

Humanity simulator (Készítette: Funk Gábor)

Készítette Doxygen 1.9.1



<b>1. Névtérmutató</b>	<b>1</b>
1.1. Névtérlista	1
<b>2. Hierarchikus mutató</b>	<b>3</b>
2.1. Osztályhierarchia	3
<b>3. Osztálymutató</b>	<b>5</b>
3.1. Osztálylista	5
<b>4. Fájlmutató</b>	<b>9</b>
4.1. Fájllista	9
<b>5. Névterek dokumentációja</b>	<b>13</b>
5.1. creature névtér-referencia	13
5.1.1. Részletes leírás	14
5.1.2. Enumerációk dokumentációja	14
5.1.2.1. ENTITY_GENDER	14
5.1.2.2. ENTITY_TYPE	14
5.1.2.3. FACING	15
5.1.2.4. LIVINGSTATE	15
5.2. creatures névtér-referencia	15
5.3. gtest_lite névtér-referencia	15
5.3.1. Részletes leírás	16
5.3.2. Függvények dokumentációja	16
5.3.2.1. almostEQ()	16
5.3.2.2. eq()	16
5.3.2.3. eqstr()	17
5.3.2.4. eqstrcase()	17
5.3.2.5. EXPECT_() [1/2]	17
5.3.2.6. EXPECT_() [2/2]	17
5.3.2.7. EXPECTSTR()	18
5.3.2.8. ge()	18
5.3.2.9. gt()	18
5.3.2.10. le()	18
5.3.2.11. lt()	18
5.3.2.12. ne()	19
5.3.2.13. nestr()	19
5.4. minerals névtér-referencia	19
5.4.1. Részletes leírás	20
5.4.2. Enumerációk dokumentációja	20
5.4.2.1. MINERAL_TYPE	20
5.4.3. Függvények dokumentációja	20
5.4.3.1. mineral_to_string()	20
5.5. sf névtér-referencia	20

5.5.1.	Enumerációk dokumentációja	21
5.5.1.1.	BlendMode	21
5.5.2.	Függvények dokumentációja	22
5.5.2.1.	file_exists_at_path()	22
5.5.2.2.	to_string()	22
5.5.3.	Változók dokumentációja	22
5.5.3.1.	BlendAdd	22
5.6.	tiles névtér-referencia	22
5.6.1.	Részletes leírás	22
5.6.2.	Enumerációk dokumentációja	22
5.6.2.1.	TILETYPE	22
5.7.	ui névtér-referencia	23
5.7.1.	Részletes leírás	23
<b>6.</b>	<b>Osztályok dokumentációja</b>	<b>25</b>
6.1.	_Is_Types< F, T > struktúrasablon-referencia	25
6.1.1.	Részletes leírás	25
6.1.2.	Tagfüggvények dokumentációja	25
6.1.2.1.	f() [1/2]	26
6.1.2.2.	f() [2/2]	26
6.1.3.	Adattagok dokumentációja	26
6.1.3.1.	convertable	26
6.2.	creature::AnglerMiner osztályreferencia	26
6.2.1.	Részletes leírás	27
6.2.2.	Konstruktorok és destruktorok dokumentációja	27
6.2.2.1.	AnglerMiner()	27
6.2.2.2.	~AnglerMiner()	27
6.2.3.	Tagfüggvények dokumentációja	27
6.2.3.1.	update_logic()	28
6.3.	creature::Bear osztályreferencia	28
6.3.1.	Részletes leírás	29
6.3.2.	Konstruktorok és destruktorok dokumentációja	29
6.3.2.1.	Bear()	29
6.3.2.2.	~Bear()	29
6.3.3.	Tagfüggvények dokumentációja	29
6.3.3.1.	die()	29
6.3.3.2.	draw_logic()	30
6.3.3.3.	get_type()	30
6.3.3.4.	select_target()	30
6.3.3.5.	update_logic()	31
6.4.	minerals::BerryBush osztályreferencia	31
6.4.1.	Részletes leírás	32

6.4.2.	Konstruktorok és destruktorok dokumentációja	32
6.4.2.1.	BerryBush()	32
6.4.3.	Tagfüggvények dokumentációja	32
6.4.3.1.	get_type()	32
6.4.3.2.	harvest()	32
6.4.3.3.	play_destroy_sound()	32
6.4.3.4.	update_logic()	32
6.5.	sf::Bound osztályreferencia	33
6.5.1.	Adattagok dokumentációja	33
6.5.1.1.	height	33
6.5.1.2.	width	33
6.6.	creature::Builder osztályreferencia	34
6.6.1.	Részletes leírás	34
6.6.2.	Konstruktorok és destruktorok dokumentációja	34
6.6.2.1.	Builder()	34
6.6.2.2.	~Builder()	35
6.6.3.	Tagfüggvények dokumentációja	35
6.6.3.1.	update_logic()	35
6.7.	ui::Button osztályreferencia	35
6.7.1.	Részletes leírás	36
6.7.2.	Konstruktorok és destruktorok dokumentációja	36
6.7.2.1.	Button()	36
6.7.3.	Tagfüggvények dokumentációja	37
6.7.3.1.	draw()	37
6.7.3.2.	onClick()	37
6.7.3.3.	setCallback()	37
6.7.3.4.	setPosition()	38
6.7.3.5.	setTexture()	38
6.7.3.6.	try_hover_animation()	38
6.8.	minerals::CityCenter osztályreferencia	39
6.8.1.	Részletes leírás	39
6.8.2.	Konstruktorok és destruktorok dokumentációja	39
6.8.2.1.	CityCenter()	40
6.8.3.	Tagfüggvények dokumentációja	40
6.8.3.1.	get_settlement_age()	40
6.8.3.2.	get_type()	40
6.8.3.3.	is_there_room_for_housing()	40
6.8.3.4.	register_new_house()	40
6.8.3.5.	update_logic()	40
6.9.	CityCenterException osztályreferencia	41
6.9.1.	Részletes leírás	41
6.9.2.	Konstruktorok és destruktorok dokumentációja	41

6.9.2.1. CityCenterException()	41
6.10. sf::Clock osztályreferencia	42
6.10.1. Tagfüggvények dokumentációja	42
6.10.1.1. getElapsedTime()	42
6.10.1.2. restart()	42
6.11. sf::ClockTime osztályreferencia	42
6.11.1. Konstruktork és destruktorok dokumentációja	42
6.11.1.1. ClockTime() [1/2]	43
6.11.1.2. ClockTime() [2/2]	43
6.11.2. Tagfüggvények dokumentációja	43
6.11.2.1. asSeconds()	43
6.11.2.2. increment()	43
6.11.2.3. reset()	43
6.12. sf::Color osztályreferencia	43
6.12.1. Konstruktork és destruktorok dokumentációja	44
6.12.1.1. Color() [1/3]	44
6.12.1.2. Color() [2/3]	44
6.12.1.3. Color() [3/3]	44
6.12.2. Adattagok dokumentációja	44
6.12.2.1. a	45
6.12.2.2. b	45
6.12.2.3. Black	45
6.12.2.4. Blue	45
6.12.2.5. g	45
6.12.2.6. Green	45
6.12.2.7. r	45
6.12.2.8. Red	45
6.12.2.9. Transparent	46
6.12.2.10. White	46
6.13. creature::Crocodile osztályreferencia	46
6.13.1. Részletes leírás	47
6.13.2. Konstruktork és destruktorok dokumentációja	47
6.13.2.1. Crocodile()	47
6.13.2.2. ~Crocodile()	47
6.13.3. Tagfüggvények dokumentációja	47
6.13.3.1. die()	47
6.13.3.2. draw_logic()	48
6.13.3.3. get_type()	48
6.13.3.4. select_target()	48
6.13.3.5. update_logic()	49
6.14. creature::EntityBase osztályreferencia	49
6.14.1. Részletes leírás	51

6.14.2. Konstruktorkok és destruktorkok dokumentációja	51
6.14.2.1. ~EntityBase()	51
6.14.3. Tagfüggvények dokumentációja	51
6.14.3.1. apply_age()	52
6.14.3.2. die()	52
6.14.3.3. get_death_timer()	52
6.14.3.4. get_gender()	52
6.14.3.5. get_save_name()	52
6.14.3.6. get_state()	53
6.14.3.7. get_type()	53
6.14.3.8. set_attack_texture()	53
6.14.3.9. set_death_texture()	53
6.14.3.10. set_health()	54
6.14.3.11. set_idle_texture()	54
6.14.3.12. set_run_texture()	54
6.14.3.13. set_save_name()	55
6.14.3.14. set_state()	55
6.14.3.15. set_walk_texture()	55
6.14.4. Adattagok dokumentációja	55
6.14.4.1. death_timer	55
6.14.4.2. facing	55
6.14.4.3. gender	56
6.14.4.4. health	56
6.14.4.5. hit_timer	56
6.14.4.6. inner_timer	56
6.14.4.7. max_age	56
6.14.4.8. posX	56
6.14.4.9. posY	57
6.14.4.10. run_speed_modifier	57
6.14.4.11. save_name	57
6.14.4.12. speed	57
6.14.4.13. state	57
6.14.4.14. texture_data	57
6.15. EntityPlacer osztályreferencia	58
6.15.1. Részletes leírás	58
6.15.2. Konstruktorkok és destruktorkok dokumentációja	58
6.15.2.1. EntityPlacer()	58
6.15.3. Tagfüggvények dokumentációja	58
6.15.3.1. reset_mouse()	59
6.15.3.2. select_entity()	59
6.15.3.3. setup_factory()	59
6.15.3.4. toggle_placing()	59

6.15.3.5. try_place_entity()	59
6.15.4. Adattagok dokumentációja	59
6.15.4.1. spacePreviouslyPressed	60
6.16. sf::Event osztályreferencia	60
6.16.1. Enumeráció-tagok dokumentációja	60
6.16.1.1. EType	60
6.16.2. Konstruktork és destruktorok dokumentációja	60
6.16.2.1. Event()	61
6.16.3. Adattagok dokumentációja	61
6.16.3.1. type	61
6.17. creature::Farmer osztályreferencia	61
6.17.1. Részletes leírás	62
6.17.2. Konstruktork és destruktorok dokumentációja	62
6.17.2.1. Farmer()	62
6.17.2.2. ~Farmer()	62
6.17.3. Tagfüggvények dokumentációja	62
6.17.3.1. update_logic()	62
6.18. creature::Fisherman osztályreferencia	63
6.18.1. Részletes leírás	64
6.18.2. Konstruktork és destruktorok dokumentációja	64
6.18.2.1. Fisherman()	64
6.18.2.2. ~Fisherman()	64
6.18.3. Tagfüggvények dokumentációja	64
6.18.3.1. try_fishing()	65
6.18.3.2. update_logic()	65
6.18.4. Adattagok dokumentációja	65
6.18.4.1. fishing	65
6.19. sf::FloatRect osztályreferencia	65
6.19.1. Konstruktork és destruktorok dokumentációja	66
6.19.1.1. FloatRect() [1/2]	66
6.19.1.2. FloatRect() [2/2]	66
6.19.2. Tagfüggvények dokumentációja	66
6.19.2.1. contains()	66
6.19.3. Adattagok dokumentációja	66
6.19.3.1. height	67
6.19.3.2. left	67
6.19.3.3. top	67
6.19.3.4. width	67
6.20. GameConfig osztályreferencia	67
6.20.1. Részletes leírás	68
6.20.2. Konstruktork és destruktorok dokumentációja	68
6.20.2.1. GameConfig()	69



