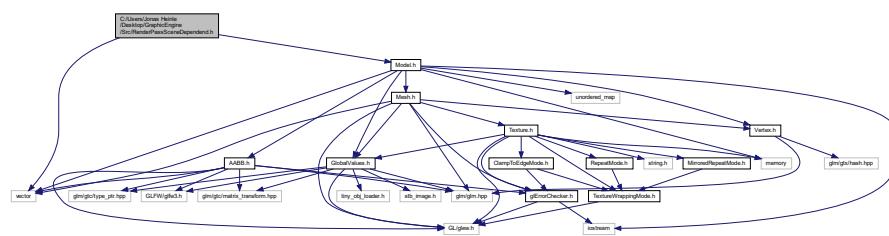
7.95 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.cpp File Reference

```
#include "RenderPassSceneDependend.h"
Include dependency graph for RenderPassSceneDependend.cpp:
```

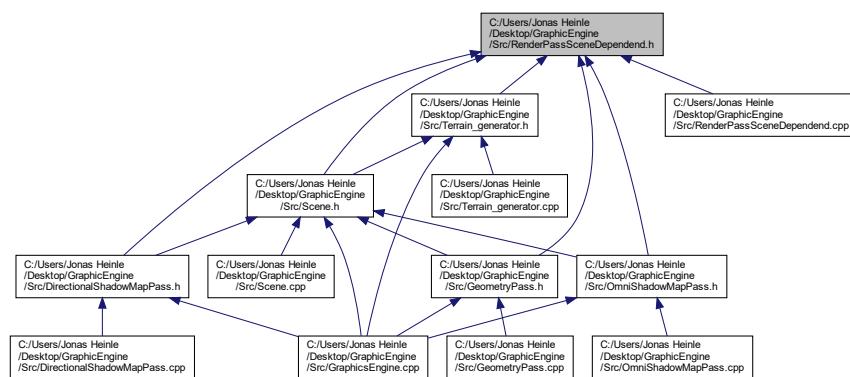


7.96 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPassSceneDependend.h File Reference

```
#include <vector>
#include "Model.h"
Include dependency graph for RenderPassSceneDependend.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [RenderPassSceneDependend](#)

7.97 RenderPassSceneDependend.h

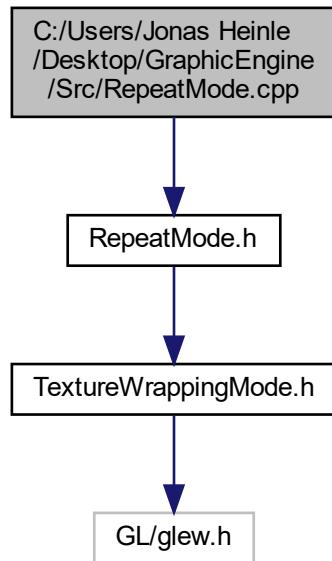
[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <vector>
3 //##include <GL\glew.h>
4 #include "Model.h"
5 //
6 //##include "Terrain_generator.h"
7 //##include "Camera.h"
8 //##include "Clouds.h"
9
10 class RenderPassSceneDependend
11 {
12 public:
13
14     RenderPassSceneDependend();
15
16     virtual void set_game_object_uniforms(glm::mat4 model, glm::mat4 normal_model, GLuint material_id) =
17         0;
18
19     virtual bool use_terrain_textures() = 0;
20
21     ~RenderPassSceneDependend();
22
23 private:
24 };
25
```

7.98 C:/Users/Jonas

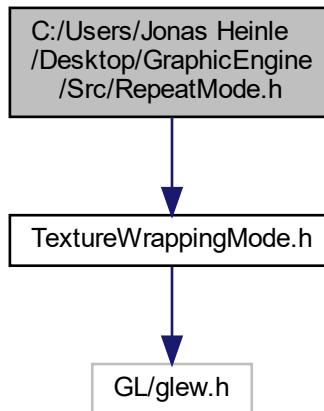
Heinle/Desktop/GraphicEngine/Src/RepeatMode.cpp File Reference

```
#include "RepeatMode.h"  
Include dependency graph for RepeatMode.cpp:
```

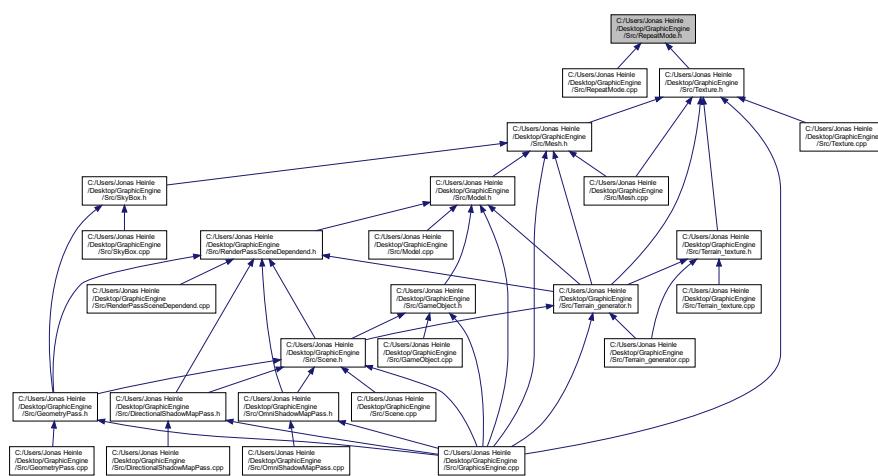


7.99 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.h File Reference

```
#include "TextureWrappingMode.h"
Include dependency graph for RepeatMode.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [RepeatMode](#)

7.100 RepeatMode.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "TextureWrappingMode.h"
3 class RepeatMode :
4     public TextureWrappingMode
5 {
6 public:
7     RepeatMode();
8
9     void activate();
10
11    ~RepeatMode();
12
13 private:
14 };
15

```

7.101 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource.h File Reference

Macros

- `#define IDI_ICON1 101`

7.101.1 Macro Definition Documentation

7.101.1.1 IDI_ICON1

```
#define IDI_ICON1 101
```

7.102 resource.h

[Go to the documentation of this file.](#)

```

1 //{{NO_DEPENDENCIES}}
2 // Von Microsoft Visual C++ generierte Includedatei.
3 // Verwendet durch PlanetExploration.rc
4 //
5 #define IDI_ICON1           101
6
7 // Next default values for new objects
8 //
9 #ifndef APSTUDIO_INVOKED
10 #ifndef APSTUDIO_READONLY_SYMBOLS
11 #define _APS_NEXT_RESOURCE_VALUE      103
12 #define _APS_NEXT_COMMAND_VALUE       40001
13 #define _APS_NEXT_CONTROL_VALUE       1001
14 #define _APS_NEXT_SYMED_VALUE        101
15 #endif
16 #endif

```

7.103 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource1.h File Reference

7.104 resource1.h

[Go to the documentation of this file.](#)

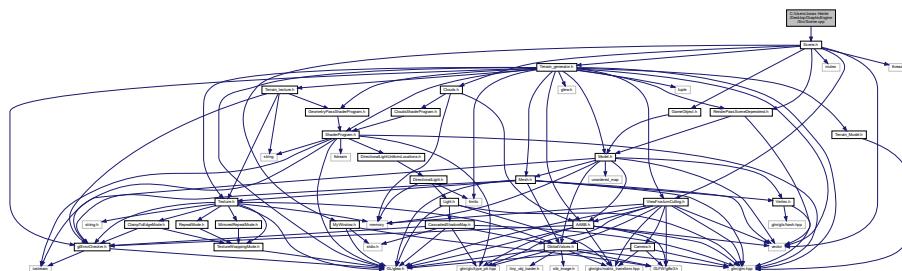
```

1 //{{NO_DEPENDENCIES}}
2 // Microsoft Visual C++ generated include file.
3 // Used by PlanetExploration.rc
4
5 // Nste Standardwerte fr neue Objekte
6 //
7 #ifndef APSTUDIO_INVOKED
8 #ifndef APSTUDIO_READONLY_SYMBOLS
9 #define _APS_NEXT_RESOURCE_VALUE      101
10 #define _APS_NEXT_COMMAND_VALUE       40001
11 #define _APS_NEXT_CONTROL_VALUE       1001
12 #define _APS_NEXT_SYMED_VALUE        101
13#endif
14#endif

```

7.105 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.cpp File Reference

```
#include "Scene.h"
Include dependency graph for Scene.cpp:
```



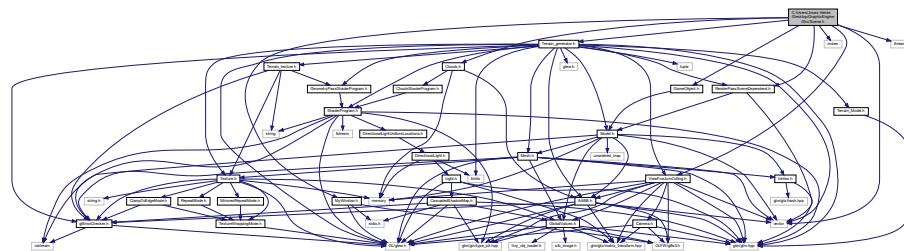
7.106 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.h File Reference

```

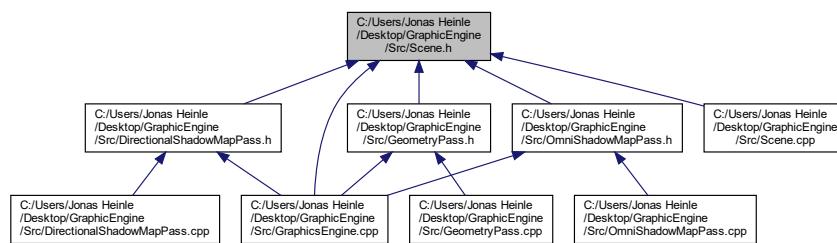
#include <vector>
#include "GameObject.h"
#include "ViewFrustumCulling.h"
#include "RenderPassSceneDependend.h"
#include "MyWindow.h"
#include <mutex>
#include "Terrain_generator.h"
#include "Clouds.h"

```

```
#include <thread>
Include dependency graph for Scene.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Scene](#)

7.107 Scene.h

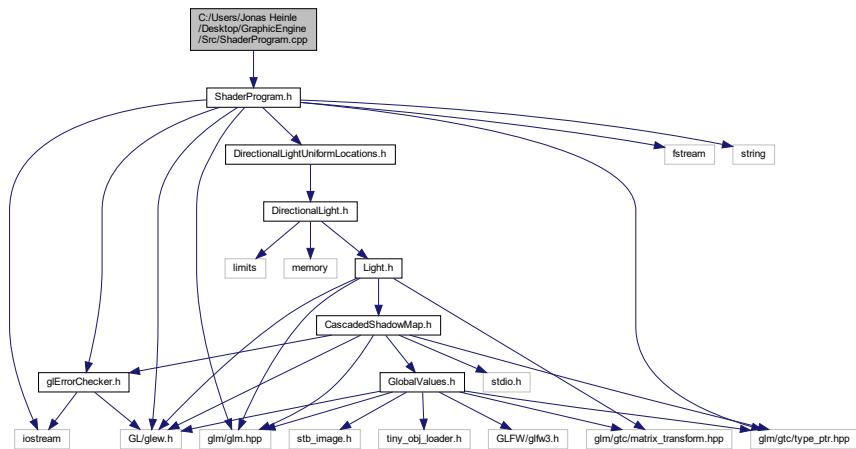
[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <vector>
3 #include "GameObject.h"
4 #include "ViewFrustumCulling.h"
5 #include "RenderPassSceneDependend.h"
6 #include "MyWindow.h"
7 #include <mutex>
8 #include "Terrain_generator.h"
9 #include "Clouds.h"
10 #include <thread>
11
12 class Scene
13 {
14
15 public:
16
17     Scene();
18
19     Scene(const Scene& other);
20
21     std::shared_ptr<Terrain_Generator> get_terrain_generator();
22
23     std::thread spwan() {
24         return std::thread([=] { load_models(); });
25     }
26
27     void init(std::shared_ptr<Camera> main_camera, MyWindow* main_window,
28               std::shared_ptr<Terrain_Generator> terrain_generator, std::shared_ptr<Clouds> clouds);
```

```
28     void render(RenderPassSceneDependend* render_pass, bool first_person_mode);
29     void add_space_ship(std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot,
30     GLuint material_id);
31     void add_ambient_object(std::string model_path, glm::vec3 translation, GLfloat scale, Rotation rot,
32     GLuint material_id);
33     void update_current_space_ship(GLuint selected_space_ship);
34     void load_models();
35     bool is_loaded();
36     GLfloat get_progress();
37     void setup_game_object_context();
38
39     void set_context_setup(bool context_setup);
40     bool get_context_setup();
41
42     ~Scene();
43
44
45 private:
46
47     bool object_is_visible(std::shared_ptr<GameObject> game_object);
48
49     std::shared_ptr<Camera> main_camera;
50     MyWindow* main_window;
51     std::shared_ptr<Terrain_Generator> terrain_generator;
52     std::shared_ptr<ViewFrustumCulling> view_frustum_culling;
53     std::shared_ptr<Clouds> clouds;
54
55     std::vector<std::shared_ptr<GameObject>> space_ships;
56     std::vector<std::shared_ptr<GameObject>> ambient_objects;
57
58     std::vector<glm::vec3> space_ship_offsets;
59     std::vector<GLfloat> rotation_offset;
60
61     GLfloat progress;
62     bool loaded_scene;
63
64     std::mutex mx_progress;
65     std::mutex mx_isLoaded;
66     std::mutex mx_space_ship;
67
68     GLuint current_space_ship_selected;
69
70     bool context_setup;
71
72 };
73
```