6.20.3. Tagfüggvények dokumentációja	69
6.20.3.1. get_config_level()	69
6.20.3.2. get_hostiles_count()	69
6.20.3.3. get_instance()	69
6.20.3.4. get_lang()	69
6.20.3.5. get_max_spawn_tries()	69
6.20.3.6. get_resource_scarcity()	70
6.20.3.7. get_screen_height()	70
6.20.3.8. get_screen_width()	70
6.20.3.9. get_sfml_lang()	70
6.20.3.10. get_target_fps()	70
6.20.3.11. get_world_size()	70
6.20.3.12. is_chromatic_aberration()	70
6.20.3.13. is_noise()	71
6.20.3.14. operator=()	71
6.20.3.15. read_from_config_file()	71
6.20.3.16. set_config_level()	71
6.20.3.17. set_world_size()	71
6.20.4. Adattagok dokumentációja	71
6.20.4.1. day_length	72
6.21. GameManager osztályreferencia	72
6.21.1. Részletes leírás	72
6.21.2. Konstruktork és destruktorok dokumentációja	73
6.21.2.1. GameManager()	73
6.21.2.2. ~GameManager()	73
6.21.3. Tagfüggvények dokumentációja	73
6.21.3.1. draw_buttons()	73
6.21.3.2. game_loop()	73
6.21.3.3. get_elapsed_time()	73
6.21.3.4. handle_unit_placement()	74
6.21.3.5. is_valid()	74
6.21.3.6. run()	74
6.21.3.7. setup_buttons()	74
6.21.3.8. simulate_tick()	74
6.21.3.9. update_buttons()	74
6.22. creature::Goat osztályreferencia	75
6.22.1. Részletes leírás	75
6.22.2. Konstruktork és destruktorok dokumentációja	75
6.22.2.1. Goat()	75
6.22.2.2. ~Goat()	76
6.22.3. Tagfüggvények dokumentációja	76
6.22.3.1. die()	76

6.22.3.2. draw_logic()	76
6.22.3.3. get_type()	77
6.22.3.4. update_logic()	77
6.23. creature::HostileInterface osztályreferencia	77
6.23.1. Részletes leírás	78
6.23.2. Konstruktorkok és destruktorkok dokumentációja	78
6.23.2.1. ~HostileInterface()	79
6.23.3. Tagfüggvények dokumentációja	79
6.23.3.1. check_aggroed()	79
6.23.3.2. hostile_run()	79
6.23.3.3. hostile_walk()	79
6.23.3.4. retarget()	79
6.23.3.5. select_target()	80
6.23.3.6. set_hostile_config()	80
6.23.3.7. try_attack()	80
6.23.4. Adattagok dokumentációja	80
6.23.4.1. attack_speed	81
6.23.4.2. damage	81
6.23.4.3. goal	81
6.23.4.4. target	81
6.24. minerals::House osztályreferencia	81
6.24.1. Részletes leírás	82
6.24.2. Konstruktorkok és destruktorkok dokumentációja	82
6.24.2.1. House()	82
6.24.3. Tagfüggvények dokumentációja	82
6.24.3.1. get_level()	82
6.24.3.2. get_type()	83
6.24.3.3. set_level()	83
6.24.3.4. try_upgrade()	83
6.24.3.5. update_logic()	83
6.24.3.6. upgrade_house()	83
6.25. creature::Human osztályreferencia	84
6.25.1. Részletes leírás	85
6.25.2. Konstruktorkok és destruktorkok dokumentációja	85
6.25.2.1. Human() [1/2]	85
6.25.2.2. Human() [2/2]	85
6.25.2.3. ~Human()	86
6.25.3. Tagfüggvények dokumentációja	86
6.25.3.1. die()	86
6.25.3.2. draw_logic()	86
6.25.3.3. get_profession_string()	87
6.25.3.4. get_type()	87

6.25.3.5. humanoid_run()	87
6.25.3.6. humanoid_walk()	87
6.25.3.7. initialize()	87
6.25.3.8. select_texture()	88
6.25.3.9. update_logic()	88
6.25.4. Adattagok dokumentációja	88
6.25.4.1. goal	89
6.25.4.2. needs_promotion	89
6.25.4.3. needs_to_be_royal	89
6.25.4.4. profession	89
6.26. HumanResources osztályreferencia	89
6.26.1. Részletes leírás	90
6.26.2. Tagfüggvények dokumentációja	90
6.26.2.1. add_resources()	90
6.26.2.2. get_count_from()	90
6.26.2.3. is_there_enough_resource()	91
6.26.2.4. remove_resources()	91
6.26.2.5. set_resources()	91
6.27. ImportInvalidEntityException osztályreferencia	92
6.27.1. Részletes leírás	92
6.27.2. Konstruktork és destruktorok dokumentációja	92
6.27.2.1. ImportInvalidEntityException()	92
6.28. ImportInvalidHousingLevelException osztályreferencia	93
6.28.1. Részletes leírás	93
6.28.2. Konstruktork és destruktorok dokumentációja	93
6.28.2.1. ImportInvalidHousingLevelException()	93
6.29. ImportInvalidHumanProfessionException osztályreferencia	93
6.29.1. Részletes leírás	94
6.29.2. Konstruktork és destruktorok dokumentációja	94
6.29.2.1. ImportInvalidHumanProfessionException()	94
6.30. ImportInvalidResourceException osztályreferencia	94
6.30.1. Részletes leírás	94
6.30.2. Konstruktork és destruktorok dokumentációja	95
6.30.2.1. ImportInvalidResourceException()	95
6.31. sf::IntRect osztályreferencia	95
6.31.1. Konstruktork és destruktorok dokumentációja	95
6.31.1.1. IntRect() [1/2]	95
6.31.1.2. IntRect() [2/2]	95
6.31.2. Adattagok dokumentációja	96
6.31.2.1. height	96
6.31.2.2. left	96
6.31.2.3. top	96

6.31.2.4. width	96
6.32. InvalidBorderSizeException osztályreferencia	96
6.32.1. Részletes leírás	97
6.32.2. Konstruktork és destruktorok dokumentációja	97
6.32.2.1. InvalidBorderSizeException()	97
6.33. minerals::Iron osztályreferencia	97
6.33.1. Részletes leírás	98
6.33.2. Konstruktork és destruktorok dokumentációja	98
6.33.2.1. Iron()	98
6.33.3. Tagfüggvények dokumentációja	98
6.33.3.1. get_type()	98
6.33.3.2. play_destroy_sound()	98
6.33.3.3. update_logic()	98
6.34. sf::Keyboard osztályreferencia	99
6.34.1. Enumeráció-tagok dokumentációja	99
6.34.1.1. Key	99
6.34.2. Tagfüggvények dokumentációja	100
6.34.2.1. isKeyPressed()	100
6.34.2.2. simulate_key_press()	100
6.34.2.3. simulate_key_release()	100
6.35. creature::KillerRobot osztályreferencia	100
6.35.1. Részletes leírás	101
6.35.2. Konstruktork és destruktorok dokumentációja	101
6.35.2.1. KillerRobot()	101
6.35.2.2. ~KillerRobot()	102
6.35.3. Tagfüggvények dokumentációja	102
6.35.3.1. die()	102
6.35.3.2. draw_logic()	102
6.35.3.3. get_type()	102
6.35.3.4. select_target()	103
6.35.3.5. update_logic()	103
6.36. creature::King osztályreferencia	103
6.36.1. Részletes leírás	104
6.36.2. Konstruktork és destruktorok dokumentációja	104
6.36.2.1. King()	104
6.36.2.2. ~King()	105
6.36.3. Tagfüggvények dokumentációja	105
6.36.3.1. update_logic()	105
6.37. creature::Living osztályreferencia	105
6.37.1. Részletes leírás	107
6.37.2. Konstruktork és destruktorok dokumentációja	107
6.37.2.1. ~Living()	107

6.37.3. Tagfüggvények dokumentációja	107
6.37.3.1. check_aggroed()	108
6.37.3.2. damage()	108
6.37.3.3. draw()	108
6.37.3.4. draw_logic()	108
6.37.3.5. get_width()	109
6.37.3.6. init_spritesheet_data()	109
6.37.3.7. look_left()	109
6.37.3.8. look_right()	110
6.37.3.9. needs_drawn()	110
6.37.3.10. retarget()	110
6.37.3.11. set_state()	110
6.37.3.12. setPosition()	111
6.37.3.13. setTexture()	111
6.37.3.14. setTheShadow()	111
6.37.3.15. shadow_logic()	112
6.37.3.16. update_logic()	112
6.37.3.17. update_spritesheet()	112
6.37.4. Adattagok dokumentációja	113
6.37.4.1. damaged_by	113
6.37.4.2. MAX_CREATURE_SIZE	113
6.38. creature::LivingTexture osztályreferencia	113
6.38.1. Részletes leírás	114
6.38.2. Adattagok dokumentációja	114
6.38.2.1. animation_speed	114
6.38.2.2. attack_texture_path	114
6.38.2.3. current_animation_time	114
6.38.2.4. death_texture	114
6.38.2.5. frame_count	114
6.38.2.6. idle_texture_path	115
6.38.2.7. run_texture_path	115
6.38.2.8. walk_texture_path	115
6.39. sf::Mouse osztályreferencia	115
6.39.1. Enumeráció-tagok dokumentációja	115
6.39.1.1. Mousedowntype	115
6.39.2. Tagfüggvények dokumentációja	116
6.39.2.1. getPosition()	116
6.39.2.2. isButtonPressed()	116
6.39.2.3. simulate_key_press()	116
6.39.2.4. simulate_key_release()	116
6.40. sf::Music osztályreferencia	116
6.40.1. Konstruktorkok és destruktorkok dokumentációja	117

6.40.1.1. Music()	117
6.40.2. Tagfüggvények dokumentációja	117
6.40.2.1. getStatus()	117
6.40.2.2. openFromFile()	117
6.40.2.3. play()	117
6.40.2.4. setLoop()	117
6.40.2.5. setVolume()	118
6.40.2.6. stop()	118
6.41. MusicLoadException osztályreferencia	118
6.41.1. Részletes leírás	118
6.41.2. Konstruktork és destruktorok dokumentációja	118
6.41.2.1. MusicLoadException()	119
6.42. MediaPlayer osztályreferencia	119
6.42.1. Részletes leírás	119
6.42.2. Konstruktork és destruktorok dokumentációja	119
6.42.2.1. MediaPlayer()	119
6.42.2.2. ~MediaPlayer()	120
6.42.3. Tagfüggvények dokumentációja	120
6.42.3.1. load_music()	120
6.42.3.2. set_volume()	120
6.42.3.3. toggle_music()	120
6.43. ObjectRegistry osztályreferencia	121
6.43.1. Részletes leírás	121
6.43.2. Tagfüggvények dokumentációja	121
6.43.2.1. register_type()	121
6.43.2.2. spawn()	121
6.44. gtest_lite::ostreamRedir osztályreferencia	122
6.44.1. Részletes leírás	122
6.44.2. Konstruktork és destruktorok dokumentációja	122
6.44.2.1. ostreamRedir()	122
6.44.2.2. ~ostreamRedir()	122
6.45. PostProcessor osztályreferencia	122
6.45.1. Részletes leírás	123
6.45.2. Konstruktork és destruktorok dokumentációja	123
6.45.2.1. PostProcessor()	123
6.45.3. Tagfüggvények dokumentációja	123
6.45.3.1. draw()	123
6.45.3.2. setColorOverlay()	124
6.45.3.3. setRenderSize()	124
6.45.3.4. setTextureFor()	124
6.45.3.5. toggle_chromatic_aberration()	125
6.45.3.6. toggle_noise()	125