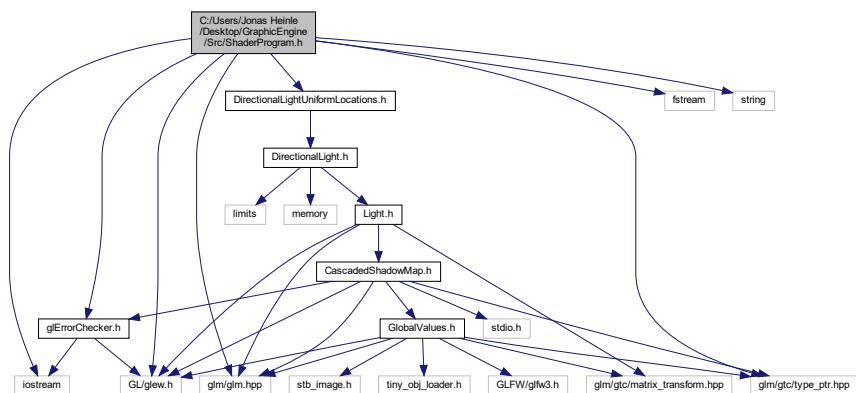
7.108 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.cpp File Reference

```
#include "ShaderProgram.h"
Include dependency graph for ShaderProgram.cpp:
```

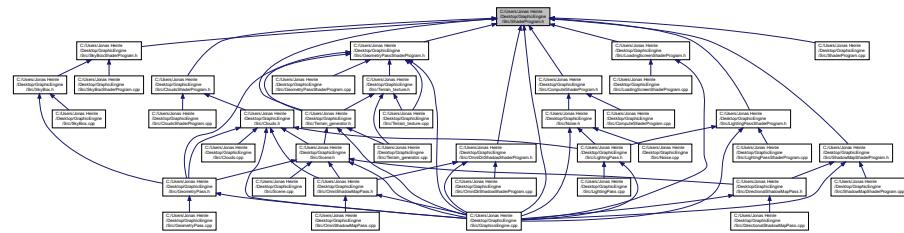


7.109 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.h File Reference

```
#include <iostream>
#include <fstream>
#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/type_ptr.hpp>
#include <string>
#include "DirectionalLightUniformLocations.h"
#include "glErrorChecker.h"
Include dependency graph for ShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [ShaderProgram](#)

7.110 ShaderProgram.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <iostream>
3 #include <fstream>
4
5 #include <GL/glew.h>
6 #include <glm/glm.hpp>
7 #include <glm/gtc/type_ptr.hpp>
8 #include <string>
9
10 // #include "DirectionalLight.h"
11 #include "DirectionalLightUniformLocations.h"
12
13 #include "glErrorChecker.h"
14
15 class ShaderProgram
16 {
17 public:
18
19     ShaderProgram();
20
21     void create_from_files(const char* vertex_location, const char* fragment_location);
22     void create_from_files(const char* vertex_location, const char* geometry_location, const char* fragment_location);
23     void create_computer_shader_program_from_file(const char* compute_location);
24
25     void use_shader_program();
26
27     void reload();
28
29     void validate_program();
30
31     ~ShaderProgram();
32
33 protected:
34
35     GLuint program_id;
36
37     // the file locations from our shaders
38     const char* vertex_location;
39     const char* fragment_location;
40     const char* geometry_location;
41     const char* compute_location;
42
43     std::string read_file(const char* file_location);
44     void add_shader(GLuint program, const char* shader_code, GLenum shader_type);
45     void compile_shader_program(const char* vertex_code, const char* fragment_code);
46     void compile_shader_program(const char* vertex_code, const char* geometry_code, const char* fragment_code);
47     void compile_compute_shader_program(const char * compute_code);
48     void compile_program();
49
50     virtual void retrieve_uniform_locations() = 0;
51

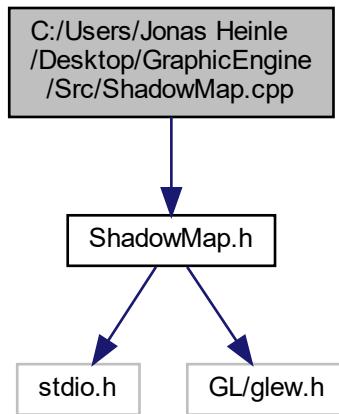
```

```
52     void clear_shader_program();
53     // for checking gl Errors
54     glErrorChecker glErrorChecker_ins;
55
56 };
57
```

7.111 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/ShadowMap.cpp File Reference

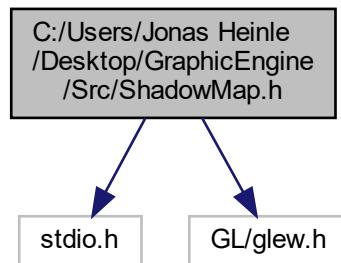
```
#include "ShadowMap.h"
Include dependency graph for ShadowMap.cpp:
```



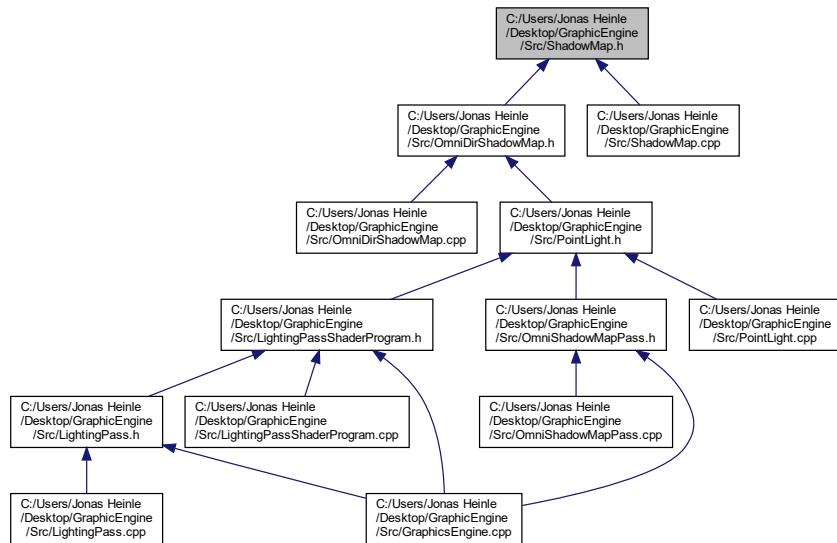
7.112 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMap.h File Reference

```
#include <stdio.h>
#include <GL/glew.h>
```

Include dependency graph for ShadowMap.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [ShadowMap](#)

7.113 ShadowMap.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include <stdio.h>
4 #include <GL/glew.h>
5
  
```

```

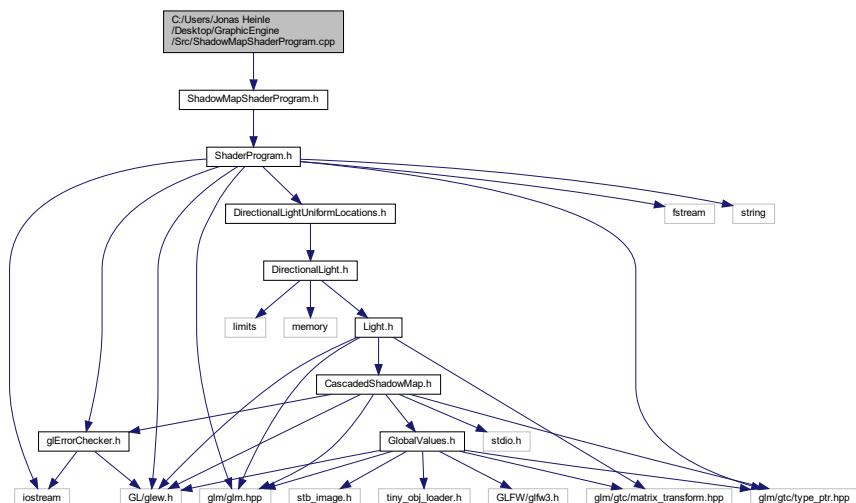
6 class ShadowMap
7 {
8 public:
9
10    ShadowMap();
11
12   virtual bool init(GLuint width, GLuint height);
13
14   virtual void write();
15
16   virtual void read(GLenum texture_unit);
17
18   GLuint get_shadow_width() { return shadow_width; };
19
20   GLuint get_shadow_height() { return shadow_height; };
21
22   GLuint get_id();
23
24   ~ShadowMap();
25
26 protected:
27
28   GLuint FBO, shadow_map;
29   GLuint shadow_width, shadow_height;
30 };
31

```

7.114 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.cpp File Reference

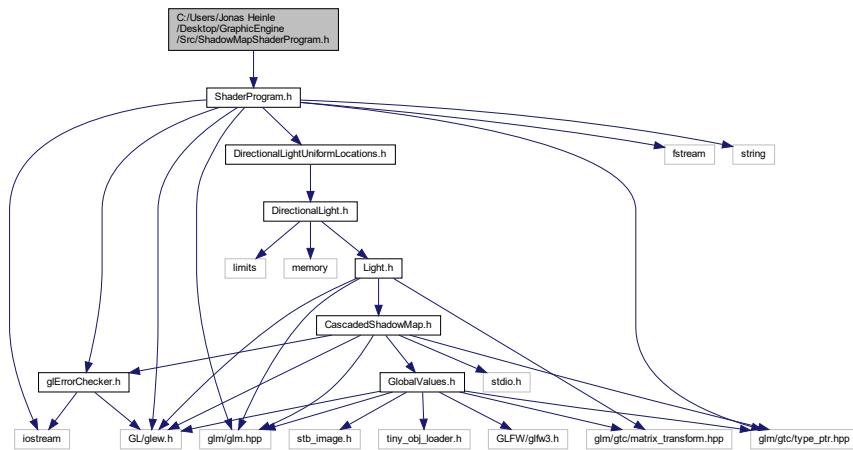
#include "ShadowMapShaderProgram.h"

Include dependency graph for ShadowMapShaderProgram.cpp:

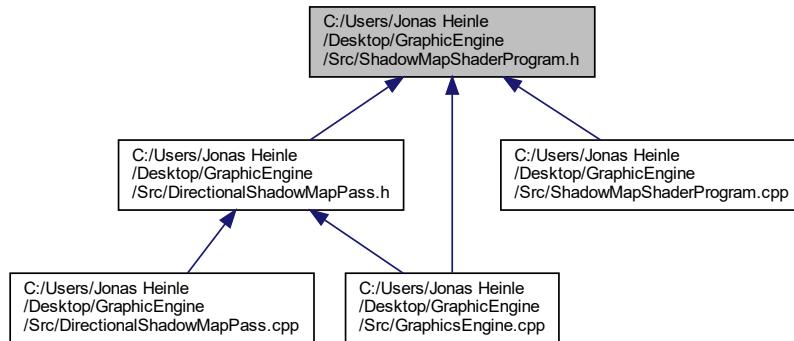


7.115 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for ShadowMapShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [ShadowMapShaderProgram](#)

7.116 ShadowMapShaderProgram.h

[Go to the documentation of this file.](#)

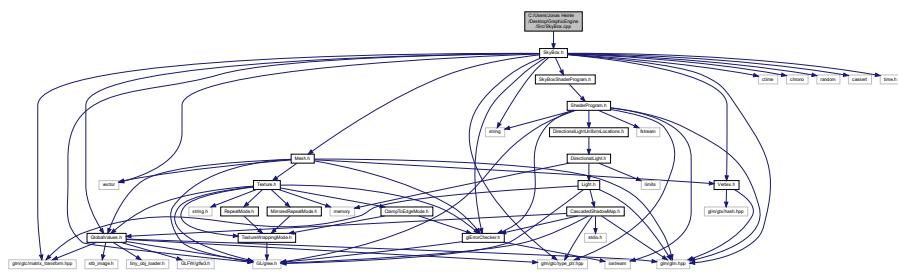
```

1 #pragma once
2 #include "ShaderProgram.h"
3 class ShadowMapShaderProgram :
4     public ShaderProgram
5 {
6 public:
7     ShadowMapShaderProgram();
8
9     GLuint get_model_location();
10    GLuint get_directional_light_transform_location();
11    void set_directional_light_transform(glm::mat4& l_trsf);
12
13    ~ShadowMapShaderProgram();
14
15 private:
16     GLuint uniform_model_location, uniform_directional_light_transform_location;
17
18     void retrieve_uniform_locations();
19
20 };
21
22

```

7.117 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.cpp File Reference

#include "SkyBox.h"
 Include dependency graph for SkyBox.cpp:



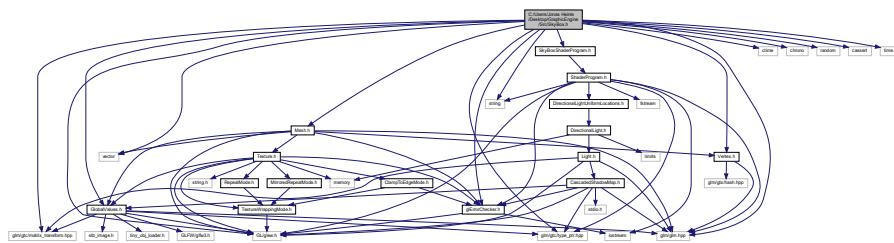
7.118 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.h File Reference

```

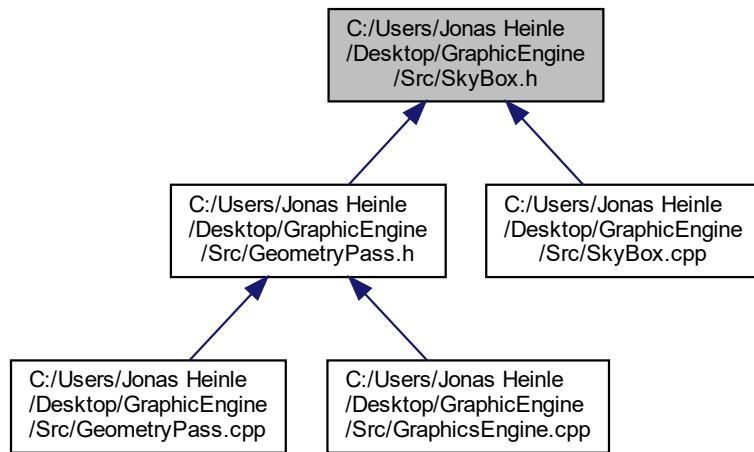
#include <vector>
#include <string>
#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "GlobalValues.h"
#include "Mesh.h"
#include "Vertex.h"
#include "SkyBoxShaderProgram.h"
#include <ctime>
#include <chrono>
#include <random>
#include <cassert>
#include <time.h>

```

```
#include "glErrorChecker.h"
Include dependency graph for SkyBox.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [SkyBox](#)

7.119 SkyBox.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <vector>
3 #include <string>
4
5 #include <GL/glew.h>
6 #include <glm/glm.hpp>
7 #include <glm/gtc/matrix_transform.hpp>
8 #include <glm/gtc/type_ptr.hpp>
9
10 #include "GlobalValues.h"
11
12 #include "Mesh.h"
13 #include "Vertex.h"
14 #include "SkyBoxShaderProgram.h"
```

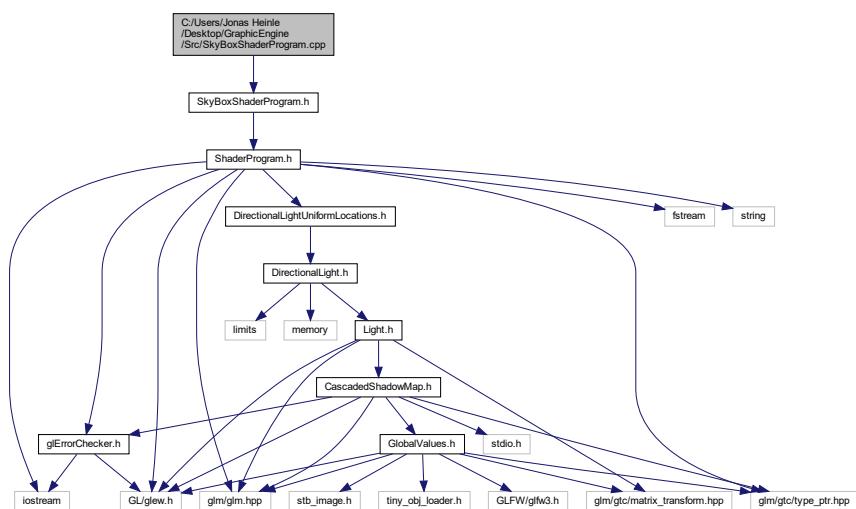
```

15
16 #include <ctime>
17 #include <chrono>
18 #include <random>
19 #include <cassert>
20 #include <time.h>
21
22 #include "glErrorChecker.h"
23
24 class SkyBox
25 {
26 public:
27     SkyBox();
28
29     SkyBox(std::vector<std::string> face_locations);
30
31     void draw_sky_box(glm::mat4 projection_matrix, glm::mat4 view_matrix, GLfloat window_width, GLfloat
32     window_height, GLfloat delta_time);
33     void reload();
34
35     ~SkyBox();
36
37 private:
38     GLfloat movement_speed = 0.1f;
39
40     GLfloat shader_playback_time;
41
42     std::shared_ptr<Mesh> sky_mesh;
43     std::shared_ptr<SkyBoxShaderProgram> sky_shader_program;
44
45     GLuint texture_id;
46     GLuint uniform_projection, uniform_view;
47
48     glErrorChecker glErrorChecker_ins;
49 };

```

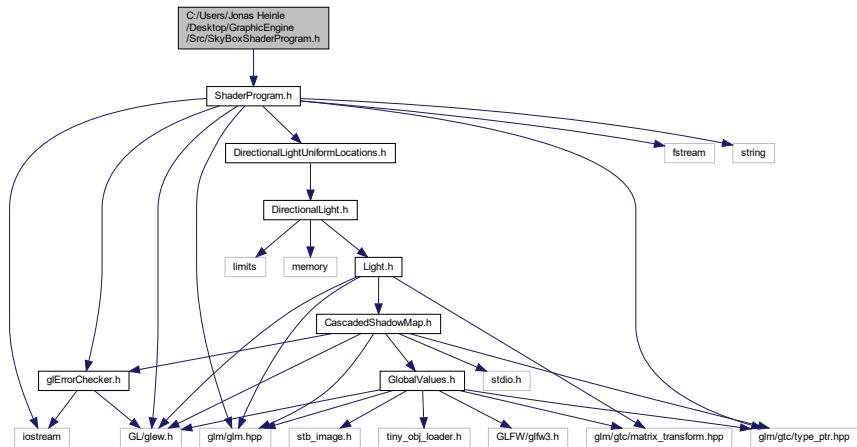
7.120 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.cpp File Reference

#include "SkyBoxShaderProgram.h"
Include dependency graph for SkyBoxShaderProgram.cpp:

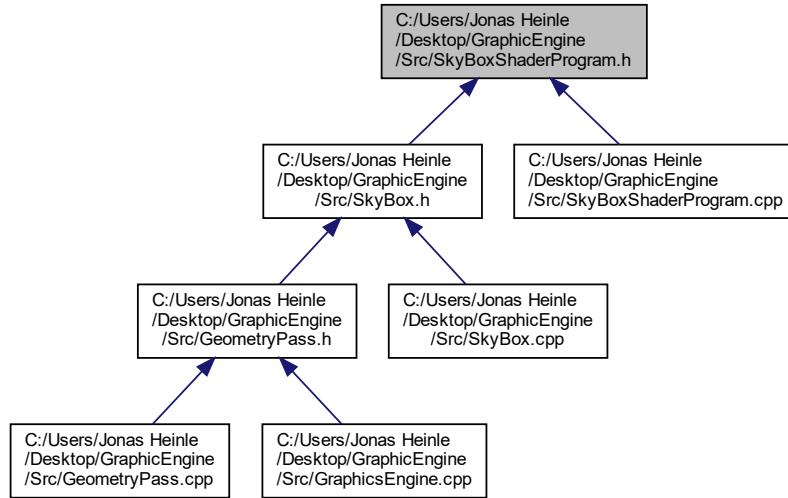


7.121 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.h File Reference

```
#include "ShaderProgram.h"
Include dependency graph for SkyBoxShaderProgram.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [SkyBoxShaderProgram](#)

7.122 SkyBoxShaderProgram.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include "ShaderProgram.h"
3
4 class SkyBoxShaderProgram:
5     public ShaderProgram
6 {
7 public:
8
9     SkyBoxShaderProgram();
10
11    GLuint get_projection_location();
12    GLuint get_view_location();
13
14
15    ~SkyBoxShaderProgram();
16
17 private:
18
19    GLuint uniform_view_location, uniform_projection_location;
20
21    void retrieve_uniform_locations();
22 };
23

```

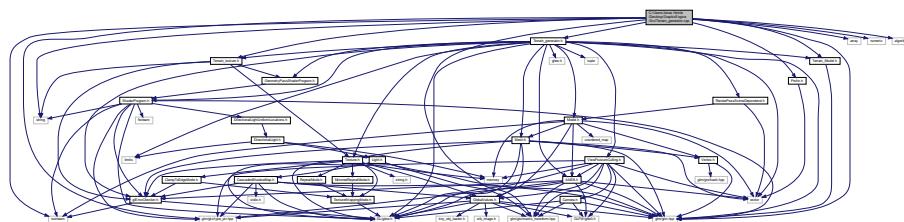
7.123 C:/Users/Jonas

Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp File Reference

```

#include <string>
#include <iostream>
#include <array>
#include <numeric>
#include <algorithm>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include "Terrain_generator.h"
#include "Perlin.h"
#include "Terrain_Model.h"
#include "Terrain_texture.h"
Include dependency graph for Terrain_generator.cpp:

```

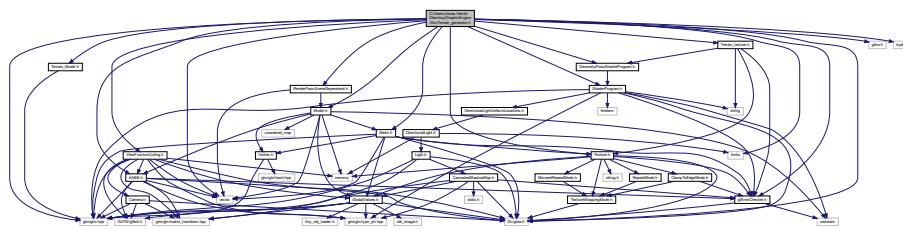


Classes

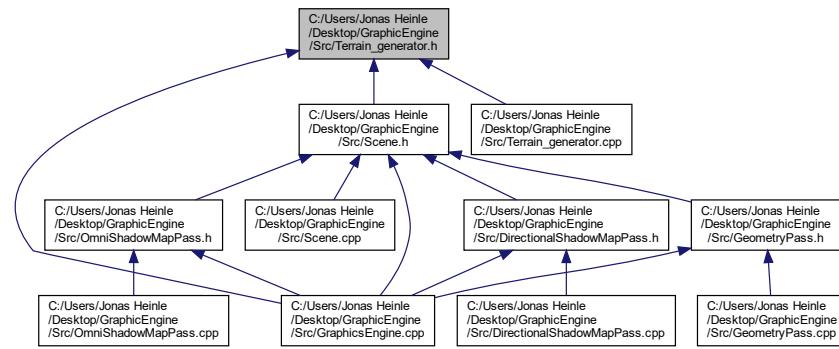
- struct [terrainType](#)

**7.124 C:/Users/Jonas
Heinle/Desktop/GraphicEngine/Src/Terrain_generator.h File
Reference**

```
#include <vector>
#include <glew.h>
#include <tuple>
#include <glm/glm.hpp>
#include <limits>
#include "RenderPassSceneDependend.h"
#include "ViewFrustumCulling.h"
#include <GL/glew.h>
#include "Mesh.h"
#include "Texture.h"
#include "Terrain_Model.h"
#include "Model.h"
#include "Terrain_texture.h"
#include "ShaderProgram.h"
#include "AABB.h"
#include "GeometryPassShaderProgram.h"
#include "glErrorChecker.h"
Include dependency graph for Terrain_generator.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class Terrain Generator

7.125 Terrain_generator.h

[Go to the documentation of this file.](#)

```

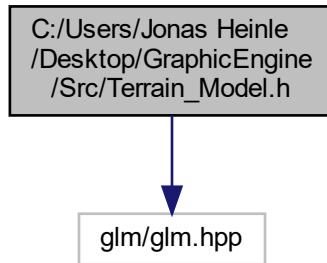
1 #pragma once
2
3 #include <vector>
4 #include <glew.h>
5 #include <tuple>
6 #include <glm/glm.hpp>
7 #include <iomanip>
8 #include "RenderPassSceneDependend.h"
9 #include "ViewFrustumCulling.h"
10
11 #include <GL/glew.h>
12
13 #include "Mesh.h"
14 #include "Texture.h"
15 #include "Terrain_Model.h"
16 #include "Model.h"
17 #include "Terrain_texture.h"
18 #include "ShaderProgram.h"
19 #include "AABB.h"
20 #include "GeometryPassShaderProgram.h"
21
22 #include "glErrorChecker.h"
23
24
25
26 class Terrain_Generator
27 {
28 public:
29
30     Terrain_Generator();
31
32     void init();
33     void retrieve_uniform_locations(std::shared_ptr<GeometryPassShaderProgram> shader_program);
34
35     ~Terrain_Generator();
36
37     void set_world_trafo(glm::mat4 model_to_world);
38     glm::mat4 get_world_trafo();
39     glm::mat4 get_normal_world_trafo();
40     void set_material_id(GLuint material_id);
41     GLuint get_material_id();
42
43     std::shared_ptr<Terrain_Texture> get_textures();
44
45     void generateMesh(GLfloat& progress);
46     void load_plants();
47
48     std::vector<std::shared_ptr<Mesh>> meshes;
49     std::vector<std::shared_ptr<AABB>> aabbs;
50
51     // TODO:
52     void expandTerrain();
53
54     std::vector<GLuint> generate_indices();
55     std::vector<float> generate_noise_map(int xOffset, int yOffset);
56     std::vector<glm::vec3> generate_vertices(const std::vector<float>& noise_map);
57
58     std::vector<glm::vec3> calc_smooth_normals(std::vector<glm::vec3> verticesPos, std::vector<glm::vec3> triangleNormals, std::vector<unsigned int> indices);
59     std::vector<glm::vec3> generate_normals(const std::vector<GLuint>& indices, const std::vector<glm::vec3>& vertices);
60
61
62     void generate_map_chunk(/*GLuint& VAO, */ int xOffset, int yOffset, std::vector<std::vector<Terrain_Model>>& chunk_models);
63
64     std::vector<glm::vec2> generate_biome(const std::vector<glm::vec3>& vertices, const std::vector<GLuint>& indices, int xOffset, int yOffset, std::vector<std::vector<Terrain_Model>>& chunk_models);
65
66     void generate_render_context();
67
68     // Parameter:
69     // view: is the camera position (GetViewMatrix)
70     // model:
71     // projection:
72     std::vector<bool> render(GLfloat ratio, std::shared_ptr<Camera> camera,
73                             std::shared_ptr<ViewFrustumCulling> view_cull, RenderPassSceneDependend* shader_pass);
74
75     void render_plants(std::vector<bool> is_chunk_rendered_flags, RenderPassSceneDependend* shader_pass);
76
77     void transform(glm::vec3 translate_vector, glm::vec3 scale);