6.45.3.7. toggle_vignette()	125
6.46. Profession osztályreferencia	126
6.46.1. Részletes leírás	126
6.46.2. Konstruktork és destruktorok dokumentációja	126
6.46.2.1. Profession()	126
6.46.3. Tagfüggvények dokumentációja	127
6.46.3.1. draw()	127
6.46.3.2. load_profession()	127
6.46.3.3. setPosition()	127
6.46.3.4. setTexture()	128
6.46.3.5. to_string()	128
6.47. RandomGenerator osztályreferencia	128
6.47.1. Részletes leírás	129
6.47.2. Konstruktork és destruktorok dokumentációja	129
6.47.2.1. RandomGenerator()	129
6.47.3. Tagfüggvények dokumentációja	129
6.47.3.1. get_instance()	129
6.47.3.2. get_random_int()	129
6.47.3.3. operator=()	130
6.48. ReadSaveFileFail osztályreferencia	130
6.48.1. Részletes leírás	130
6.48.2. Konstruktork és destruktorok dokumentációja	131
6.48.2.1. ReadSaveFileFail()	131
6.49. sf::RectangleShape osztályreferencia	131
6.49.1. Tagfüggvények dokumentációja	131
6.49.1.1. setFillColor()	131
6.49.1.2. setPosition()	131
6.49.1.3. setSize()	132
6.49.2. Adattagok dokumentációja	132
6.49.2.1. position	132
6.50. sf::RenderStates osztályreferencia	132
6.50.1. Konstruktork és destruktorok dokumentációja	132
6.50.1.1. RenderStates()	132
6.50.2. Tagfüggvények dokumentációja	133
6.50.2.1. setBlendMode()	133
6.50.2.2. setTransform()	133
6.50.3. Adattagok dokumentációja	133
6.50.3.1. blendMode	133
6.50.3.2. transform	133
6.51. sf::RenderWindow osztályreferencia	133
6.51.1. Konstruktork és destruktorok dokumentációja	134
6.51.1.1. RenderWindow() [1/3]	134

6.51.1.2. <a href="#">RenderWindow()</a> [2/3]	134
6.51.1.3. <a href="#">RenderWindow()</a> [3/3]	134
6.51.2. <a href="#">Tagfüggvények dokumentációja</a>	134
6.51.2.1. <a href="#">clear()</a> [1/2]	135
6.51.2.2. <a href="#">clear()</a> [2/2]	135
6.51.2.3. <a href="#">close()</a>	135
6.51.2.4. <a href="#">create()</a>	135
6.51.2.5. <a href="#">display()</a>	135
6.51.2.6. <a href="#">draw()</a> [1/3]	135
6.51.2.7. <a href="#">draw()</a> [2/3]	135
6.51.2.8. <a href="#">draw()</a> [3/3]	136
6.51.2.9. <a href="#">isOpen()</a>	136
6.51.2.10. <a href="#">pollEvent()</a>	136
6.51.2.11. <a href="#">setFramerateLimit()</a>	136
6.52. <a href="#">minerals::ResourceStructure osztályreferencia</a>	136
6.52.1. <a href="#">Részletes leírás</a>	137
6.52.2. <a href="#">Konstruktorok és destruktorkok dokumentációja</a>	137
6.52.2.1. <a href="#">ResourceStructure()</a>	137
6.52.2.2. <a href="#">~ResourceStructure()</a>	138
6.52.3. <a href="#">Tagfüggvények dokumentációja</a>	138
6.52.3.1. <a href="#">get_harvested()</a>	138
6.52.3.2. <a href="#">get_needs_remove()</a>	138
6.52.3.3. <a href="#">harvest()</a>	138
6.52.3.4. <a href="#">play_destroy_sound()</a>	138
6.52.4. <a href="#">Adattagok dokumentációja</a>	138
6.52.4.1. <a href="#">harvested</a>	139
6.52.4.2. <a href="#">inner_timer</a>	139
6.52.4.3. <a href="#">needs_remove</a>	139
6.53. <a href="#">RoleOption struktúráreferencia</a>	139
6.53.1. <a href="#">Részletes leírás</a>	139
6.53.2. <a href="#">Adattagok dokumentációja</a>	139
6.53.2.1. <a href="#">create</a>	140
6.53.2.2. <a href="#">requirements</a>	140
6.54. <a href="#">SaveHelper osztályreferencia</a>	140
6.54.1. <a href="#">Részletes leírás</a>	140
6.54.2. <a href="#">Tagfüggvények dokumentációja</a>	140
6.54.2.1. <a href="#">get_roles()</a>	140
6.54.2.2. <a href="#">getCreatureFactory()</a>	141
6.54.2.3. <a href="#">getHumanFactory()</a>	141
6.54.2.4. <a href="#">getResourceFactory()</a>	141
6.54.2.5. <a href="#">trim_brackets()</a>	141
6.55. <a href="#">SaveManager osztályreferencia</a>	141



6.55.1. Részletes leírás . . . . .	142
6.55.2. Konstruktork és destruktorok dokumentációja . . . . .	142
6.55.2.1. SaveManager() . . . . .	142
6.55.3. Tagfüggvények dokumentációja . . . . .	142
6.55.3.1. deleteFile() . . . . .	142
6.55.3.2. loadFile() . . . . .	142
6.55.3.3. saveFile() . . . . .	143
6.56. Shadowable osztályreferencia . . . . .	143
6.56.1. Részletes leírás . . . . .	144
6.56.2. Konstruktork és destruktorok dokumentációja . . . . .	144
6.56.2.1. ~Shadowable() . . . . .	144
6.56.3. Tagfüggvények dokumentációja . . . . .	144
6.56.3.1. drawShadow() . . . . .	144
6.56.3.2. get_height_offset() . . . . .	145
6.56.3.3. get_shadow_strength() . . . . .	145
6.56.3.4. get_skew_offset() . . . . .	145
6.56.3.5. set_height_offset() . . . . .	145
6.56.3.6. set_shadow_strength() . . . . .	146
6.56.3.7. set_skew_offset() . . . . .	146
6.56.3.8. setShadow() . . . . .	146
6.56.3.9. setShadowDayNightCycle() . . . . .	147
6.56.3.10. setShadowPosition() . . . . .	147
6.56.3.11. setShadowTexture() . . . . .	147
6.56.4. Adattagok dokumentációja . . . . .	147
6.56.4.1. height_offset . . . . .	148
6.57. SimulationException osztályreferencia . . . . .	148
6.57.1. Részletes leírás . . . . .	148
6.57.2. Konstruktork és destruktorok dokumentációja . . . . .	148
6.57.2.1. SimulationException() . . . . .	148
6.58. creature::Soldier osztályreferencia . . . . .	149
6.58.1. Részletes leírás . . . . .	149
6.58.2. Konstruktork és destruktorok dokumentációja . . . . .	149
6.58.2.1. Soldier() . . . . .	149
6.58.2.2. ~Soldier() . . . . .	150
6.58.3. Tagfüggvények dokumentációja . . . . .	150
6.58.3.1. update_logic() . . . . .	150
6.59. sf::Sound osztályreferencia . . . . .	150
6.59.1. Konstruktork és destruktorok dokumentációja . . . . .	151
6.59.1.1. ~Sound() . . . . .	151
6.59.2. Tagfüggvények dokumentációja . . . . .	151
6.59.2.1. play() . . . . .	151
6.59.2.2. setBuffer() . . . . .	151

6.59.2.3. stop()	151
6.60. sf::SoundBuffer osztályreferencia	151
6.60.1. Tagfüggvények dokumentációja	152
6.60.1.1. loadFromFile()	152
6.61. SoundPlayer osztályreferencia	152
6.61.1. Részletes leírás	152
6.61.2. Tagfüggvények dokumentációja	152
6.61.2.1. load_sound()	152
6.61.2.2. play_sound()	153
6.61.2.3. stop_sound()	153
6.62. sf::SoundSource osztályreferencia	153
6.62.1. Enumeráció-tagok dokumentációja	154
6.62.1.1. SoundSourceType	154
6.62.2. Konstruktork és destruktorok dokumentációja	154
6.62.2.1. SoundSource()	154
6.62.2.2. ~SoundSource()	154
6.62.3. Adattagok dokumentációja	154
6.62.3.1. type	154
6.63. sf::Sprite osztályreferencia	155
6.63.1. Konstruktork és destruktorok dokumentációja	155
6.63.1.1. Sprite()	155
6.63.1.2. ~Sprite()	155
6.63.2. Tagfüggvények dokumentációja	155
6.63.2.1. draw()	155
6.63.2.2. getGlobalBounds() [1/2]	156
6.63.2.3. getGlobalBounds() [2/2]	156
6.63.2.4. getLocalBounds()	156
6.63.2.5. getPosition()	156
6.63.2.6. getTexture()	156
6.63.2.7. setColor()	156
6.63.2.8. setOrigin()	156
6.63.2.9. setPosition()	157
6.63.2.10. setRotation()	157
6.63.2.11. setScale()	157
6.63.2.12. setTexture()	157
6.63.2.13. setTextureRect()	157
6.64. minerals::Stone osztályreferencia	158
6.64.1. Részletes leírás	158
6.64.2. Konstruktork és destruktorok dokumentációja	158
6.64.2.1. Stone()	158
6.64.3. Tagfüggvények dokumentációja	159
6.64.3.1. get_type()	159

6.64.3.2. play_destroy_sound()	159
6.64.3.3. update_logic()	159
6.65. creature::Stonemason osztályreferencia	159
6.65.1. Részletes leírás	160
6.65.2. Konstruktork és destruktorok dokumentációja	160
6.65.2.1. Stonemason()	160
6.65.2.2. ~Stonemason()	161
6.65.3. Tagfüggvények dokumentációja	161
6.65.3.1. try_mine()	161
6.65.3.2. update_logic()	161
6.65.4. Adattagok dokumentációja	162
6.65.4.1. mining_iron	162
6.66. minerals::Structure osztályreferencia	162
6.66.1. Részletes leírás	163
6.66.2. Konstruktork és destruktorok dokumentációja	163
6.66.2.1. Structure()	163
6.66.2.2. ~Structure()	163
6.66.3. Tagfüggvények dokumentációja	163
6.66.3.1. draw()	164
6.66.3.2. draw_logic()	164
6.66.3.3. get_type()	164
6.66.3.4. needs_drawn()	164
6.66.3.5. setPosition()	165
6.66.3.6. setTexture()	165
6.66.3.7. update_logic()	165
6.66.4. Adattagok dokumentációja	166
6.66.4.1. MAX_OBJECT_SIZE	166
6.66.4.2. posx	166
6.66.4.3. posy	166
6.67. StructureException osztályreferencia	166
6.67.1. Részletes leírás	167
6.67.2. Konstruktork és destruktorok dokumentációja	167
6.67.2.1. StructureException()	167
6.68. TerrainContainer< T > osztálysablon-referencia	167
6.68.1. Részletes leírás	168
6.68.2. Konstruktork és destruktorok dokumentációja	168
6.68.2.1. TerrainContainer() [1/2]	168
6.68.2.2. TerrainContainer() [2/2]	169
6.68.2.3. ~TerrainContainer()	169
6.68.3. Tagfüggvények dokumentációja	169
6.68.3.1. clear()	169
6.68.3.2. clear_at()	169