```

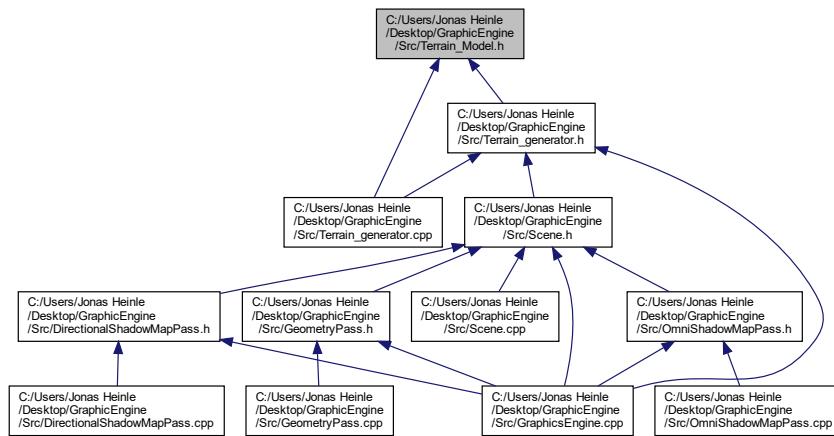
```
78     void changeMaxHeight(float newMaxHeight);
79     void regenerate();
80
81
82
83     // get noise Parameters return (octaves, meshHeight, noiseScale, persistence, lacunarity)
84     std::tuple<int, float, float, float> getNoiseParameters() {
85         return { octaves, meshHeight, noiseScale, persistence, lacunarity };
86     }
87
88     // Set noise Parameters set(octaves, meshHeight, noiseScale, persistence, lacunarity)
89     void setNoiseParameters(std::tuple<int, float, float, float, float> noiseParam) {
90         this->octaves = std::get<0>(noiseParam);
91         this->meshHeight = std::get<1>(noiseParam);
92         this->noiseScale = std::get<2>(noiseParam);
93         this->persistence = std::get<3>(noiseParam);
94         this->lacunarity = std::get<4>(noiseParam);
95
96         regenerate();
97     }
98
99 private:
100     // Terrain Map params
101     float WATER_HEIGHT = 0.1;
102     int chunk_render_distance = 3;
103
104     // Number of chunks
105     int xMapChunks = 10;
106     int yMapChunks = 10;
107
108     // size of one chunk
109     int chunkWidth = 256;
110     int chunkHeight = 256;
111     int gridPosX = 0;
112     int gridPosY = 0;
113
114
115     // Noise params
116     int octaves = 5;
117     // height can be changed in the runtime
118     float meshHeight = 64; // Vertical scaling
119     float noiseScale = 64; // Horizontal scaling
120     float persistence = 0.5;
121     float lacunarity = 2;
122
123     //Texture terrainTex;
124     std::shared_ptr<Terrain_Texture> t_texture;
125     std::vector<float> biomHeights;
126
127
128     // Terrain Models
129     // spawn information of all Models of the Terrain
130     std::vector<std::vector<Terrain_Model>> chunk_models;
131     vector<std::shared_ptr<Model>> trees, bushes, stones;
132
133
134
135     glm::mat4 model_to_world;
136     GLuint material_id;
137     // TODO should I release the Data space wenn the programm is closed or not used?
138     // Terrain Mesh
139     std::vector<std::vector<Vertex>> vertices_per_shape;
140     std::vector<std::vector<unsigned int>> indices_per_shape;
141     std::vector<std::vector<GLfloat>> aabbs_per_chunck;
142     // for gl Error checking
143     glErrorChecker glErrorChecker_ins;
144 }
```

7.126 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_Model.h File Reference

```
#include <glm/glm.hpp>
Include dependency graph for Terrain_Model.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Terrain_Model](#)

7.127 Terrain_Model.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2
3 #include<glm/glm.hpp>
```

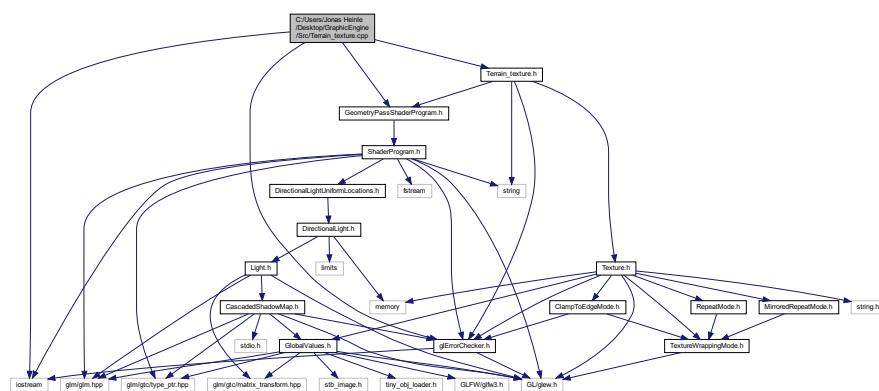
```

4
5
6 class Terrain_Model
7 {
8 public:
9     enum M_TYPE
10    {
11        BUSH,
12        TREE,
13        STONE,
14    };
15
16 Terrain_Model(M_TYPE type, glm::vec3 pos, int xOffset, int yOffset, int variation = 0) {
17     this -> type = type;
18     this->pos = pos;
19     this->xOffset = xOffset;
20     this->yOffset = yOffset;
21     this->variation = variation;
22
23     this->angle = glm::radians((float)(rand() % 360));
24 }
25
26
27     M_TYPE type;
28     glm::vec3 pos;
29     int xOffset;
30     int yOffset;
31     int variation;
32     float angle;
33
34 private:
35
36 };

```

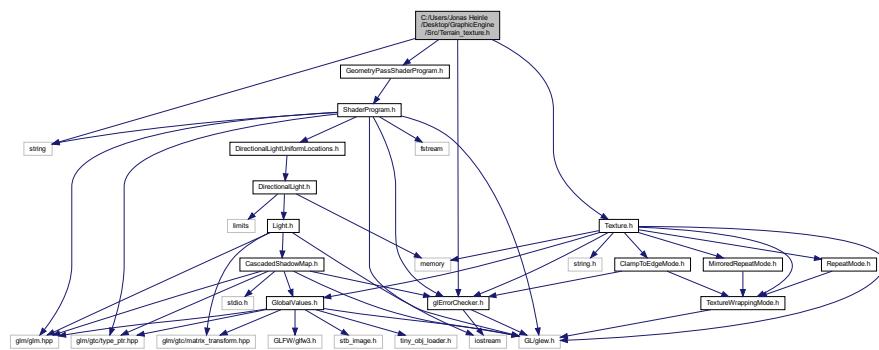
7.128 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.cpp File Reference

```
#include <iostream>
#include "Terrain_texture.h"
#include "GeometryPassShaderProgram.h"
#include "glErrorChecker.h"
Include dependency graph for Terrain_texture.cpp:
```

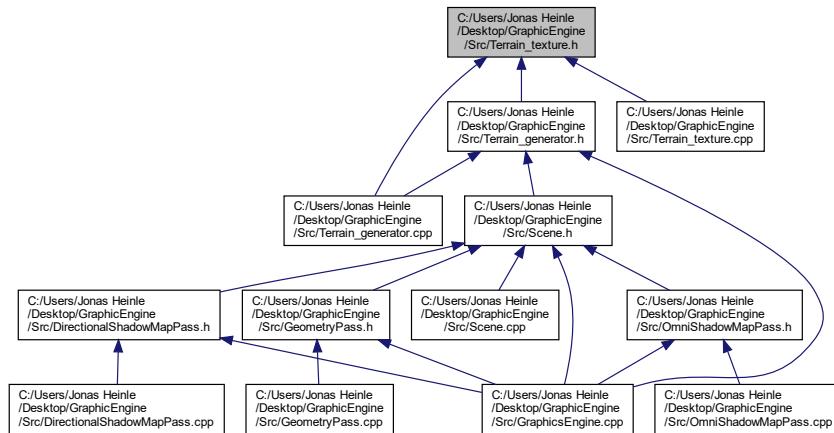


7.129 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.h File Reference

```
#include <string>
#include "GeometryPassShaderProgram.h"
#include "Texture.h"
#include "glErrorChecker.h"
Include dependency graph for Terrain_texture.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Terrain_Texture](#)

7.130 Terrain_texture.h

[Go to the documentation of this file.](#)

```

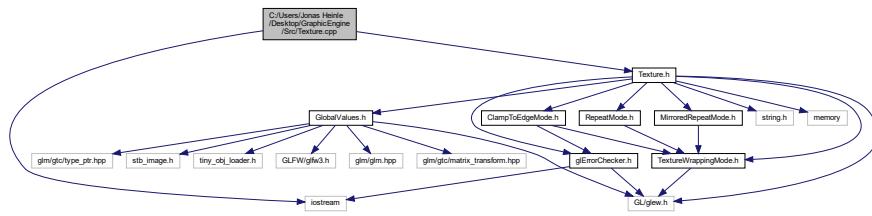
1 #pragma once
2
3 #include <string>
4 #include "GeometryPassShaderProgram.h"
5
6 #include "Texture.h"
7 #include "glErrorChecker.h"
8
9 class Terrain_Texture {
10
11 public:
12     Terrain_Texture();
13
14     enum BIOMETYPE
15     {
16         DEEP_WATER,
17         SHALLOW_WATER,
18         SAND,
19         LIGHT_GRASS,
20         DARK_GRASS,
21         LIGHT_ROCK,
22         DARK_ROCK,
23         SNOW
24     };
25
26     bool load_texture(char * file_location, BIOMETYPE type);
27
28     void retreive_uniform_locations(std::shared_ptr<GeometryPassShaderProgram> shader_program);
29
30     void use_texture();
31     void unbind_texture();
32     void setHeights(std::vector<float> heights, float max_height);
33     void clear_texture_context();
34     bool load_all_texture();
35
36
37     ~Terrain_Texture();
38 private:
39     GLuint deep_water_texture_id;
40     GLuint water_texture_id;
41     GLuint sand_texture_id;
42     GLuint light_gras_texture_id;
43     GLuint dark_gras_texture_id;
44     GLuint light_rock_texture_id;
45     GLuint dark_rock_texture_id;
46     GLuint snow_texture_id;
47     /* gather all texture ids here ? -> GL_TEXTURE"X" enum later
48     |
49     |
50     V */ //GLuint texture_ids[8];
51
52
53
54     GLuint isTerrainValue_id;
55     GLuint biomHeight_id;
56     float biomHeights[8];
57     GLuint max_height_id;
58     float max_height;
59
60     int width[8], height[8], bit_depth[8];
61
62     std::shared_ptr<TextureWrappingMode> wrapping_mode;
63
64     char* file_location;
65     glErrorChecker glErrorChecker_ins;
66
67     // Hard coded Program ID
68     GLuint program_id;
69
70 };

```

7.131 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.cpp File Reference

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

Include dependency graph for Texture.cpp:

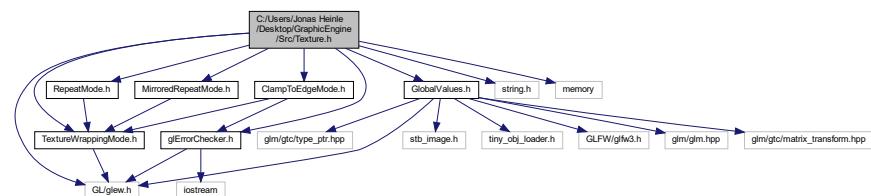


7.132 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.h File Reference

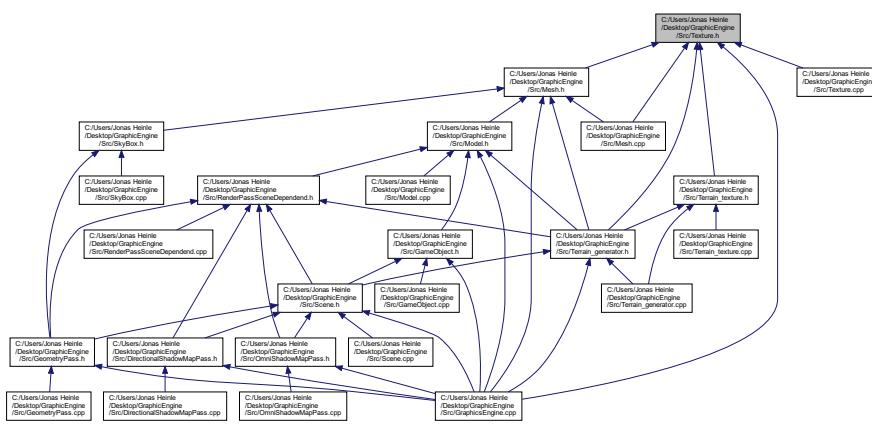
```

#include <GL/glew.h>
#include <string.h>
#include "TextureWrappingMode.h"
#include "RepeatMode.h"
#include "MirroredRepeatMode.h"
#include "ClampToEdgeMode.h"
#include <memory>
#include "GlobalValues.h"
#include "glErrorChecker.h"
  
```

Include dependency graph for Texture.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [Texture](#)

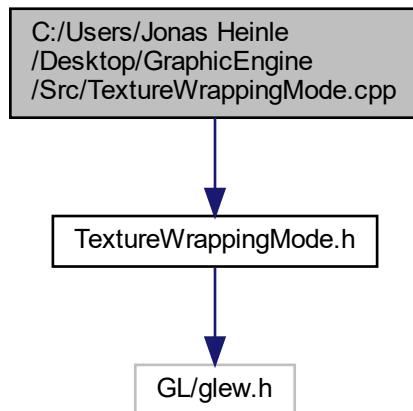
7.133 Texture.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <GL/glew.h>
3
4 #include <string.h>
5 #include "TextureWrappingMode.h"
6 #include "RepeatMode.h"
7 #include "MirroredRepeatMode.h"
8 #include "ClampToEdgeMode.h"
9
10 #include <memory>
11
12 #include "GlobalValues.h"
13
14 #include "glErrorChecker.h"
15
16
17 class Texture
18 {
19 public:
20
21     Texture();
22     Texture(const char* file_loc, std::shared_ptr<TextureWrappingMode> wrapping_mode);
23
24     bool load_texture_without_alpha_channel();
25     bool load_texture_with_alpha_channel();
26     std::string get_filename();
27
28     void use_texture();
29     void unbind_texture();
30     void clear_texture_context();
31     GLuint get_id();
32
33     ~Texture();
34
35 private:
36
37     GLuint textureID;
38     int width, height, bit_depth;
39
40     std::shared_ptr<TextureWrappingMode> wrapping_mode;
41
42     std::string file_location;
43     //const char* file_location; // // Git merge conflict
44
45     // this is for checking GL errors.
46     glErrorChecker glErrorChecker_ins;
47 };
48
```

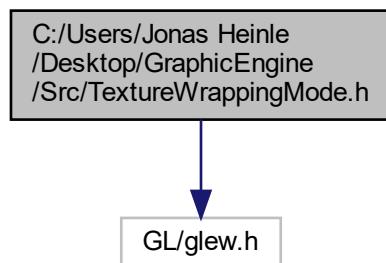
7.134 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.cpp File Reference

```
#include "TextureWrappingMode.h"  
Include dependency graph for TextureWrappingMode.cpp:
```

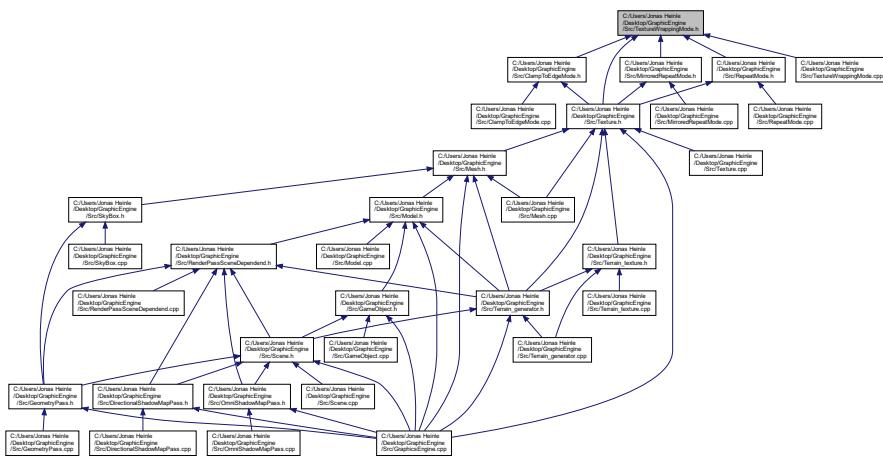


7.135 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.h File Reference

```
#include <GL/glew.h>  
Include dependency graph for TextureWrappingMode.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `TextureWrappingMode`

7.136 TextureWrappingMode.h

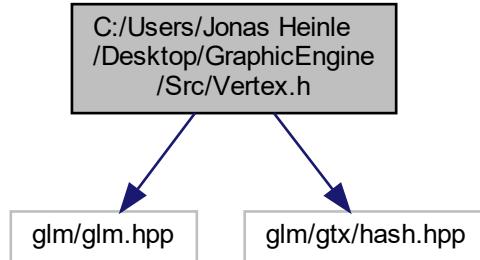
[Go to the documentation of this file.](#)

```
1 #pragma once
2 //lets mimic and c++-style interface :
3 #include <GL/glew.h>
4
5 class TextureWrappingMode
6 {
7 public:
8     TextureWrappingMode();
9
10    virtual void activate() = 0;
11
12    ~TextureWrappingMode();
13
14 private:
15 };
16
```

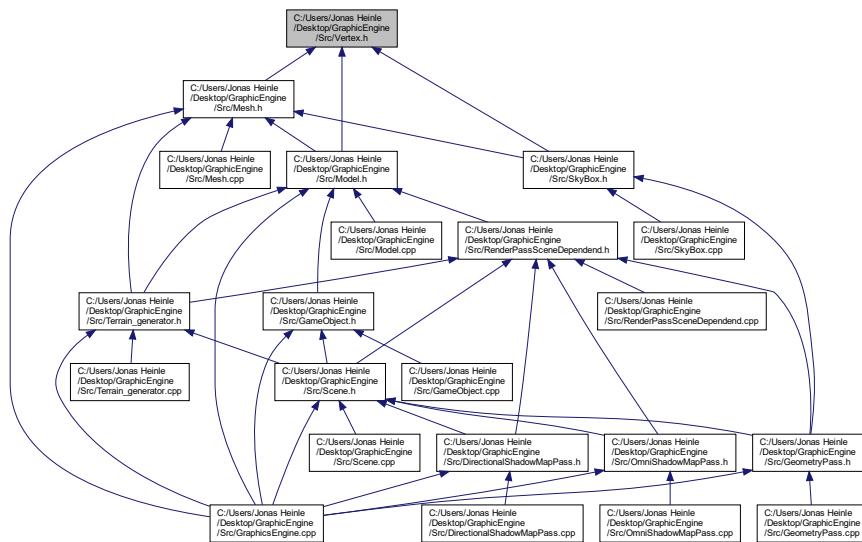
7.137 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Vertex.h File Reference

```
#include <glm/glm.hpp>
#include <glm/gtx/hash.hpp>
```

Include dependency graph for Vertex.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [Vertex](#)
- struct [std::hash< Vertex >](#)

Namespaces

- namespace [std](#)

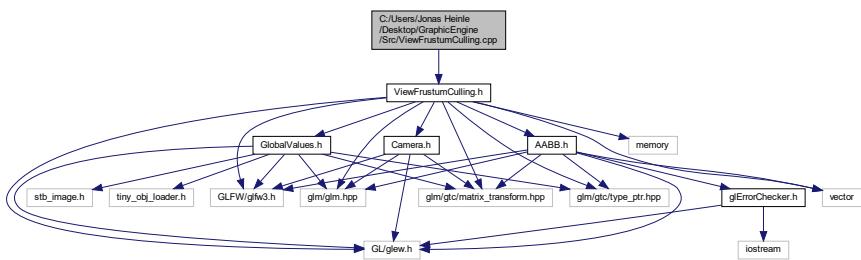
7.138 Vertex.h

[Go to the documentation of this file.](#)

```
1 #pragma once
2 #include <glm/glm.hpp>
3 #include <glm/gtx/hash.hpp>
4
5 struct Vertex
6 {
7     public:
8
9     // this is public because to access the size of pos, norm, tex Cood.
10    glm::vec3 position;
11    glm::vec3 normal;
12    glm::vec2 texture_coords;
13
14    Vertex() {
15        this->position = glm::vec3(0);
16        this->normal = glm::vec3(0);
17        this->texture_coords = glm::vec2(0);
18
19    }
20
21
22    Vertex(glm::vec3 pos, glm::vec3 normal, glm::vec2 texture_coords) {
23        this->position = pos;
24        this->normal = normal;
25        this->texture_coords = texture_coords;
26
27    };
28
29    glm::vec3 get_position() {
30        return position;
31    }
32
33    glm::vec3 get_normal() {
34        return normal;
35    }
36
37    glm::vec2 get_tex_coors() {
38        return texture_coords;
39    }
40
41    bool operator==(const Vertex& other) const {
42
43        return position == other.position
44            && normal == other.normal
45            && texture_coords == other.texture_coords;
46
47    }
48
49 };
50
51 namespace std {
52
53     template<> struct hash<Vertex> {
54
55         size_t operator()(Vertex const& vert) const {
56
57             size_t h1 = hash<glm::vec3>()(vert.position);
58             size_t h2 = hash<glm::vec3>()(vert.normal);
59             size_t h3 = hash<glm::vec2>()(vert.texture_coords);
60
61             // combine hashed wonderfully :)))
62             return ((h1 ^ (h2 << 1)) >> 1) ^ h3;
63
64     };
65 }
```

7.139 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.cpp File Reference

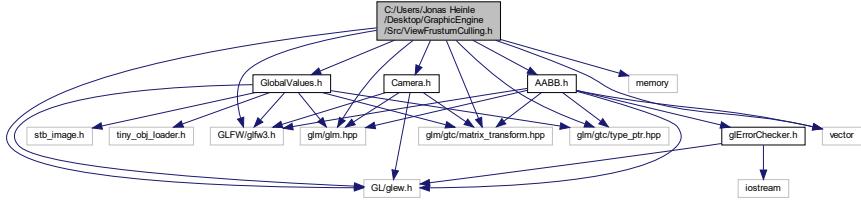
```
#include "ViewFrustumCulling.h"
Include dependency graph for ViewFrustumCulling.cpp:
```



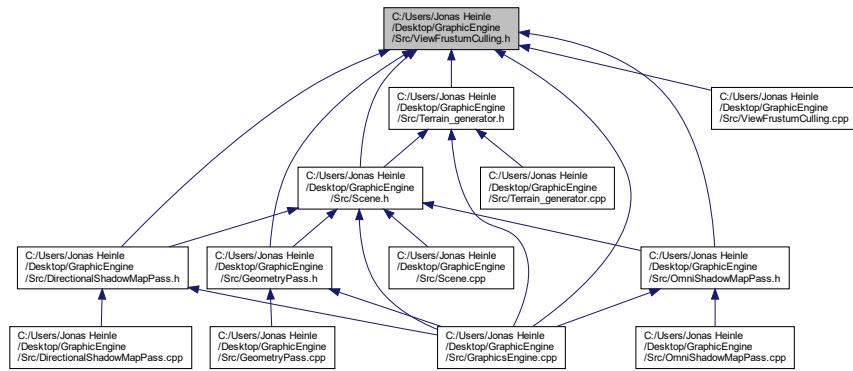
7.140 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.h File Reference

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>
#include <vector>
#include "Camera.h"
#include "AABB.h"
#include "GlobalValues.h"
#include <memory>
```

Include dependency graph for ViewFrustumCulling.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [ViewFrustumCulling](#)

7.141 ViewFrustumCulling.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <GL/glew.h>
3 #include <GLFW/glfw3.h>
4 #include <glm/glm.hpp>
5 #include <glm/gtc/matrix_transform.hpp>
6 #include <glm/gtc/type_ptr.hpp>
7 #include <vector>
8 #include "Camera.h"
9 #include "AABB.h"
10 #include "GlobalValues.h"
11 #include <memory>
12
13 class ViewFrustumCulling
14 {
15 public:
16
17     ViewFrustumCulling();
18
19     bool is_inside(GLfloat ratio, std::shared_ptr<Camera> main_camera, std::shared_ptr<AABB>
20                 bounding_box,
21                               glm::mat4 model);
22
23     void render_view_frustum();
24
25     ~ViewFrustumCulling();
26
27 private:
28
29     //render view frustum
30     unsigned int VBO, VAO, EBO;
31
32     unsigned int m_drawCount;
33
34     //we get that as input
35     GLfloat near_plane, far_plane, fov, ratio;
36
37     //calculate as soon as we become params
38     GLfloat tan, near_height, near_width, far_height, far_width;
39
40     std::shared_ptr<Camera> main_camera;
41
42     glm::vec3 dir, near_center, far_center;
43
44     //all corners of the frustum
  
```