6.68.3.3. draw()	170
6.68.3.4. generate_world()	170
6.68.3.5. get_height()	170
6.68.3.6. get_seed()	171
6.68.3.7. get_width()	171
6.68.3.8. is_on_screen()	171
6.68.3.9. is_valid_coordinate()	171
6.68.3.10. operator[]() [1/2]	172
6.68.3.11. operator[]() [2/2]	172
6.68.3.12. resize()	172
6.68.3.13. set_seed()	173
6.68.3.14. swap_at()	173
6.68.4. Adattagok dokumentációja	173
6.68.4.1. TILE_SIZE	173
6.69. gtest_lite::Test struktúráreferencia	174
6.69.1. Részletes leírás	175
6.69.2. Konstruktork és destruktorok dokumentációja	175
6.69.2.1. ~Test()	175
6.69.3. Tagfüggvények dokumentációja	175
6.69.3.1. astatus()	175
6.69.3.2. begin()	175
6.69.3.3. end()	175
6.69.3.4. expect()	176
6.69.3.5. fail()	176
6.69.3.6. getTest()	176
6.69.4. Adattagok dokumentációja	176
6.69.4.1. ablocks	176
6.69.4.2. failed	176
6.69.4.3. name	177
6.69.4.4. null	177
6.69.4.5. os	177
6.69.4.6. status	177
6.69.4.7. sum	177
6.69.4.8. tmp	177
6.70. sf::Texture osztályreferencia	178
6.70.1. Konstruktork és destruktorok dokumentációja	178
6.70.1.1. Texture() [1/2]	178
6.70.1.2. ~Texture()	178
6.70.1.3. Texture() [2/2]	178
6.70.2. Tagfüggvények dokumentációja	178
6.70.2.1. getSize()	178
6.70.2.2. loadFromFile()	179

6.70.2.3. operator=()	179
6.71. Textureable osztályreferencia	179
6.71.1. Részletes leírás	180
6.71.2. Konstruktorkok és destruktorkok dokumentációja	180
6.71.2.1. ~Textureable()	180
6.71.3. Tagfüggvények dokumentációja	180
6.71.3.1. draw()	180
6.71.3.2. setPosition()	180
6.71.3.3. setTexture()	181
6.72. TextureManager osztályreferencia	181
6.72.1. Részletes leírás	182
6.72.2. Tagfüggvények dokumentációja	182
6.72.2.1. clear()	182
6.72.2.2. getInstance()	182
6.72.2.3. getTexture()	182
6.72.2.4. loadTexture()	183
6.73. tiles::Tile osztályreferencia	183
6.73.1. Részletes leírás	184
6.73.2. Tagfüggvények dokumentációja	184
6.73.2.1. draw()	184
6.73.2.2. get_type()	184
6.73.2.3. init()	184
6.73.2.4. setPosition()	185
6.73.2.5. setTexture()	185
6.74. sf::Transform osztályreferencia	185
6.74.1. Konstruktorkok és destruktorkok dokumentációja	186
6.74.1.1. Transform() [1/2]	186
6.74.1.2. Transform() [2/2]	186
6.74.2. Tagfüggvények dokumentációja	186
6.74.2.1. combine()	186
6.74.2.2. transformPoint()	187
6.74.2.3. translate() [1/2]	187
6.74.2.4. translate() [2/2]	187
6.74.3. Adattagok dokumentációja	187
6.74.3.1. matrix	187
6.75. minerals::Tree osztályreferencia	187
6.75.1. Részletes leírás	188
6.75.2. Konstruktorkok és destruktorkok dokumentációja	188
6.75.2.1. Tree()	188
6.75.3. Tagfüggvények dokumentációja	188
6.75.3.1. get_type()	188
6.75.3.2. play_destroy_sound()	188

6.75.3.3. update_logic()	188
6.76. sf::Vector2f osztályreferencia	189
6.76.1. Konstruktork és destruktork dokumentációja	189
6.76.1.1. Vector2f() [1/2]	189
6.76.1.2. Vector2f() [2/2]	189
6.76.2. Adattagok dokumentációja	189
6.76.2.1. x	190
6.76.2.2. y	190
6.77. sf::Vector2i osztályreferencia	190
6.77.1. Konstruktork és destruktork dokumentációja	190
6.77.1.1. Vector2i() [1/2]	190
6.77.1.2. Vector2i() [2/2]	190
6.77.2. Adattagok dokumentációja	191
6.77.2.1. x	191
6.77.2.2. y	191
6.78. sf::VideoMode osztályreferencia	191
6.78.1. Konstruktork és destruktork dokumentációja	191
6.78.1.1. VideoMode()	192
6.78.2. Tagfüggvények dokumentációja	192
6.78.2.1. getDesktopMode()	192
6.78.2.2. isValid()	192
6.78.3. Adattagok dokumentációja	192
6.78.3.1. bitsPerPixel	192
6.78.3.2. height	192
6.78.3.3. width	192
6.79. creature::Woodcutter osztályreferencia	193
6.79.1. Részletes leírás	193
6.79.2. Konstruktork és destruktork dokumentációja	193
6.79.2.1. Woodcutter()	193
6.79.2.2. ~Woodcutter()	194
6.79.3. Tagfüggvények dokumentációja	194
6.79.3.1. update_logic()	194
6.80. World osztályreferencia	194
6.80.1. Részletes leírás	195
6.80.2. Konstruktork és destruktork dokumentációja	196
6.80.2.1. World()	196
6.80.2.2. ~World()	196
6.80.3. Tagfüggvények dokumentációja	196
6.80.3.1. clear()	196
6.80.3.2. draw()	196
6.80.3.3. get_border_height()	197
6.80.3.4. get_border_width()	197

6.80.3.5. populate_world()	197
6.80.3.6. regenerate()	197
6.80.3.7. set_border_height()	197
6.80.3.8. set_border_width()	198
6.80.3.9. spawn_entity_at_pos()	198
6.80.3.10. spawn_human()	198
6.80.3.11. try_develop_random_role()	198
6.80.3.12. update_world()	199
6.81. WorldBase osztályreferencia	199
6.81.1. Részletes leírás	201
6.81.2. Konstruktork és destruktorok dokumentációja	201
6.81.2.1. ~WorldBase()	201
6.81.3. Tagfüggvények dokumentációja	201
6.81.3.1. build_city_center_at()	201
6.81.3.2. get_current_city_center()	201
6.81.3.3. get_excluded_entities()	202
6.81.3.4. get_position_nearby_town()	202
6.81.3.5. get_random_house_pos()	202
6.81.3.6. get_random_suitable_position()	202
6.81.3.7. get_resources()	203
6.81.3.8. get_structure_type()	203
6.81.3.9. getTileAt()	203
6.81.3.10. remove_structure_at()	204
6.81.3.11. spawn_entity()	204
6.81.3.12. spawn_structure()	204
6.81.3.13. spawn_structure_at()	205
6.81.3.14. upgrade_house_at()	205
6.81.4. Adattagok dokumentációja	205
6.81.4.1. camp_needs_spawn	206
6.81.4.2. current_city_center	206
6.81.4.3. entities	206
6.81.4.4. houses	206
6.81.4.5. humans	206
6.81.4.6. MAX_OBJECT_SIZE	206
6.81.4.7. sound_player	207
6.81.4.8. structures	207
6.81.4.9. terrain	207
6.82. WorldBaseSerializble osztályreferencia	207
6.82.1. Részletes leírás	208
6.82.2. Tagfüggvények dokumentációja	208
6.82.2.1. clear()	208
6.82.2.2. reinitialize_self()	208

6.82.3. Barát és kapcsolódó függvények dokumentációja	208
6.82.3.1. operator<<	209
6.82.3.2. operator>>	209
6.82.4. Adattagok dokumentációja	209
6.82.4.1. elapsed_time	209
6.82.4.2. saved_size	209
6.83. YAMLParser osztályreferencia	209
6.83.1. Részletes leírás	210
6.83.2. Konstruktorkok és destruktorkok dokumentációja	210
6.83.2.1. YAMLParser()	210
6.83.3. Tagfüggvények dokumentációja	210
6.83.3.1. get_value_of_key()	210
6.83.3.2. parse_file()	210
6.83.3.3. try_generate_config_file()	210
<b>7. Fájlok dokumentációja</b>	<b>211</b>
7.1. src/creatures/EntityBase.cpp fájlreferencia	211
7.2. src/creatures/EntityBase.d fájlreferencia	211
7.3. src/creatures/EntityBase.h fájlreferencia	211
7.4. src/creatures/EntityUtils.h fájlreferencia	212
7.5. src/creatures/Goat.cpp fájlreferencia	212
7.6. src/creatures/Goat.d fájlreferencia	212
7.7. src/creatures/Goat.h fájlreferencia	212
7.8. src/creatures/HostileInterface.cpp fájlreferencia	213
7.9. src/creatures/HostileInterface.d fájlreferencia	213
7.10. src/creatures/HostileInterface.h fájlreferencia	213
7.11. src/creatures/hostiles/Bear.cpp fájlreferencia	213
7.12. src/creatures/hostiles/Bear.d fájlreferencia	214
7.13. src/creatures/hostiles/Bear.h fájlreferencia	214
7.14. src/creatures/hostiles/Crocodile.cpp fájlreferencia	214
7.15. src/creatures/hostiles/Crocodile.d fájlreferencia	214
7.16. src/creatures/hostiles/Crocodile.h fájlreferencia	214
7.17. src/creatures/hostiles/KillerRobot.cpp fájlreferencia	215
7.18. src/creatures/hostiles/KillerRobot.d fájlreferencia	215
7.19. src/creatures/hostiles/KillerRobot.h fájlreferencia	215
7.20. src/creatures/humans/AnglerMiner.cpp fájlreferencia	215
7.21. src/creatures/humans/AnglerMiner.d fájlreferencia	216
7.22. src/creatures/humans/AnglerMiner.h fájlreferencia	216
7.23. src/creatures/humans/Builder.cpp fájlreferencia	216
7.24. src/creatures/humans/Builder.d fájlreferencia	216
7.25. src/creatures/humans/Builder.h fájlreferencia	216
7.26. src/creatures/humans/Farmer.cpp fájlreferencia	217