```
44     //near plane
45     glm::vec3 near_top_left, near_top_right, near_bottom_left, near_bottom_right;
46     //far plane
47     glm::vec3 far_top_left, far_top_right, far_bottom_left, far_bottom_right;
48
49     //planes in Hesse normal form
50     //layout: [0]: near plane, [1] far plane, [2] front, [3] bottom, [4]: left, [5]: right
51     struct frustum_plane {
52
53         glm::vec3 normal;
54         glm::vec3 position;
55
56     };
57
58     frustum_plane frustum_planes[NUM_FRUSTUM_PLANES];
59
60     bool corners_outside_plane(std::vector<glm::vec3> aabb_corners, frustum_plane plane, GLuint
61     outcode_pattern);
62
63     GLfloat plane_point_distance(frustum_plane plane, glm::vec3 corner);
64
65     void update_frustum_param(GLfloat near_plane, GLfloat far_plane, GLfloat fov, GLfloat ratio,
66     std::shared_ptr<Camera> main_camera);
67
68     void init(std::vector<glm::vec3> frustum_corner);
69
70 };
```

Index

- ~AABB
 - AABB, 12
- ~Camera
 - Camera, 15
- ~CascadedShadowMap
 - CascadedShadowMap, 19
- ~ClampToEdgeMode
 - ClampToEdgeMode, 24
- ~Clouds
 - Clouds, 26
- ~CloudsShaderProgram
 - CloudsShaderProgram, 33
- ~ComputeShaderProgram
 - ComputeShaderProgram, 37
- ~DirectionalLight
 - DirectionalLight, 42
- ~DirectionalShadowMapPass
 - DirectionalShadowMapPass, 49
- ~GBuffer
 - GBuffer, 56
- ~GameObject
 - GameObject, 51
- ~GeometryPass
 - GeometryPass, 59
- ~GeometryPassShaderProgram
 - GeometryPassShaderProgram, 63
- ~Light
 - Light, 71
- ~LightingPass
 - LightingPass, 73
- ~LightingPassShaderProgram
 - LightingPassShaderProgram, 77
- ~LoadingScreenShaderProgram
 - LoadingScreenShaderProgram, 88
- ~Material
 - Material, 89
- ~Mesh
 - Mesh, 91
- ~MirroredRepeatMode
 - MirroredRepeatMode, 94
- ~Model
 - Model, 96
- ~MyWindow
 - MyWindow, 99
- ~Noise
 - Noise, 102
- ~OmniDirShadowMap
 - OmniDirShadowMap, 106
- ~OmniDirShadowShaderProgram
 - OmniDirShadowShaderProgram, 110
- ~OmniShadowMapPass
 - OmniShadowMapPass, 114
- ~PointLight
 - PointLight, 117
- ~Quad
 - Quad, 120
- ~RenderPassSceneDependend
 - RenderPassSceneDependend, 122
- ~RepeatMode
 - RepeatMode, 125
- ~Scene
 - Scene, 128
- ~ShaderProgram
 - ShaderProgram, 133
- ~ShadowMap
 - ShadowMap, 143
- ~ShadowMapShaderProgram
 - ShadowMapShaderProgram, 148
- ~SkyBox
 - SkyBox, 150
- ~SkyBoxShaderProgram
 - SkyBoxShaderProgram, 153
- ~Terrain_Generator
 - Terrain_Generator, 155
- ~Terrain_Texture
 - Terrain_Texture, 170
- ~Texture
 - Texture, 176
- ~TextureWrappingMode
 - TextureWrappingMode, 179
- ~ViewFrustumCulling
 - ViewFrustumCulling, 183
- AABB, 11
 - ~AABB, 12
 - AABB, 11
 - get_corners, 12
 - init, 12
 - render, 13
- aabbs
 - Terrain_Generator, 165
- activate
 - ClampToEdgeMode, 24
 - MirroredRepeatMode, 94
 - RepeatMode, 125
 - TextureWrappingMode, 179
- add_ambient_object
 - Scene, 128
- add_shader

ShaderProgram, 133
 add_space_ship
 Scene, 128
 ambient_intensity
 Light, 72
 angle
 Terrain_Model, 168
 areErrorPrintAll
 glErrorChecker, 67
 arePreError
 glErrorChecker, 68
 available_shadow_map_resolutions
 GraphicsEngine.cpp, 224
 axis
 Rotation, 126

 BIOMETYPE
 Terrain_Texture, 170
 BUSH
 Terrain_Model, 168

 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.cpp, 185
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/AABB.h, 185, 186
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.cpp, 187
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Camera.h, 187, 188
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.cpp, 189
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CascadedShadowMap.h, 189, 190
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.cpp, 191
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ClampToEdgeMode.h, 191, 192
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.cpp, 193
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Clouds.h, 193, 194
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.cpp, 196
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/CloudsShaderProgram.h, 196, 197
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.cpp, 198
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ComputeShaderProgram.h, 198, 199
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.cpp, 200
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLight.h, 200, 201
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalLightUniformLocations.h, 202
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.cpp, 203
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/DirectionalShadowMapPass.h, 203, 204

 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.cpp, 205
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GameObject.h, 205, 206
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.cpp, 206
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GBuffer.h, 207
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.cpp, 208
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPass.h, 208, 209
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShader, 210
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GeometryPassShader, 211, 212
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/glErrorChecker.h, 212, 213
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GlobalValues.h, 214, 217
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/GraphicsEngine.cpp, 217
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.cpp, 231
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Light.h, 231, 232
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.cpp, 233
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPass.h, 233, 234
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/LightingPassShaderProgram, 235
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram, 238
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src>LoadingScreenShaderProgram.h, 238
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.cpp, 239
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Material.h, 240
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.cpp, 242
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Mesh.h, 242, 243
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatMode.h, 244
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MirroredRepeatModeProgram, 245, 246
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.cpp, 246
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Model.h, 246, 247
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.cpp, 248
 C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/MyWindow.h, 249

C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.cpp
251
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Noise.h
251, 252
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.h
253
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.cpp
254, 255
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.h
255
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.cpp
256, 257
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.h
257
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/OmniDCTextureMap.cpp
257, 258
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.cpp
259
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Perlin.h
259, 262
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.cpp
264
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/PointLight.h
264, 265
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.cpp
266
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Quad.h
266, 267
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPass.h
268
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RenderPass.h
268, 269
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.h
270
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/RepeatMode.h
271, 272
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource.h
272
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/resource1.h
273
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.cpp
273
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Scene.h
273, 274
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.h
276
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShaderProgram.h
276, 277
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapper.h
278
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapper.h
278, 279
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapper.h
280
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ShadowMapper.h
281
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.cpp
282
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBox.h
282, 283
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.h
284
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/SkyBoxShaderProgram.h
285, 286
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.cpp
286
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_generator.h
287, 288
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_Model.h
290
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.cpp
291
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Terrain_texture.h
292
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.cpp
293
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/Texture.h
294, 295
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/TextureWrappingMode.h
296
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.cpp
300
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.h
300, 301
C:/Users/Jonas Heinle/Desktop/GraphicEngine/Src/ViewFrustumCulling.h
300, 301
DirectionalLight, 43
Terrain_Generator, 156
DirectionalLight, 43
Terrain_Generator, 156
DirectionalLight, 43
calculate_viewmatrix
Camera, 14
Camera, 14
calculate_viewmatrix, 15
Camera, 15
get_camera_direction, 15
Camera, 15
get_far_plane, 16
get_far_plane, 16
get_near_plane, 16
get_near_plane, 16
get_up_axis, 16
get_up_axis, 16
key_control, 16
set_camera_position, 17
set_camera_position, 17
set_fov, 17
set_fov, 17
set_near_plane, 17
set_near_plane, 17
cascaded shadow intensity

GraphicsEngine.cpp, 224
 CascadedShadowMap, 18
 ~CascadedShadowMap, 19
 CascadedShadowMap, 19
 FBO, 21
 get_id, 19
 get_intensity, 19
 get_num_active_cascades, 19
 get_pcf_radius, 20
 get_shadow_height, 20
 get_shadow_width, 20
 glErrorChecker_ins, 21
 init, 20
 intensity, 21
 num_active_cascades, 21
 pcf_radius, 22
 read, 20
 set_intensity, 21
 set_pcf_radius, 21
 shadow_height, 22
 shadow_map, 22
 shadow_width, 22
 write, 21
 changeMaxHeight
 Terrain_Generator, 156
 changeVertex
 Mesh, 91
 chooseen_space_ship
 GraphicsEngine.cpp, 225
 ClampToEdgeMode, 23
 ~ClampToEdgeMode, 24
 activate, 24
 ClampToEdgeMode, 24
 clear_shader_program
 ShaderProgram, 134
 clear_texture_context
 Terrain_Texture, 171
 Texture, 176
 cloud_cirrus_effect
 GraphicsEngine.cpp, 225
 cloud_density
 GraphicsEngine.cpp, 225
 cloud_mesh_scale
 GraphicsEngine.cpp, 225
 cloud_movement_direction
 GraphicsEngine.cpp, 225
 cloud_pillowness
 GraphicsEngine.cpp, 225
 cloud_powder_effect
 GraphicsEngine.cpp, 225
 cloud_scale
 GraphicsEngine.cpp, 225
 cloud_speed
 GraphicsEngine.cpp, 226
 Clouds, 25
 ~Clouds, 26
 Clouds, 26
 get_cirrus_effect, 26
 get_density, 26
 get_mesh_scale, 26
 get_model, 27
 get_movement_direction, 27
 get_movement_speed, 27
 get_normal_model, 27
 get_pillowness, 27
 get_powder_effect, 27
 get_rad, 27
 get_scale, 27
 get_shader_program, 28
 init, 28
 read, 28
 render, 28
 set_cirrus_effect, 29
 set_density, 29
 set_movement_direction, 29
 set_movement_speed, 29
 set_pillowness, 29
 set_powder_effect, 30
 set_scale, 30
 set_translation, 30
 update_window_params, 30
 clouds
 GraphicsEngine.cpp, 226
 CloudsShaderProgram, 31
 ~CloudsShaderProgram, 33
 CloudsShaderProgram, 33
 get_model_location, 33
 get_projection_location, 33
 get_view_location, 33
 retrieve_uniform_locations, 33
 uniform_model_location, 34
 uniform_projection_location, 34
 uniform_view_location, 34
 color
 Light, 72
 COMMONVALS
 GlobalValues.h, 215
 compile_compute_shader_program
 ShaderProgram, 134
 compile_program
 ShaderProgram, 135
 compile_shader_program
 ShaderProgram, 135, 136
 compute_location
 ShaderProgram, 140
 ComputeShaderProgram, 35
 ~ComputeShaderProgram, 37
 ComputeShaderProgram, 37
 get_cell_location, 37
 get_num_cell_location, 37
 reload, 38
 retrieve_uniform_locations, 38
 set_noise, 39
 constant
 PointLight, 118
 create

GBuffer, 56
 create_computer_shader_program_from_file
 ShaderProgram, 136
 create_from_files
 ShaderProgram, 137, 138
 create_geometry_pass_shader_program
 GraphicsEngine.cpp, 220
 create_grad_noise
 Noise, 102
 create_lighting_pass_shader_program
 GraphicsEngine.cpp, 220
 create_loading_screen_shader_program
 GraphicsEngine.cpp, 220
 create_noise_textures
 GraphicsEngine.cpp, 221
 create_omni_shadow_map_shader_program
 GraphicsEngine.cpp, 221
 create_render_context
 Model, 96
 create_shader_programs
 GraphicsEngine.cpp, 221
 create_shadow_map_shader_program
 GraphicsEngine.cpp, 222
 create_worley_noise
 Noise, 102

DARK_GRASS
 Terrain_Texture, 170

DARK_ROCK
 Terrain_Texture, 170

DEEP_WATER
 Terrain_Texture, 170

degrees
 Rotation, 126

delta_time
 GraphicsEngine.cpp, 226

diffuse_intensity
 Light, 72