7.27. src/creatures/humans/Farmer.d fájlreferencia . . . . .	217
7.28. src/creatures/humans/Farmer.h fájlreferencia . . . . .	217
7.29. src/creatures/humans/Fisherman.cpp fájlreferencia . . . . .	217
7.30. src/creatures/humans/Fisherman.d fájlreferencia . . . . .	218
7.31. src/creatures/humans/Fisherman.h fájlreferencia . . . . .	218
7.32. src/creatures/humans/Human.cpp fájlreferencia . . . . .	218
7.33. src/creatures/humans/Human.d fájlreferencia . . . . .	218
7.34. src/creatures/humans/Human.h fájlreferencia . . . . .	218
7.35. src/creatures/humans/King.cpp fájlreferencia . . . . .	219
7.36. src/creatures/humans/King.d fájlreferencia . . . . .	219
7.37. src/creatures/humans/King.h fájlreferencia . . . . .	219
7.38. src/creatures/humans/Soldier.cpp fájlreferencia . . . . .	219
7.39. src/creatures/humans/Soldier.d fájlreferencia . . . . .	220
7.40. src/creatures/humans/Soldier.h fájlreferencia . . . . .	220
7.41. src/creatures/humans/Stonemason.cpp fájlreferencia . . . . .	220
7.42. src/creatures/humans/Stonemason.d fájlreferencia . . . . .	220
7.43. src/creatures/humans/Stonemason.h fájlreferencia . . . . .	220
7.44. src/creatures/humans/Woodcutter.cpp fájlreferencia . . . . .	221
7.45. src/creatures/humans/Woodcutter.d fájlreferencia . . . . .	221
7.46. src/creatures/humans/Woodcutter.h fájlreferencia . . . . .	221
7.47. src/creatures/Living.cpp fájlreferencia . . . . .	221
7.48. src/creatures/Living.d fájlreferencia . . . . .	222
7.49. src/creatures/Living.h fájlreferencia . . . . .	222
7.50. src/EntityPlacer.cpp fájlreferencia . . . . .	222
7.51. src/EntityPlacer.d fájlreferencia . . . . .	222
7.52. src/EntityPlacer.h fájlreferencia . . . . .	222
7.53. src/exceptions/FileExceptions.h fájlreferencia . . . . .	223
7.54. src/exceptions/MusicLoadException.h fájlreferencia . . . . .	223
7.55. src/exceptions/SimulationException.h fájlreferencia . . . . .	224
7.56. src/exceptions/WorldExceptions.h fájlreferencia . . . . .	224
7.57. src/external/memtrace.cpp fájlreferencia . . . . .	224
7.58. src/external/memtrace.h fájlreferencia . . . . .	224
7.59. src/external/modified_gtest_lite.h fájlreferencia . . . . .	224
7.59.1. Makródefiníciók dokumentációja . . . . .	227
7.59.1.1. ADD_FAILURE . . . . .	227
7.59.1.2. ASSERT_ . . . . .	227
7.59.1.3. ASSERT_EQ . . . . .	228
7.59.1.4. ASSERT_NO_THROW [1/2] . . . . .	228
7.59.1.5. ASSERT_NO_THROW [2/2] . . . . .	228
7.59.1.6. ASSERTTHROW . . . . .	228
7.59.1.7. CREATE_Has_ . . . . .	228
7.59.1.8. CREATE_Has_fn_ . . . . .	229

7.59.1.9. END	229
7.59.1.10. ENDM	229
7.59.1.11. ENDMsg	229
7.59.1.12. EXPECT_ANY_THROW	229
7.59.1.13. EXPECT_DOUBLE_EQ	229
7.59.1.14. EXPECT_ENVCASEEQ	229
7.59.1.15. EXPECT_ENVEQ	230
7.59.1.16. EXPECT_EQ	230
7.59.1.17. EXPECT_FALSE	230
7.59.1.18. EXPECT_FLOAT_EQ	230
7.59.1.19. EXPECT_GE	230
7.59.1.20. EXPECT_GT	230
7.59.1.21. EXPECT_LE	230
7.59.1.22. EXPECT_LT	231
7.59.1.23. EXPECT_NE	231
7.59.1.24. EXPECT_NO_THROW	231
7.59.1.25. EXPECT_STRCASEEQ	231
7.59.1.26. EXPECT_STRCASENE	231
7.59.1.27. EXPECT_STREQ	231
7.59.1.28. EXPECT_STRNE	232
7.59.1.29. EXPECT_THROW	232
7.59.1.30. EXPECT_THROW_THROW	232
7.59.1.31. EXPECT_TRUE	232
7.59.1.32. EXPECTTHROW	232
7.59.1.33. Nem célszerű közvetlenül használni, vagy módosítani	232
7.59.1.34. FAIL	232
7.59.1.35. GTEND	233
7.59.1.36. GTINIT	233
7.59.1.37. SUCCEED	233
7.59.1.38. TEST	233
7.59.2. Függvények dokumentációja	233
7.59.2.1. hasMember()	233
7.60. src/fake_sfml/fake_sfml.cpp fájlreferencia	233
7.61. src/fake_sfml/fake_sfml.d fájlreferencia	234
7.62. src/fake_sfml/fake_sfml.h fájlreferencia	234
7.63. src/GameConfig.cpp fájlreferencia	235
7.63.1. Függvények dokumentációja	235
7.63.1.1. trim()	235
7.64. src/GameConfig.d fájlreferencia	235
7.65. src/GameConfig.h fájlreferencia	235
7.65.1. Enumerációk dokumentációja	235
7.65.1.1. Language	235

7.66. src/GameManager.cpp fájlreferencia . . . . .	235
7.67. src/GameManager.d fájlreferencia . . . . .	236
7.68. src/GameManager.h fájlreferencia . . . . .	236
7.69. src/HumanResources.cpp fájlreferencia . . . . .	236
7.70. src/HumanResources.d fájlreferencia . . . . .	236
7.71. src/HumanResources.h fájlreferencia . . . . .	236
7.72. src/main.cpp fájlreferencia . . . . .	236
7.72.1. Függvények dokumentációja . . . . .	237
7.72.1.1. main() . . . . .	237
7.73. src/main.d fájlreferencia . . . . .	237
7.74. src/MusicPlayer.cpp fájlreferencia . . . . .	237
7.75. src/MusicPlayer.d fájlreferencia . . . . .	237
7.76. src/MusicPlayer.h fájlreferencia . . . . .	237
7.77. src/PostProcessor.cpp fájlreferencia . . . . .	237
7.78. src/PostProcessor.d fájlreferencia . . . . .	237
7.79. src/PostProcessor.h fájlreferencia . . . . .	237
7.80. src/Profession.cpp fájlreferencia . . . . .	238
7.81. src/Profession.d fájlreferencia . . . . .	238
7.82. src/Profession.h fájlreferencia . . . . .	238
7.83. src/Random_Gen.cpp fájlreferencia . . . . .	238
7.84. src/Random_Gen.d fájlreferencia . . . . .	238
7.85. src/Random_Gen.h fájlreferencia . . . . .	238
7.86. src/SaveHelpers.cpp fájlreferencia . . . . .	238
7.87. src/SaveHelpers.d fájlreferencia . . . . .	239
7.88. src/SaveHelpers.h fájlreferencia . . . . .	239
7.89. src/SaveManager.cpp fájlreferencia . . . . .	239
7.90. src/SaveManager.d fájlreferencia . . . . .	240
7.91. src/SaveManager.h fájlreferencia . . . . .	240
7.92. src/Shadowable.cpp fájlreferencia . . . . .	240
7.93. src/Shadowable.d fájlreferencia . . . . .	240
7.94. src/Shadowable.h fájlreferencia . . . . .	240
7.95. src/SoundPlayer.cpp fájlreferencia . . . . .	240
7.96. src/SoundPlayer.d fájlreferencia . . . . .	240
7.97. src/SoundPlayer.h fájlreferencia . . . . .	240
7.98. src/terrain_tiles/Tile.cpp fájlreferencia . . . . .	241
7.99. src/terrain_tiles/Tile.d fájlreferencia . . . . .	241
7.100src/terrain_tiles/Tile.h fájlreferencia . . . . .	241
7.101src/TerrainContainer.hpp fájlreferencia . . . . .	241
7.101.1.Részletes leírás . . . . .	242
7.102src/TerrainContainer_def.hpp fájlreferencia . . . . .	242
7.103src/Textureable.h fájlreferencia . . . . .	242
7.104src/TextureManager.cpp fájlreferencia . . . . .	242

7.105src/TextureManager.d fájlreferencia . . . . .	242
7.106src/TextureManager.h fájlreferencia . . . . .	242
7.107src/ui/button.cpp fájlreferencia . . . . .	243
7.108src/ui/button.d fájlreferencia . . . . .	243
7.109src/ui/button.h fájlreferencia . . . . .	243
7.110src/Utils.cpp fájlreferencia . . . . .	243
7.110.1.Függvények dokumentációja . . . . .	243
7.110.1.1.distance_to() . . . . .	244
7.110.1.2.log_text() . . . . .	244
7.110.1.3.warn_text() . . . . .	244
7.111src/Utils.d fájlreferencia . . . . .	244
7.112src/Utils.h fájlreferencia . . . . .	244
7.112.1.Makródefiníciók dokumentációja . . . . .	245
7.112.1.1.WITH_SFML_RENDER . . . . .	245
7.112.2.Függvények dokumentációja . . . . .	245
7.112.2.1.distance_to() . . . . .	245
7.112.2.2.log_text() . . . . .	245
7.112.2.3.warn_text() . . . . .	246
7.113src/World.cpp fájlreferencia . . . . .	246
7.114src/World.d fájlreferencia . . . . .	246
7.115src/World.hpp fájlreferencia . . . . .	246
7.115.1.Részletes leírás . . . . .	247
7.116src/world_object/BerryBush.cpp fájlreferencia . . . . .	247
7.117src/world_object/BerryBush.d fájlreferencia . . . . .	247
7.118src/world_object/BerryBush.h fájlreferencia . . . . .	247
7.119src/world_object/CityCenter.cpp fájlreferencia . . . . .	247
7.120src/world_object/CityCenter.d fájlreferencia . . . . .	248
7.121src/world_object/CityCenter.h fájlreferencia . . . . .	248
7.122src/world_object/House.cpp fájlreferencia . . . . .	248
7.123src/world_object/House.d fájlreferencia . . . . .	248
7.124src/world_object/House.h fájlreferencia . . . . .	248
7.125src/world_object/Iron.cpp fájlreferencia . . . . .	249
7.126src/world_object/Iron.d fájlreferencia . . . . .	249
7.127src/world_object/Iron.h fájlreferencia . . . . .	249
7.128src/world_object/ResourceStructure.cpp fájlreferencia . . . . .	249
7.129src/world_object/ResourceStructure.d fájlreferencia . . . . .	249
7.130src/world_object/ResourceStructure.h fájlreferencia . . . . .	249
7.131src/world_object/Stone.cpp fájlreferencia . . . . .	250
7.132src/world_object/Stone.d fájlreferencia . . . . .	250
7.133src/world_object/Stone.h fájlreferencia . . . . .	250
7.134src/world_object/Structure.cpp fájlreferencia . . . . .	250
7.135src/world_object/Structure.d fájlreferencia . . . . .	250

---

7.136src/world_object/Structure.h fájlreferencia . . . . .	250
7.137src/world_object/Tree.cpp fájlreferencia . . . . .	251
7.138src/world_object/Tree.d fájlreferencia . . . . .	251
7.139src/world_object/Tree.h fájlreferencia . . . . .	251
7.140src/WorldBase.cpp fájlreferencia . . . . .	251
7.141src/WorldBase.d fájlreferencia . . . . .	252
7.142src/WorldBaseSerializable.cpp fájlreferencia . . . . .	252
7.142.1.Függvények dokumentációja . . . . .	252
7.142.1.1.operator<<() . . . . .	252
7.142.1.2.operator>>() . . . . .	252
7.143src/WorldBaseSerializable.d fájlreferencia . . . . .	252
7.144src/YAMLParse.cpp fájlreferencia . . . . .	252
7.145src/YAMLParse.d fájlreferencia . . . . .	252
7.146src/YAMLParse.h fájlreferencia . . . . .	252
<b>Tárgymutató</b>	<b>253</b>



# 1. fejezet

## Névtérmutató

### 1.1. Névtérlista

Az összes névtér listája rövid leírásokkal:

<a href="#">creature</a>	Az összes élőlény és entitás ebben a névtérben van . . . . .	<a href="#">13</a>
<a href="#">creatures</a>	. . . . .	<a href="#">15</a>
<a href="#">gtest_lite</a>	Gtest_lite: a keretrendszer függvényinek és objektumainak névtére . . . . .	<a href="#">15</a>
<a href="#">minerals</a>	Az összes struktúra ebben a névtérben van . . . . .	<a href="#">19</a>
<a href="#">sf</a>	. . . . .	<a href="#">20</a>
<a href="#">tiles</a>	Az összes terepkocka elem ebben a névtérben van . . . . .	<a href="#">22</a>
<a href="#">ui</a>	Az összes UI elem ebben a névtérben van . . . . .	<a href="#">23</a>





## 2. fejezet

# Hierarchikus mutató

### 2.1. Osztályhierarchia

Majdnem (de nem teljesen) betűrendbe szedett leszármazási lista:

_Is_Types< F, T > . . . . .	25
sf::Bound . . . . .	33
sf::Clock . . . . .	42
sf::ClockTime . . . . .	42
sf::Color . . . . .	43
creature::EntityBase . . . . .	49
creature::Living . . . . .	105
creature::Goat . . . . .	75
creature::HostileInterface . . . . .	77
creature::Bear . . . . .	28
creature::Crocodile . . . . .	46
creature::KillerRobot . . . . .	100
creature::Human . . . . .	84
creature::Builder . . . . .	34
creature::Farmer . . . . .	61
creature::Fisherman . . . . .	63
creature::AnglerMiner . . . . .	26
creature::King . . . . .	103
creature::Soldier . . . . .	149
creature::Stonemason . . . . .	159
creature::AnglerMiner . . . . .	26
creature::Woodcutter . . . . .	193
EntityPlacer . . . . .	58
sf::Event . . . . .	60
sf::FloatRect . . . . .	65
GameConfig . . . . .	67
GameManager . . . . .	72
HumanResources . . . . .	89
sf::IntRect . . . . .	95
sf::Keyboard . . . . .	99
creature::LivingTexture . . . . .	113
std::logic_error . . . . .	
ImportInvalidHumanProfessionException . . . . .	93
InvalidBorderSizeException . . . . .	96

sf::Mouse	115
sf::Music	116
MusicPlayer	119
ObjectRegistry	121
gtest_lite::ostreamRedir	122
PostProcessor	122
RandomGenerator	128
sf::RectangleShape	131
sf::RenderStates	132
sf::RenderWindow	133
RoleOption	139
std::runtime_error	
SimulationException	148
CityCenterException	41
ImportInvalidEntityException	92
ImportInvalidHousingLevelException	93
ImportInvalidResourceException	94
MusicLoadException	118
ReadSaveFileFail	130
StructureException	166
SaveHelper	140
SaveManager	141
Shadowable	143
creature::Living	105
minerals::Structure	162
minerals::CityCenter	39
minerals::House	81
minerals::ResourceStructure	136
minerals::BerryBush	31
minerals::Iron	97
minerals::Stone	158
minerals::Tree	187
sf::Sound	150
sf::SoundBuffer	151
SoundPlayer	152
sf::SoundSource	153
sf::Sprite	155
TerrainContainer< T >	167
TerrainContainer< tiles::Tile * >	167
gtest_lite::Test	174
sf::Texture	178
Textureable	179
Profession	126
creature::Living	105
minerals::Structure	162
tiles::Tile	183
ui::Button	35
TextureManager	181
sf::Transform	185
sf::Vector2f	189
sf::Vector2i	190
sf::VideoMode	191
WorldBase	199
WorldBaseSerialiazble	207
World	194
YAMLParse	209

## 3. fejezet

# Osztálymutató

### 3.1. Osztálylista

Az összes osztály, struktúra, unió és interfész listája rövid leírásokkal:

<a href="#">_Is_Types&lt; F, T &gt;</a>	25
Segédsablon típuskonverzió futás közbeni ellenőrzésere	
<a href="#">creature::AnglerMiner</a>	26
Az "AnglerMiner" szakmájú ember osztály leírása	
<a href="#">creature::Bear</a>	28
A medve osztály leírása	
<a href="#">minerals::BerryBush</a>	31
A bokor osztály leírása. Ételt ad, ha kitermelik	
<a href="#">sf::Bound</a>	33
<a href="#">creature::Builder</a>	34
Az építész szakmájú ember osztály leírása	
<a href="#">ui::Button</a>	35
A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak	
<a href="#">minerals::CityCenter</a>	39
A városközpont osztály leírása. E köré épülnek a házak	
<a href="#">CityCenterException</a>	41
Akkor kell dobni, ha a városközpont hibásan működött	
<a href="#">sf::Clock</a>	42
<a href="#">sf::ClockTime</a>	42
<a href="#">sf::Color</a>	43
<a href="#">creature::Crocodile</a>	46
A krokodil osztály leírása	
<a href="#">creature::EntityBase</a>	49
Egy alap, nem rajzolható entitás osztálya	
<a href="#">EntityPlacer</a>	58
Az entitások a kattintással való lerakása	
<a href="#">sf::Event</a>	60
<a href="#">creature::Farmer</a>	61
A farmer szakmájú ember osztály leírása	
<a href="#">creature::Fisherman</a>	63
A halász szakmájú ember osztály leírása	
<a href="#">sf::FloatRect</a>	65
<a href="#">GameConfig</a>	67
A világ szimulációjának leírása	
<a href="#">GameManager</a>	72
A világ szimulálásáért és a kirazolás irányításáért felelős osztály	

<a href="#">creature::Goat</a>	
A kecske osztály leírása	75
<a href="#">creature::HostileInterface</a>	
A vadállat entitások interface leírása	77
<a href="#">minerals::House</a>	
A ház osztály leírása. Szinttől függően idéz embereket	81
<a href="#">creature::Human</a>	
Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik	84
<a href="#">HumanResources</a>	
Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva	89
<a href="#">ImportInvalidEntityException</a>	
Akkor kell dobni, ha egy entitás hibásan lett beolvasva	92
<a href="#">ImportInvalidHousingLevelException</a>	
Akkor kell dobni, ha egy ház hibásan lett beolvasva	93
<a href="#">ImportInvalidHumanProfessionException</a>	
Akkor kell dobni, ha egy szakma hibásan lett beolvasva	93
<a href="#">ImportInvalidResourceException</a>	
Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva	94
<a href="#">sf::IntRect</a>	95
<a href="#">InvalidBorderSizeException</a>	
Akkor kell dobni, ha egy világhatárnak nem jó értéket akarnak beállítani	96
<a href="#">minerals::Iron</a>	
A vasérc osztály leírása. Vasat ad, amikor kitermelik	97
<a href="#">sf::Keyboard</a>	99
<a href="#">creature::KillerRobot</a>	
A gyilkos robot osztály leírása	100
<a href="#">creature::King</a>	
A király szakmájú ember osztály leírása	103
<a href="#">creature::Living</a>	
Az élő entitások interface leírása	105
<a href="#">creature::LivingTexture</a>	
Az élő entitások kinézetének adatai	113
<a href="#">sf::Mouse</a>	115
<a href="#">sf::Music</a>	116
<a href="#">MusicLoadException</a>	
Akkor kell dobni, ha egy zene hibásan lett lejátszva vagy hibás a beolvasott zene	118
<a href="#">MusicPlayer</a>	
A zene játészó osztály leírása	119
<a href="#">ObjectRegistry</a>	
Az entitások és más világ objektumok lerakásának intézéséért felelős osztály	121
<a href="#">gtest_lite::ostreamRedir</a>	122
<a href="#">PostProcessor</a>	
A grafikus szépítő osztály leírása	122
<a href="#">Profession</a>	
A szakma osztály leírása	126
<a href="#">RandomGenerator</a>	
Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály	128
<a href="#">ReadSaveFileFail</a>	
Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás	130
<a href="#">sf::RectangleShape</a>	131
<a href="#">sf::RenderStates</a>	132
<a href="#">sf::RenderWindow</a>	133
<a href="#">minerals::ResourceStructure</a>	
Az erőforrás struktúra osztály leírása	136
<a href="#">RoleOption</a>	
Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni	139
<a href="#">SaveHelper</a>	
Factory-k	140

<a href="#">SaveManager</a>	
A fájl menedzseléshez szolgáló osztály leírása . . . . .	141
<a href="#">Shadowable</a>	
Az árnyékoláshoz szükséges interface . . . . .	143
<a href="#">SimulationException</a>	
Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik . . . . .	148
<a href="#">creature::Soldier</a>	
A katona szakmájú ember osztály leírása . . . . .	149
<a href="#">sf::Sound</a>	150
<a href="#">sf::SoundBuffer</a>	151
<a href="#">SoundPlayer</a>	
A hanglejátszó osztály leírása . . . . .	152
<a href="#">sf::SoundSource</a>	153
<a href="#">sf::Sprite</a>	155
<a href="#">minerals::Stone</a>	
A kő osztály leírása. követ ad, amikor kitermelik . . . . .	158
<a href="#">creature::Stonemason</a>	
A bányász szakmájú ember osztály leírása . . . . .	159
<a href="#">minerals::Structure</a>	
A struktúra osztály leírása . . . . .	162
<a href="#">StructureException</a>	
Akkor kell dobni, ha egy struktúra hibásan működött . . . . .	166
<a href="#">TerrainContainer&lt; T &gt;</a>	
A világ terepét tároló osztály . . . . .	167
<a href="#">gtest_lite::Test</a>	174
<a href="#">sf::Texture</a>	178
<a href="#">Textureable</a>	
Egy interface, ami a textúrázáshoz kell . . . . .	179
<a href="#">TextureManager</a>	
A Textúra kezelő osztály . . . . .	181
<a href="#">tiles::Tile</a>	
A terepkocka osztály leírása . . . . .	183
<a href="#">sf::Transform</a>	185
<a href="#">minerals::Tree</a>	
A fa osztály leírása. Fát ad, ha kitermelik . . . . .	187
<a href="#">sf::Vector2f</a>	189
<a href="#">sf::Vector2i</a>	190
<a href="#">sf::VideoMode</a>	191
<a href="#">creature::Woodcutter</a>	
A favágó szakmájú ember osztály leírása . . . . .	193
<a href="#">World</a>	
A világ osztály leírása . . . . .	194
<a href="#">WorldBase</a>	
A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza . . . . .	199
<a href="#">WorldBaseSerialiazble</a>	
A világ osztály bővítése, rendelkezik insertorral és extractorral . . . . .	207
<a href="#">YAMLParse</a>	
Egy YAML (Yet Another Markup Language) fájl beolvasó osztály . . . . .	209