directional_light_starting_color
 GraphicsEngine.cpp, 226

directional_light_starting_position
 GraphicsEngine.cpp, 226

directional_shadow_map_pass
 GraphicsEngine.cpp, 226

DirectionalLight, 39
 ~DirectionalLight, 42
 calc_orthogonal_projections, 43
 calculate_light_transform, 43
 DirectionalLight, 42
 get_ambient_intensity, 43
 get_cascaded_light_matrices, 43
 get_cascaded_slots, 43
 get_color, 44
 get_diffuse_intensity, 44
 get_direction, 44
 get_light_view_matrix, 44
 get_shadow_map, 44
 set_ambient_intensity, 44
 set_color, 45

 set_diffuse_intensity, 45
 set_direction, 45
 update_shadow_map, 45

DirectionalLightUniformLocations, 46
 uniform_ambient_intensity_location, 46
 uniform_color_location, 46
 uniform_diffuse_intensity_location, 46
 uniform_direction_location, 47
 uniform_shadow_intensity_location, 47

DirectionalShadowMapPass, 47
 ~DirectionalShadowMapPass, 49
 DirectionalShadowMapPass, 48
 execute, 49
 set_game_object_uniforms, 49
 use_terrain_textures, 49

draw_sky_box
 SkyBox, 150

execute
 DirectionalShadowMapPass, 49
 GeometryPass, 59
 LightingPass, 73
 OmniShadowMapPass, 114

expand
 Mesh, 92

expandTerrain
 Terrain_Generator, 156

exponent
 PointLight, 119

fade
 Perlin.h, 260

far_plane
 GraphicsEngine.cpp, 226
 PointLight, 119

far_plane_shadow
 GraphicsEngine.cpp, 226

FBO
 CascadedShadowMap, 21
 ShadowMap, 145

fov
 GraphicsEngine.cpp, 227

fragment_location
 ShaderProgram, 140

g_buffer_geometry_pass_shader_program
 GraphicsEngine.cpp, 227

g_buffer_lighting_pass_shader_program
 GraphicsEngine.cpp, 227

G_BUFFER_SIZE
 GlobalValues.h, 216

GameObject, 50
 ~GameObject, 51
 GameObject, 50, 51
 get_aabb, 51
 get_material_id, 51
 get_model, 51
 get_normal_world_trafo, 52
 get_world_trafo, 52

init, 52
 render, 53
 rotate, 53
 scale, 53
 set_material_id, 53
 translate, 54
GBuffer, 55
 ~GBuffer, 56
 create, 56
 GBuffer, 55
 get_id, 56
 read, 56
 update_window_params, 56
 use_gbuffer, 56
gbuffer
 GraphicsEngine.cpp, 227
generate biome
 Terrain_Generator, 156
generate_indices
 Terrain_Generator, 157
generate_map_chunk
 Terrain_Generator, 157
generate_noise_map
 Terrain_Generator, 158
generate_normals
 Terrain_Generator, 159
generate_render_context
 Terrain_Generator, 159
generate_vertices
 Terrain_Generator, 160
generateMesh
 Terrain_Generator, 160
geometry_location
 ShaderProgram, 140
geometry_pass
 GraphicsEngine.cpp, 227
GeometryPass, 57
 ~GeometryPass, 59
 execute, 59
 GeometryPass, 58
 set_game_object_uniforms, 59
 use_terrain_textures, 59
GeometryPassShaderProgram, 60
 ~GeometryPassShaderProgram, 63
 GeometryPassShaderProgram, 63
 get_biom_height_id, 63
 get_biom_texture_location, 63
 get_material_id_location, 64
 get_max_height_id, 64
 get_model_location, 64
 get_normal_modal_location, 64
 get_program_id, 64
 get_projection_location, 64
 get_terrain_id, 64
 get_view_location, 65
 retrieve_uniform_locations, 65
 uniform_biom_texture_locations, 65
 uniform_biomHeight_id, 65
 uniform_isTerrainValue_id, 65
 uniform_material_id_location, 66
 uniform_max_height_id, 66
 uniform_model_location, 66
 uniform_normal_model_location, 66
 uniform_projection_location, 66
 uniform_view_location, 66
get_aabb
 GameObject, 51
 Model, 96
get_ambient_intensity
 DirectionalLight, 43
get_base_dir
 Model, 96
get_biom_height_id
 GeometryPassShaderProgram, 63
get_biom_texture_location
 GeometryPassShaderProgram, 63
get_buffer_height
 MyWindow, 99
get_buffer_width
 MyWindow, 99
get_camera_direction
 Camera, 15
get_camera_position
 Camera, 15
get_cascade_endpoint_location
 LightingPassShaderProgram, 78
get_cascaded_light_matrices
 DirectionalLight, 43
get_cascaded_slots
 DirectionalLight, 43
get_cell_location
 ComputeShaderProgram, 37
get_cirrus_effect
 Clouds, 26
get_cirrus_effect_location
 LightingPassShaderProgram, 78
get_cloud_powderness_effect
 LightingPassShaderProgram, 78
get_clouds
 Scene, 128
get_color
 DirectionalLight, 44
get_context_setup
 Scene, 129
get_corners
 AABB, 12
get_density
 Clouds, 26
get_diffuse_intensity
 DirectionalLight, 44
get_direction
 DirectionalLight, 44
get_directional_light_ambient_intensity_location
 LightingPassShaderProgram, 78
get_directional_light_color_location
 LightingPassShaderProgram, 78

get_directional_light_diffuse_intensity_location
 LightingPassShaderProgram, 78

get_directional_light_direction_location
 LightingPassShaderProgram, 78

get_directional_light_shadow_intensity_location
 LightingPassShaderProgram, 78

get_directional_light_transform_location
 LightingPassShaderProgram, 79

 ShadowMapShaderProgram, 148

get_directional_shadow_map_location
 LightingPassShaderProgram, 79

get_eye_position_location
 LightingPassShaderProgram, 79

get_far_plane
 Camera, 16

 PointLight, 118

get_far_plane_location
 OmniDirShadowShaderProgram, 110

get_filename
 Texture, 176

get_fov
 Camera, 16

get_g_albedo_location
 LightingPassShaderProgram, 79

get_g_clouds_location
 LightingPassShaderProgram, 79

get_g_directional_light_position_location
 LightingPassShaderProgram, 79

get_g_frag_depth_location
 LightingPassShaderProgram, 79

get_g_normal_location
 LightingPassShaderProgram, 80

get_g_position_location
 LightingPassShaderProgram, 80

get_id
 CascadedShadowMap, 19

 GBuffer, 56

 ShadowMap, 144

 Texture, 177

get_intensity
 CascadedShadowMap, 19

get_keys
 MyWindow, 99

get_light_view_matrix
 DirectionalLight, 44

get_material_id
 GameObject, 51

 Terrain_Generator, 160

get_material_id_location
 GeometryPassShaderProgram, 64

get_max_height_id
 GeometryPassShaderProgram, 64

get_mesh_scale
 Clouds, 26

get_model
 Clouds, 27

 GameObject, 51

get_model_location

CloudsShaderProgram, 33

GeometryPassShaderProgram, 64

OmniDirShadowShaderProgram, 110

ShadowMapShaderProgram, 148

get_movement_direction
 Clouds, 27

get_movement_speed
 Clouds, 27

get_near_plane
 Camera, 16

get_normal
 Vertex, 181

get_normal_modal_location
 GeometryPassShaderProgram, 64

get_normal_model
 Clouds, 27

get_normal_world_trafo
 GameObject, 52

 Terrain_Generator, 161

get_num_active_cascades
 CascadedShadowMap, 19

get_num_cell_location
 ComputeShaderProgram, 37

get_omni_light_pos_location
 OmniDirShadowShaderProgram, 110

get_omni_shadow_map
 PointLight, 118

get_pcf_radius
 CascadedShadowMap, 20

get_pillowness
 Clouds, 27

get_position
 PointLight, 118

 Vertex, 181

get_position_of_current_ship
 Scene, 129

get_powder_effect
 Clouds, 27

get_program_id
 GeometryPassShaderProgram, 64

get_progress
 Scene, 129

get_projection_location
 CloudsShaderProgram, 33

 GeometryPassShaderProgram, 64

 SkyBoxShaderProgram, 153

get_rad
 Clouds, 27

get_random_number_location
 LightingPassShaderProgram, 80

get_right_axis
 Camera, 16

get_scale
 Clouds, 27

get_shader_program
 Clouds, 28

get_shadow_height
 CascadedShadowMap, 20

ShadowMap, 144
 get_shadow_map
 DirectionalLight, 44
 get_shadow_width
 CascadedShadowMap, 20
 ShadowMap, 144
 get_should_close
 MyWindow, 99
 get_terrain_generator
 Scene, 129
 get_terrain_id
 GeometryPassShaderProgram, 64
 get_tex_coors
 Vertex, 181
 get_textures
 Terrain_Generator, 161
 get_uniform_absorption_location
 LightingPassShaderProgram, 80
 get_uniform_cloud_model
 LightingPassShaderProgram, 80
 get_uniform_cloud_offset
 LightingPassShaderProgram, 80
 get_uniform_cloud_rad_location
 LightingPassShaderProgram, 80
 get_uniform_cloud_scale_location
 LightingPassShaderProgram, 81
 get_uniform_cloud_threshold_location
 LightingPassShaderProgram, 81
 get_uniform_IOR_location
 LightingPassShaderProgram, 81
 get_uniform_material_id_location
 LightingPassShaderProgram, 81
 get_uniform_material_metallic_location
 LightingPassShaderProgram, 81
 get_uniform_material_roughness_location
 LightingPassShaderProgram, 81
 get_uniform_num_active_cascades_location
 LightingPassShaderProgram, 81
 get_uniform_omni_dir_shadow_map_location
 LightingPassShaderProgram, 82
 get_uniform_pcf_radius_location
 LightingPassShaderProgram, 82
 get_uniform_pillowness_location
 LightingPassShaderProgram, 82
 get_uniform_point_light_far_plane_location
 LightingPassShaderProgram, 82
 get_up_axis
 Camera, 16
 get_view_location
 CloudsShaderProgram, 33
 GeometryPassShaderProgram, 65
 SkyBoxShaderProgram, 153
 get_window
 MyWindow, 99
 get_world_trafo
 GameObject, 52
 Terrain_Generator, 161
 get_x_change
 MyWindow, 100
 get_y_change
 MyWindow, 100
 get_yaw
 Camera, 16
 getIndices
 Mesh, 92
 getNoiseParameters
 Terrain_Generator, 161
 getPermutationVector
 Perlin.h, 260
 getVertices
 Mesh, 92
 glErrorChecker, 67
 areErrorPrintAll, 67
 arePreError, 68
 glErrorChecker_ins
 CascadedShadowMap, 21
 ShaderProgram, 140
 GlobalValues.h
 COMMONVALS, 215
 G_BUFFER_SIZE, 216
 MAX_MATERIALS, 216
 MAX_POINT_LIGHTS, 216
 MAX_RESOLUTION_X, 215
 MAX_RESOLUTION_Y, 215
 NUM BIOM_TEXTURES, 215
 NUM CELLS, 216
 NUM CLOUDS, 216
 NUM_FRUSTUM_PLANES, 216
 NUM_MAX_CASCADES, 215
 NUM_MIN_CASCADES, 215
 NUM_NOISE_TEXTURES, 216
 grad
 Perlin.h, 261
 GraphicsEngine.cpp
 available_shadow_map_resolutions, 224
 cascaded_shadow_intensity, 224
 choose_space_ship, 225
 cloud_cirrus_effect, 225
 cloud_density, 225
 cloud_mesh_scale, 225
 cloud_movement_direction, 225
 cloud_pillowness, 225
 cloud_powder_effect, 225
 cloud_scale, 225
 cloud_speed, 226
 clouds, 226
 create_geometry_pass_shader_program, 220
 create_lighting_pass_shader_program, 220
 create_loading_screen_shader_program, 220
 create_noise_textures, 221
 create_omni_shadow_map_shader_program, 221
 create_shader_programs, 221
 create_shadow_map_shader_program, 222
 delta_time, 226
 directional_light_starting_color, 226
 directional_light_starting_position, 226

directional_shadow_map_pass, 226
far_plane, 226
far_plane_shadow, 226
fov, 227
g_buffer_geometry_pass_shader_program, 227
g_buffer_lighting_pass_shader_program, 227
gbuffer, 227
geometry_pass, 227
last_time, 227
lighting_pass, 227
loading_screen, 227
loading_screen_finished, 228
loading_screen_shader_program, 228
loading_screen_tex, 228
logo, 228
main, 222
main_camera, 228
main_light, 228
material_counter, 228
materials, 223
near_plane, 228
noise, 229
num_shadow_cascades, 229
omni_dir_shadow_shader_program, 229
omni_shadow_map_pass, 229
ornament1, 229
pcf_radius, 229
point_light_count, 229
point_lights, 223
reload_noise_programs, 224
reload_shader_programs, 224
shadow_map_res_index, 229
shadow_map_resolution, 230
shadow_map_shader_program, 230
shadow_resolution_changed, 230
sound_volume, 230
ssao_enabled, 230
ssr_enabled, 230
STB_IMAGE_IMPLEMENTATION, 220
terrain_height, 230
tGenerator, 230
TINYOBJLOADER_IMPLEMENTATION, 220

height
 terrainType, 174

IDI_ICON1
 resource.h, 272

init
 AABB, 12
 CascadedShadowMap, 20
 Clouds, 28
 GameObject, 52
 LightingPass, 74
 Noise, 103
 OmniDirShadowMap, 107
 Quad, 120
 Scene, 129
 ShadowMap, 144

 Terrain_Generator, 161
initialize
 MyWindow, 100
intensity
 CascadedShadowMap, 21
is_inside
 ViewFrustumCulling, 183
is_loaded
 Scene, 129

key_control
 Camera, 16

last_time
 GraphicsEngine.cpp, 227

lerp
 Perlin.h, 261

Light, 70
 ~Light, 71
 ambient_intensity, 72
 color, 72
 diffuse_intensity, 72
 Light, 71
 light_proj, 72

LIGHT_GRASS
 Terrain_Texture, 170

light_proj
 Light, 72

LIGHT_ROCK
 Terrain_Texture, 170

lighting_pass
 GraphicsEngine.cpp, 227

LightingPass, 72
 ~LightingPass, 73
 execute, 73
 init, 74
 LightingPass, 73

LightingPassShaderProgram, 74
 ~LightingPassShaderProgram, 77
 get_cascade_endpoint_location, 78
 get_cirrus_effect_location, 78
 get_cloud_powderness_effect, 78
 get_directional_light_ambient_intensity_location,
 78
 get_directional_light_color_location, 78
 get_directional_light_diffuse_intensity_location, 78
 get_directional_light_direction_location, 78
 get_directional_light_shadow_intensity_location,
 78
 get_directional_light_transform_location, 79
 get_directional_shadow_map_location, 79
 get_eye_position_location, 79
 get_g_albedo_location, 79
 get_g_clouds_location, 79
 get_g_directional_light_position_location, 79
 get_g_frag_depth_location, 79
 get_g_normal_location, 80
 get_g_position_location, 80
 get_random_number_location, 80

get_uniform_absorption_location, 80
 get_uniform_cloud_model, 80
 get_uniform_cloud_offset, 80
 get_uniform_cloud_rad_location, 80
 get_uniform_cloud_scale_location, 81
 get_uniform_cloud_threshold_location, 81
 get_uniform_IOR_location, 81
 get_uniform_material_id_location, 81
 get_uniform_material_metallic_location, 81
 get_uniform_material_roughness_location, 81
 get_uniform_num_active_cascades_location, 81
 get_uniform_omni_dir_shadow_map_location, 82
 get_uniform_pcf_radius_location, 82
 get_uniform_pillowness_location, 82
 get_uniform_point_light_far_plane_location, 82
 LightingPassShaderProgram, 77
 set_cloud_texture, 82
 set_noise_textures, 82
 set_point_lights, 83
 uniform_absorption_coeff_location, 83
 uniform_ambient_intensity, 83
 uniform_color, 84
 uniform_constant, 84
 uniform_diffuse_intensity, 84
 uniform_exponent, 84
 uniform_far_plane, 84
 uniform_IOR_location, 84
 uniform_linear, 84
 uniform_metallic_location, 84
 uniform_position, 85
 uniform_roughness_location, 85
 uniform_shadow_map, 85
 linear
 PointLight, 119