## 4. fejezet

# Fájlmutató

### 4.1. Fájllista

Az összes fájl listája rövid leírásokkal:

src/EntityPlacer.cpp	222
src/EntityPlacer.d	222
src/EntityPlacer.h	222
src/GameConfig.cpp	235
src/GameConfig.d	235
src/GameConfig.h	235
src/GameManager.cpp	235
src/GameManager.d	236
src/GameManager.h	236
src/HumanResources.cpp	236
src/HumanResources.d	236
src/HumanResources.h	236
src/main.cpp	236
src/main.d	237
src/MusicPlayer.cpp	237
src/MusicPlayer.d	237
src/MusicPlayer.h	237
src/PostProcessor.cpp	237
src/PostProcessor.d	237
src/PostProcessor.h	237
src/Profession.cpp	238
src/Profession.d	238
src/Profession.h	238
src/Random_Gen.cpp	238
src/Random_Gen.d	238
src/Random_Gen.h	238
src/SaveHelpers.cpp	238
src/SaveHelpers.d	239
src/SaveHelpers.h	239
src/SaveManager.cpp	239
src/SaveManager.d	240
src/SaveManager.h	240
src/Shadowable.cpp	240
src/Shadowable.d	240
src/Shadowable.h	240

src/SoundPlayer.cpp	240
src/SoundPlayer.d	240
src/SoundPlayer.h	240
src/TerrainContainer.hpp	
A Világ terepének a deklarálása ebben a fájlba van	241
src/TerrainContainer_def.hpp	242
src/Textureable.h	242
src/TextureManager.cpp	242
src/TextureManager.d	242
src/TextureManager.h	242
src/Utils.cpp	243
src/Utils.d	244
src/Utils.h	244
src/World.cpp	246
src/World.d	246
src/World.hpp	
A Világ osztály, ami a fő szimulációs elemek tárolásáért felelős	246
src/WorldBase.cpp	251
src/WorldBase.d	252
src/WorldBaseSerializable.cpp	252
src/WorldBaseSerializable.d	252
src/YAMLParser.cpp	252
src/YAMLParser.d	252
src/YAMLParser.h	252
src/creatures/EntityBase.cpp	211
src/creatures/EntityBase.d	211
src/creatures/EntityBase.h	211
src/creatures/EntityUtils.h	212
src/creatures/Goat.cpp	212
src/creatures/Goat.d	212
src/creatures/Goat.h	212
src/creatures/HostileInterface.cpp	213
src/creatures/HostileInterface.d	213
src/creatures/HostileInterface.h	213
src/creatures/Living.cpp	221
src/creatures/Living.d	222
src/creatures/Living.h	222
src/creatures/hostiles/Bear.cpp	213
src/creatures/hostiles/Bear.d	214
src/creatures/hostiles/Bear.h	214
src/creatures/hostiles/Crocodile.cpp	214
src/creatures/hostiles/Crocodile.d	214
src/creatures/hostiles/Crocodile.h	214
src/creatures/hostiles/KillerRobot.cpp	215
src/creatures/hostiles/KillerRobot.d	215
src/creatures/hostiles/KillerRobot.h	215
src/creatures/humans/AnglerMiner.cpp	215
src/creatures/humans/AnglerMiner.d	216
src/creatures/humans/AnglerMiner.h	216
src/creatures/humans/Builder.cpp	216
src/creatures/humans/Builder.d	216
src/creatures/humans/Builder.h	216
src/creatures/humans/Farmer.cpp	217
src/creatures/humans/Farmer.d	217
src/creatures/humans/Farmer.h	217
src/creatures/humans/Fisherman.cpp	217
src/creatures/humans/Fisherman.d	218
src/creatures/humans/Fisherman.h	218



src/creatures/humans/Human.cpp	218
src/creatures/humans/Human.d	218
src/creatures/humans/Human.h	218
src/creatures/humans/King.cpp	219
src/creatures/humans/King.d	219
src/creatures/humans/King.h	219
src/creatures/humans/Soldier.cpp	219
src/creatures/humans/Soldier.d	220
src/creatures/humans/Soldier.h	220
src/creatures/humans/Stonemason.cpp	220
src/creatures/humans/Stonemason.d	220
src/creatures/humans/Stonemason.h	220
src/creatures/humans/Woodcutter.cpp	221
src/creatures/humans/Woodcutter.d	221
src/creatures/humans/Woodcutter.h	221
src/exceptions/FileExceptions.h	223
src/exceptions/MusicLoadException.h	223
src/exceptions/SimulationException.h	224
src/exceptions/WorldExceptions.h	224
src/external/memtrace.cpp	224
src/external/memtrace.h	224
src/external/modified_gtest_lite.h	224
src/fake_sfml/fake_sfml.cpp	233
src/fake_sfml/fake_sfml.d	234
src/fake_sfml/fake_sfml.h	234
src/terrain_tiles/Tile.cpp	241
src/terrain_tiles/Tile.d	241
src/terrain_tiles/Tile.h	241
src/ui/button.cpp	243
src/ui/button.d	243
src/ui/button.h	243
src/world_object/BerryBush.cpp	247
src/world_object/BerryBush.d	247
src/world_object/BerryBush.h	247
src/world_object/CityCenter.cpp	247
src/world_object/CityCenter.d	248
src/world_object/CityCenter.h	248
src/world_object/House.cpp	248
src/world_object/House.d	248
src/world_object/House.h	248
src/world_object/Iron.cpp	249
src/world_object/Iron.d	249
src/world_object/Iron.h	249
src/world_object/ResourceStructure.cpp	249
src/world_object/ResourceStructure.d	249
src/world_object/ResourceStructure.h	249
src/world_object/Stone.cpp	250
src/world_object/Stone.d	250
src/world_object/Stone.h	250
src/world_object/Structure.cpp	250
src/world_object/Structure.d	250
src/world_object/Structure.h	250
src/world_object/Tree.cpp	251
src/world_object/Tree.d	251
src/world_object/Tree.h	251



## 5. fejezet

# Névterek dokumentációja

### 5.1. creature névtér-referencia

Az összes élőlény és entitás ebben a névtérben van.

#### Osztályok

- class [EntityBase](#)  
*Egy alap, nem rajzolható entitás osztálya.*
- class [LivingTexture](#)  
*Az élő entitások kinézetének adatai.*
- class [Goat](#)  
*A kecske osztály leírása.*
- class [HostileInterface](#)  
*A vadállat entítások interface leírása.*
- class [Bear](#)  
*A medve osztály leírása.*
- class [Crocodile](#)  
*A krokodil osztály leírása.*
- class [KillerRobot](#)  
*A gyilkos robot osztály leírása.*
- class [AnglerMiner](#)  
*Az "AnglerMiner" szakmájú ember osztály leírása.*
- class [Builder](#)  
*Az építész szakmájú ember osztály leírása.*
- class [Farmer](#)  
*A farmer szakmájú ember osztály leírása.*
- class [Fisherman](#)  
*A halász szakmájú ember osztály leírása.*
- class [Human](#)  
*Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik.*
- class [King](#)  
*A király szakmájú ember osztály leírása.*
- class [Soldier](#)  
*A katona szakmájú ember osztály leírása.*

- class [Stonemason](#)  
*A bányász szakmájú ember osztály leírása.*
- class [Woodcutter](#)  
*A favágó szakmájú ember osztály leírása.*
- class [Living](#)  
*Az élő entitások interface leírása.*

## Enumerációk

- enum class [ENTITY\\_TYPE](#) : char { [HUMAN](#) , [ANIMAL](#) , [ROBOTIC](#) }
- enum class [ENTITY\\_GENDER](#) : char { [MALE](#) , [FEMALE](#) }
- enum class [FACING](#) : bool { [RIGHT](#) , [LEFT](#) }
- enum class [LIVINGSTATE](#) : int {  
    [IDLE](#) , [RUN](#) , [WALK](#) , [DEATH](#) ,  
    [ATTACKING](#) , [DOING\\_ITS\\_WORK](#) }

### 5.1.1. Részletes leírás

Az összes élőlény és entitás ebben a névtérben van.

### 5.1.2. Enumerációk dokumentációja

#### 5.1.2.1. ENTITY\_GENDER

```
enum creature::ENTITY_GENDER : char [strong]
```

Enumeráció-értékek

MALE	
FEMALE	

#### 5.1.2.2. ENTITY\_TYPE

```
enum creature::ENTITY_TYPE : char [strong]
```

Enumeráció-értékek

HUMAN	
ANIMAL	
ROBOTIC	

### 5.1.2.3. FACING

```
enum creature::FACING : bool [strong]
```

Enumeráció-értékek

RIGHT	
LEFT	

### 5.1.2.4. LIVINGSTATE

```
enum creature::LIVINGSTATE : int [strong]
```

Enumeráció-értékek

IDLE	
RUN	
WALK	
DEATH	
ATTACKING	
DOING_ITS_WORK	

## 5.2. creatures névtér-referencia

## 5.3. gtest\_lite névtér-referencia

[gtest\\_lite](#): a keretrendszer függvényinek és objektumainak névtére

### Osztályok

- struct [Test](#)
- class [ostreamRedir](#)

### Függvények

- `template<typename T1 , typename T2 >`  
`std::ostream & EXPECT_ (T1 exp, T2 act, bool(*pred)(T1, T1), const char *file, int line, const char *expr,`  
`const char *lhs="elvart", const char *rhs="aktual")`  
*általános sablon a várt értékhez.*

- `template<typename T1, typename T2 >`  
`std::ostream & EXPECT_ (T1 *exp, T2 *act, bool(*pred)(T1 *, T1 *), const char *file, int line, const char *expr, const char *lhs="elvart", const char *rhs="aktual")`  
*pointerre specializált sablon a várt értékhez.*
- `std::ostream & EXPECTSTR (const char *exp, const char *act, bool(*pred)(const char *, const char *), const char *file, int line, const char *expr, const char *lhs="elvart", const char *rhs="aktual")`
- `template<typename T >`  
`bool eq (T a, T b)`
- `bool eqstr (const char *a, const char *b)`
- `bool eqstrcase (const char *a, const char *b)`
- `template<typename T >`  
`bool ne (T a, T b)`
- `bool nestr (const char *a, const char *b)`
- `template<typename T >`  
`bool le (T a, T b)`
- `template<typename T >`  
`bool lt (T a, T b)`
- `template<typename T >`  
`bool ge (T a, T b)`
- `template<typename T >`  
`bool gt (T a, T b)`
- `template<typename T >`  
`bool almostEQ (T a, T b)`

### 5.3.1. Részletes leírás

`gtest_lite`: a keretrendszer függvényeinek és objektumainak névtére

### 5.3.2. Függvények dokumentációja

#### 5.3.2.1. almostEQ()

```
template<typename T >
bool gtest_lite::almostEQ (
    T a,
    T b )
```

Segédsablon valós számok összehasonlításához Nem bombabiztos, de nekünk most jó lesz Elméleti hátér:  
<http://www.cygnus-software.com/papers/comparingfloats/comparingfloats.htm>

#### 5.3.2.2. eq()

```
template<typename T >
bool gtest_lite::eq (
    T a,
    T b )
```

segéd sablonok a relációkhoz. azért nem STL (algorithm), mert csak a függvény lehet, hogy menjen a deduckció

### 5.3.2.3. eqstr()

```
bool gtest_lite::eqstr (
    const char * a,
    const char * b ) [inline]
```

### 5.3.2.4. eqstrcase()

```
bool gtest_lite::eqstrcase (
    const char * a,
    const char * b ) [inline]
```

### 5.3.2.5. EXPECT\_() [1/2]

```
template<typename T1 , typename T2 >
std::ostream& gtest_lite::EXPECT_ (
    T1 * exp,
    T2 * act,
    bool(*) (T1 *, T1 *) pred,
    const char * file,
    int line,
    const char * expr,
    const char * lhs = "elvart",
    const char * rhs = "aktual" )
```

pointerre specializált sablon a várt értékhez.