 load_all_texture
 Terrain_Texture, 171
 load_model_in_ram
 Model, 96
 load_models
 Scene, 129
 load_plants
 Terrain_Generator, 162
 load_texture
 Terrain_Texture, 171
 load_texture_with_alpha_channel
 Texture, 177
 load_texture_without_alpha_channel
 Texture, 177
 loading_screen
 GraphicsEngine.cpp, 227
 loading_screen_finished
 GraphicsEngine.cpp, 228
 loading_screen_shader_program
 GraphicsEngine.cpp, 228
 loading_screen_tex
 GraphicsEngine.cpp, 228
 LoadingScreenShaderProgram, 85
 ~LoadingScreenShaderProgram, 88
 LoadingScreenShaderProgram, 88
 LoadingScreenShaderProgram, 88
 logo
 GraphicsEngine.cpp, 228
 M_TYPE
 Terrain_Model, 167
 main
 GraphicsEngine.cpp, 222
 main_camera
 GraphicsEngine.cpp, 228
 main_light
 GraphicsEngine.cpp, 228
 Material, 88
 ~Material, 89
 Material, 89
 use_material, 89
 material_counter
 GraphicsEngine.cpp, 228
 materials
 GraphicsEngine.cpp, 223
 MAX_MATERIALS
 GlobalValues.h, 216
 MAX_POINT_LIGHTS
 GlobalValues.h, 216
 MAX_RESOLUTION_X
 GlobalValues.h, 215
 MAX_RESOLUTION_Y
 GlobalValues.h, 215
 Mesh, 90
 ~Mesh, 91
 changeVertex, 91
 expand, 92
 getIndices, 92
 getVertices, 92
 Mesh, 90, 91
 render, 92
 transform_Mesh, 92
 meshes
 Terrain_Generator, 165
 MirroredRepeatMode, 93
 ~MirroredRepeatMode, 94
 activate, 94
 MirroredRepeatMode, 94
 Model, 95
 ~Model, 96
 create_render_context, 96
 get_aabb, 96
 get_base_dir, 96
 load_model_in_ram, 96
 Model, 95
 render, 97
 transform_model, 97
 mouse_control
 Camera, 17
 MyWindow, 98
 ~MyWindow, 99
 get_buffer_height, 99
 get_buffer_width, 99
 get_keys, 99

get_should_close, 99
get_window, 99
get_x_change, 100
get_y_change, 100
initialize, 100
MyWindow, 98, 99
set_buffer_size, 100
swap_buffers, 100
update_viewport, 100

near_plane
 GraphicsEngine.cpp, 228

Noise, 101
 ~Noise, 102
 create_grad_noise, 102
 create_worley_noise, 102
 init, 103
 Noise, 101
 read_grad_noise, 103
 read_worley_noise, 104
 set_num_cells, 104
 update, 104

noise
 GraphicsEngine.cpp, 229

normal
 Vertex, 181

num_active_cascades
 CascadedShadowMap, 21

NUM BIOM_TEXTURES
 GlobalValues.h, 215

NUM CELLS
 GlobalValues.h, 216

NUM CLOUDS
 GlobalValues.h, 216

NUM FRUSTUM_PLANES
 GlobalValues.h, 216

NUM MAX CASCADES
 GlobalValues.h, 215

NUM MIN CASCADES
 GlobalValues.h, 215

NUM NOISE_TEXTURES
 GlobalValues.h, 216

num_shadow_cascades
 GraphicsEngine.cpp, 229

omni_dir_shadow_map
 PointLight, 119

omni_dir_shadow_shader_program
 GraphicsEngine.cpp, 229

omni_shadow_map_pass
 GraphicsEngine.cpp, 229

OmniDirShadowMap, 105
 ~OmniDirShadowMap, 106
 init, 107
 OmniDirShadowMap, 106
 read, 107
 write, 107

OmniDirShadowShaderProgram, 108
 ~OmniDirShadowShaderProgram, 110

get_far_plane_location, 110
get_model_location, 110
get_omni_light_pos_location, 110
OmniDirShadowShaderProgram, 110
reload, 110
set_light_matrices, 111

OmniShadowMapPass, 111
 ~OmniShadowMapPass, 114
execute, 114
OmniShadowMapPass, 113
set_game_object_uniforms, 114
use_terrain_textures, 114

operator()
 std::hash< Vertex >, 70

operator==
 Vertex, 181

ornament1
 GraphicsEngine.cpp, 229

pcf_radius
 CascadedShadowMap, 22
 GraphicsEngine.cpp, 229

Perlin.h
 fade, 260
 getPermutationVector, 260
 grad, 261
 lerp, 261
 perlin_noise, 261

perlin_noise
 Perlin.h, 261

point_light_count
 GraphicsEngine.cpp, 229

point_lights
 GraphicsEngine.cpp, 223

PointLight, 115
 ~PointLight, 117
 calculate_light_transform, 117
 constant, 118
 exponent, 119
 far_plane, 119
 get_far_plane, 118
 get_omni_shadow_map, 118
 get_position, 118
 linear, 119
 omni_dir_shadow_map, 119
 PointLight, 117
 position, 119
 set_position, 118
 use_light, 118

pos
 Terrain_Model, 168

position
 PointLight, 119
 Vertex, 181

program_id
 ShaderProgram, 141

Quad, 120
 ~Quad, 120

init, 120
 Quad, 120
 render, 121

 read
 CascadedShadowMap, 20
 Clouds, 28
 GBuffer, 56
 OmniDirShadowMap, 107
 ShadowMap, 144
 read_file
 ShaderProgram, 138
 read_grad_noise
 Noise, 103
 read_worley_noise
 Noise, 104
 regenerate
 Terrain_Generator, 162
 reload
 ComputeShaderProgram, 38
 OmniDirShadowShaderProgram, 110
 ShaderProgram, 138
 SkyBox, 150
 reload_noise_programs
 GraphicsEngine.cpp, 224
 reload_shader_programs
 GraphicsEngine.cpp, 224
 render
 AABB, 13
 Clouds, 28
 GameObject, 53
 Mesh, 92
 Model, 97
 Quad, 121
 Scene, 130
 Terrain_Generator, 163
 render_plants
 Terrain_Generator, 163
 render_view_frustum
 ViewFrustumCulling, 183
 RenderPassSceneDependend, 121
 ~RenderPassSceneDependend, 122
 RenderPassSceneDependend, 122
 set_game_object_uniforms, 123
 use_terrain_textures, 123
 RepeatMode, 124
 ~RepeatMode, 125
 activate, 125
 RepeatMode, 125
 resource.h
 IDI_ICON1, 272
 retrieve_uniform_locations
 Terrain_Generator, 164
 Terrain_Texture, 172
 retrieve_uniform_locations
 CloudsShaderProgram, 33
 ComputeShaderProgram, 38
 GeometryPassShaderProgram, 65
 ShaderProgram, 139

 rotate
 GameObject, 53
 Rotation, 125
 axis, 126
 degrees, 126

 SAND
 Terrain_Texture, 170
 scale
 GameObject, 53
 Scene, 126
 ~Scene, 128
 add_ambient_object, 128
 add_space_ship, 128
 get_clouds, 128
 get_context_setup, 129
 get_position_of_current_ship, 129
 get_progress, 129
 get_terrain_generator, 129
 init, 129
 is_loaded, 129
 load_models, 129
 render, 130
 Scene, 128
 set_context_setup, 130
 setup_game_object_context, 130
 spwan, 130
 update_current_space_ship, 130
 set_ambient_intensity
 DirectionalLight, 44
 set_buffer_size
 MyWindow, 100
 set_camera_position
 Camera, 17
 set_cirrus_effect
 Clouds, 29
 set_cloud_texture
 LightingPassShaderProgram, 82
 set_color
 DirectionalLight, 45
 set_context_setup
 Scene, 130
 set_density
 Clouds, 29
 set_diffuse_intensity
 DirectionalLight, 45
 set_direction
 DirectionalLight, 45
 set_directional_light_transform
 ShadowMapShaderProgram, 148
 set_far_plane
 Camera, 17
 set_fov
 Camera, 17
 set_game_object_uniforms
 DirectionalShadowMapPass, 49
 GeometryPass, 59
 OmniShadowMapPass, 114
 RenderPassSceneDependend, 123

set_intensity
 CascadedShadowMap, 21

set_light_matrices
 OmniDirShadowShaderProgram, 111

set_material_id
 GameObject, 53
 Terrain_Generator, 164

set_movement_direction
 Clouds, 29

set_movement_speed
 Clouds, 29

set_near_plane
 Camera, 17

set_noise
 ComputeShaderProgram, 39

set_noise_textures
 LightingPassShaderProgram, 82

set_num_cells
 Noise, 104

set_pcf_radius
 CascadedShadowMap, 21

set_pillowness
 Clouds, 29

set_point_lights
 LightingPassShaderProgram, 83

set_position
 PointLight, 118

set_powder_effect
 Clouds, 30

set_scale
 Clouds, 30

set_translation
 Clouds, 30

set_world_trafo
 Terrain_Generator, 164

setHeights
 Terrain_Texture, 172

setNoiseParameters
 Terrain_Generator, 165

setup_game_object_context
 Scene, 130

ShaderProgram, 131
 ~ShaderProgram, 133
 add_shader, 133
 clear_shader_program, 134
 compile_compute_shader_program, 134
 compile_program, 135
 compile_shader_program, 135, 136
 compute_location, 140
 create_computer_shader_program_from_file, 136
 create_from_files, 137, 138
 fragment_location, 140
 geometry_location, 140
 glErrorChecker_ins, 140
 program_id, 141
 read_file, 138
 reload, 138
 retrieve_uniform_locations, 139

 ShaderProgram, 133
 use_shader_program, 139
 validate_program, 139
 vertex_location, 141

shadow_height
 CascadedShadowMap, 22
 ShadowMap, 145

shadow_map
 CascadedShadowMap, 22
 ShadowMap, 145

shadow_map_res_index
 GraphicsEngine.cpp, 229

shadow_map_resolution
 GraphicsEngine.cpp, 230

shadow_map_shader_program
 GraphicsEngine.cpp, 230

shadow_resolution_changed
 GraphicsEngine.cpp, 230

shadow_width
 CascadedShadowMap, 22
 ShadowMap, 145

ShadowMap, 141
 ~ShadowMap, 143
 FBO, 145
 get_id, 144
 get_shadow_height, 144
 get_shadow_width, 144
 init, 144
 read, 144
 shadow_height, 145
 shadow_map, 145
 shadow_width, 145
 ShadowMap, 143
 write, 144

ShadowMapShaderProgram, 146
 ~ShadowMapShaderProgram, 148
 get_directional_light_transform_location, 148
 get_model_location, 148
 set_directional_light_transform, 148
 ShadowMapShaderProgram, 148

SHALLOW_WATER
 Terrain_Texture, 170

SkyBox, 149
 ~SkyBox, 150
 draw_sky_box, 150
 reload, 150
 SkyBox, 149

SkyBoxShaderProgram, 151
 ~SkyBoxShaderProgram, 153
 get_projection_location, 153
 get_view_location, 153
 SkyBoxShaderProgram, 153

SNOW
 Terrain_Texture, 170

sound_volume
 GraphicsEngine.cpp, 230

spwan
 Scene, 130

ssao_enabled
 GraphicsEngine.cpp, 230
 ssr_enabled
 GraphicsEngine.cpp, 230
 STB_IMAGE_IMPLEMENTATION
 GraphicsEngine.cpp, 220
 std, 9
 std::hash< Vertex >, 69
 operator(), 70
 STONE
 Terrain_Model, 168
 swap_buffers
 MyWindow, 100

 Terrain_Generator, 154
 ~Terrain_Generator, 155
 aabbs, 165
 calc_smooth_normals, 156
 changeMaxHeight, 156
 expandTerrain, 156
 generate_biome, 156
 generate_indices, 157
 generate_map_chunk, 157
 generate_noise_map, 158
 generate_normals, 159
 generate_render_context, 159
 generate_vertices, 160
 generateMesh, 160
 get_material_id, 160
 get_normal_world_trafo, 161
 get_textures, 161
 get_world_trafo, 161
 getNoiseParameters, 161
 init, 161
 load_plants, 162
 meshes, 165
 regenerate, 162
 render, 163
 render_plants, 163
 retrieive_uniform_locations, 164
 set_material_id, 164
 set_world_trafo, 164
 setNoiseParameters, 165
 Terrain_Generator, 155
 transform, 165
 terrain_height
 GraphicsEngine.cpp, 230
 Terrain_Model, 166
 angle, 168
 BUSH, 168
 M_TYPE, 167
 pos, 168
 STONE, 168
 Terrain_Model, 168
 TREE, 168
 type, 168
 variation, 168
 xOffset, 168
 yOffset, 169

 Terrain_Texture, 169
 ~Terrain_Texture, 170
 BIOMETYPE, 170
 clear_texture_context, 171
 DARK_GRASS, 170
 DARK_ROCK, 170
 DEEP_WATER, 170
 LIGHT_GRASS, 170
 LIGHT_ROCK, 170
 load_all_texture, 171
 load_texture, 171
 retrive_uniform_locations, 172
 SAND, 170
 setHeights, 172
 SHALLOW_WATER, 170
 SNOW, 170
 Terrain_Texture, 172
 unbind_texture, 172
 use_texture, 173
 terrainType, 174
 height, 174
 terrainType, 174
 texCoord, 174
 texCoord
 terrainType, 174
 Texture, 175
 ~Texture, 176
 clear_texture_context, 176
 get_filename, 176
 get_id, 177
 load_texture_with_alpha_channel, 177
 load_texture_without_alpha_channel, 177
 Texture, 176
 unbind_texture, 177
 use_texture, 178
 texture_coords
 Vertex, 182
 TextureWrappingMode, 178
 ~TextureWrappingMode, 179
 activate, 179
 TextureWrappingMode, 179
 tGenerator
 GraphicsEngine.cpp, 230
 TINYOBJLOADER_IMPLEMENTATION
 GraphicsEngine.cpp, 220
 transform
 Terrain_Generator, 165
 transform_Mesh
 Mesh, 92
 transform_model
 Model, 97
 translate
 GameObject, 54
 TREE
 Terrain_Model, 168
 type
 Terrain_Model, 168
 unbind_texture

Terrain_Texture, 172
Texture, 177
uniform_absorption_coeff_location
 LightingPassShaderProgram, 83
uniform_ambient_intensity
 LightingPassShaderProgram, 83
uniform_ambient_intensity_location
 DirectionalLightUniformLocations, 46
uniform_biom_texture_locations
 GeometryPassShaderProgram, 65
uniform_biomHeight_id
 GeometryPassShaderProgram, 65
uniform_color
 LightingPassShaderProgram, 84
uniform_color_location
 DirectionalLightUniformLocations, 46
uniform_constant
 LightingPassShaderProgram, 84
uniform_diffuse_intensity
 LightingPassShaderProgram, 84
uniform_diffuse_intensity_location
 DirectionalLightUniformLocations, 46
uniform_direction_location
 DirectionalLightUniformLocations, 47
uniform_exponent
 LightingPassShaderProgram, 84
uniform_far_plane
 LightingPassShaderProgram, 84
uniform_IOR_location
 LightingPassShaderProgram, 84
uniform_isTerrainValue_id
 GeometryPassShaderProgram, 65
uniform_linear
 LightingPassShaderProgram, 84
uniform_material_id_location
 GeometryPassShaderProgram, 66
uniform_max_height_id
 GeometryPassShaderProgram, 66
uniform_metallic_location
 LightingPassShaderProgram, 84
uniform_model_location
 CloudsShaderProgram, 34
 GeometryPassShaderProgram, 66
uniform_normal_model_location
 GeometryPassShaderProgram, 66
uniform_position
 LightingPassShaderProgram, 85
uniform_projection_location
 CloudsShaderProgram, 34
 GeometryPassShaderProgram, 66
uniform_roughness_location
 LightingPassShaderProgram, 85
uniform_shadow_intensity_location
 DirectionalLightUniformLocations, 47
uniform_shadow_map
 LightingPassShaderProgram, 85
uniform_view_location
 CloudsShaderProgram, 34

GeometryPassShaderProgram, 66
update
 Noise, 104
update_current_space_ship
 Scene, 130
update_shadow_map
 DirectionalLight, 45
update_viewport
 MyWindow, 100
update_window_params
 Clouds, 30
 GBuffer, 56
use_gbuffer
 GBuffer, 56
use_light
 PointLight, 118
use_material
 Material, 89
use_shader_program
 ShaderProgram, 139
use_terrain_textures
 DirectionalShadowMapPass, 49
 GeometryPass, 59
 OmniShadowMapPass, 114
 RenderPassSceneDependend, 123
use_texture
 Terrain_Texture, 173
 Texture, 178

validate_program
 ShaderProgram, 139
variation
 Terrain_Model, 168
Vertex, 180
 get_normal, 181
 get_position, 181
 get_tex_coors, 181
 normal, 181
 operator==, 181
 position, 181
 texture_coords, 182
 Vertex, 180
vertex_location
 ShaderProgram, 141
ViewFrustumCulling, 182
 ~ViewFrustumCulling, 183
 is_inside, 183
 render_view_frustum, 183
 ViewFrustumCulling, 182

write
 CascadedShadowMap, 21
 OmniDirShadowMap, 107
 ShadowMap, 144

xOffset
 Terrain_Model, 168

yOffset
 Terrain_Model, 169