### 5.3.2.6. EXPECT\_() [2/2]

```
template<typename T1 , typename T2 >
std::ostream& gtest_lite::EXPECT_ (
    T1 exp,
    T2 act,
    bool(*) (T1, T1) pred,
    const char * file,
    int line,
    const char * expr,
    const char * lhs = "elvart",
    const char * rhs = "aktual" )
```

általános sablon a várt értékhez.

### 5.3.2.7. EXPECTSTR()

```
std::ostream& gtest_lite::EXPECTSTR (
    const char * exp,
    const char * act,
    bool(*) (const char *, const char *) pred,
    const char * file,
    int line,
    const char * expr,
    const char * lhs = "elvart",
    const char * rhs = "aktual" ) [inline]
```

stringek összehasonlításához. azért nem spec. mert a sima EQ-ra másként kell működnie.

### 5.3.2.8. ge()

```
template<typename T >
bool gtest_lite::ge (
    T a,
    T b )
```

### 5.3.2.9. gt()

```
template<typename T >
bool gtest_lite::gt (
    T a,
    T b )
```

### 5.3.2.10. le()

```
template<typename T >
bool gtest_lite::le (
    T a,
    T b )
```

### 5.3.2.11. lt()

```
template<typename T >
bool gtest_lite::lt (
    T a,
    T b )
```



**5.3.2.12. ne()**

```
template<typename T >
bool gtest_lite::ne (
    T a,
    T b )
```

**5.3.2.13. nestr()**

```
bool gtest_lite::nestr (
    const char * a,
    const char * b ) [inline]
```

**5.4. minerals névtér-referencia**

Az összes struktúra ebben a névtérben van.

**Osztályok**

- class [BerryBush](#)  
*A bokor osztály leírása. Ételet ad, ha kitermelik.*
- class [CityCenter](#)  
*A városközpont osztály leírása. E köré épülnek a házak.*
- class [House](#)  
*A ház osztály leírása. Szinttől függően idéz embereket.*
- class [Iron](#)  
*A vasérc osztály leírása. Vasat ad, amikor kitermelik.*
- class [ResourceStructure](#)  
*Az erőforrás struktúra osztály leírása.*
- class [Stone](#)  
*A kő osztály leírása. követ ad, amikor kitermelik.*
- class [Structure](#)  
*A struktúra osztály leírása.*
- class [Tree](#)  
*A fa osztály leírása. Fát ad, ha kitermelik.*

**Enumerációk**

- enum class [MINERAL\\_TYPE](#) : char {  
    [STONE](#) , [WOOD](#) , [IRON](#) , [FOOD](#) ,  
    [HOUSING](#) , [CITY\\_CENTER](#) }

**Függvények**

- std::string [mineral\\_to\\_string](#) ([MINERAL\\_TYPE](#) type)  
*Mentést elősegítő függvények.*

### 5.4.1. Részletes leírás

Az összes struktúra ebben a névtérben van.

### 5.4.2. Enumerációk dokumentációja

#### 5.4.2.1. MINERAL\_TYPE

```
enum minerals::MINERAL_TYPE : char [strong]
```

Enumeráció-értékek

STONE	
WOOD	
IRON	
FOOD	
HOUSING	
CITY_CENTER	

### 5.4.3. Függvények dokumentációja

#### 5.4.3.1. mineral\_to\_string()

```
std::string minerals::mineral_to_string (
    MINERAL_TYPE type )
```

Mentést elősegítő függvények.

## 5.5. sf névtér-referencia

### Osztályok

- class [Vector2f](#)
- class [Transform](#)
- class [FloatRect](#)
- class [Vector2i](#)
- class [Texture](#)
- class [Bound](#)
- class [Color](#)
- class [IntRect](#)
- class [Sprite](#)

- class [Event](#)
- class [ClockTime](#)
- class [Clock](#)
- class [SoundBuffer](#)
- class [Sound](#)
- class [SoundSource](#)
- class [Music](#)
- class [RectangleShape](#)
- class [Keyboard](#)
- class [RenderStates](#)
- class [VideoMode](#)
- class [RenderWindow](#)
- class [Mouse](#)

## Enumerációk

- enum class [BlendMode](#) {  
[None](#) , [Alpha](#) , [Additive](#) , [Multiply](#) ,  
[BlendAdd](#) }

## Függvények

- std::string [to\\_string](#) (const [Color](#) &c)
- bool [file\\_exists\\_at\\_path](#) (const std::string &name)

## Változók

- constexpr [BlendMode BlendAdd](#) = BlendMode::BlendAdd

### 5.5.1. Enumerációk dokumentációja

#### 5.5.1.1. BlendMode

```
enum sf::BlendMode [strong]
```

##### Enumeráció-értékek

None	
Alpha	
Additive	
Multiply	
BlendAdd	

## 5.5.2. Függvények dokumentációja

### 5.5.2.1. file\_exists\_at\_path()

```
bool sf::file_exists_at_path (
    const std::string & name ) [inline]
```

### 5.5.2.2. to\_string()

```
std::string sf::to_string (
    const Color & c ) [inline]
```

## 5.5.3. Változók dokumentációja

### 5.5.3.1. BlendAdd

```
constexpr BlendMode sf::BlendAdd = BlendMode::BlendAdd [inline], [constexpr]
```

## 5.6. tiles névtér-referencia

Az összes terepkocka elem ebben a névtérben van.

### Osztályok

- class [Tile](#)  
*A terepkocka osztály leírása.*

### Enumerációk

- enum class [TILETYPE](#) : char { [GRASS](#) , [WATER](#) , [MOUNTAIN](#) }

### 5.6.1. Részletes leírás

Az összes terepkocka elem ebben a névtérben van.

## 5.6.2. Enumerációk dokumentációja

### 5.6.2.1. TILETYPE

```
enum tiles::TILETYPE : char [strong]
```

## Enumeráció-értékek

GRASS	
WATER	
MOUNTAIN	

## 5.7. ui névtér-referencia

Az összes UI elem ebben a névtérben van.

### Osztályok

- class [Button](#)

*A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak.*

#### 5.7.1. Részletes leírás

Az összes UI elem ebben a névtérben van.



## 6. fejezet

# Osztályok dokumentációja

### 6.1. `_Is_Types< F, T >` struktúrasablon-referencia

Segédsablon típuskonverzió futás közbeni ellenőrzésére.

```
#include <modified_gtest_lite.h>
```

#### Statikus publikus tagfüggvények

- `template<typename D >`  
`static char(& f (D))[1]`
- `template<typename D >`  
`static char(& f (...))[2]`

#### Statikus publikus attribútumok

- `static bool const convertible = sizeof(f<T>(F())) == 1`

#### 6.1.1. Részletes leírás

```
template<typename F, typename T>  
struct _Is_Types< F, T >
```

Segédsablon típuskonverzió futás közbeni ellenőrzésére.

#### 6.1.2. Tagfüggvények dokumentációja

### 6.1.2.1. f() [1/2]

```
template<typename F , typename T >
template<typename D >
static char(& _Is_Types< F, T >::f (
    ... )) [2] [static]
```

### 6.1.2.2. f() [2/2]

```
template<typename F , typename T >
template<typename D >
static char(& _Is_Types< F, T >::f (
    D )) [1] [static]
```

## 6.1.3. Adattagok dokumentációja

### 6.1.3.1. convertible

```
template<typename F , typename T >
bool const _Is_Types< F, T >::convertible = sizeof(f<T>(F())) == 1 [static]
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

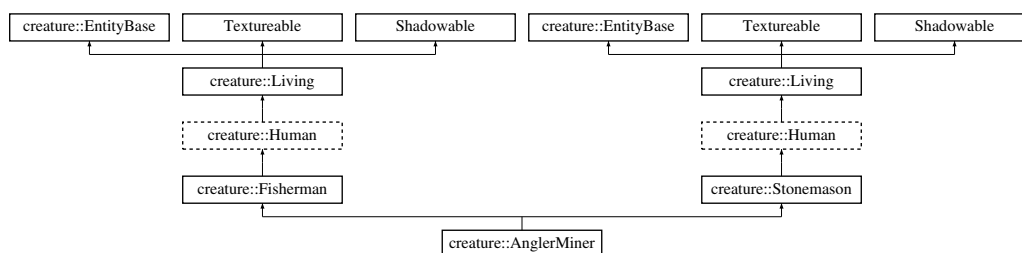
- [src/external/modified\\_gtest\\_lite.h](#)

## 6.2. creature::AnglerMiner osztályreferencia

Az "AnglerMiner" szakmájú ember osztály leírása.

```
#include <AnglerMiner.h>
```

A creature::AnglerMiner osztály származási diagramja:





## Publikus tagfüggvények

- `AnglerMiner` (int x, int y, `ENTITY_GENDER` gender\_modifier)  
*Inicializál egy AnglerMiner-t egy pontos x és y koordinátára és beállítja az attribútumait.*
- void `update_logic` (`World` &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- `~AnglerMiner` ()  
*Az AnglerMiner destruktora.*

## További örökölt tagok

### 6.2.1. Részletes leírás

Az "AnglerMiner" szakmájú ember osztály leírása.

Ez egy speciális szakma, ami tud követ és vasat bányászni és ha akar, még halászni is tud.

### 6.2.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.2.2.1. AnglerMiner()

```
creature::AnglerMiner::AnglerMiner (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy AnglerMiner-t egy pontos x és y koordinátára és beállítja az attribútumait.

#### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	Az "AnglerMiner" neve.

#### 6.2.2.2. ~AnglerMiner()

```
creature::AnglerMiner::~~AnglerMiner ( )
```

Az `AnglerMiner` destruktora.

### 6.2.3. Tagfüggvények dokumentációja

### 6.2.3.1. update\_logic()

```
void creature::AnglerMiner::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

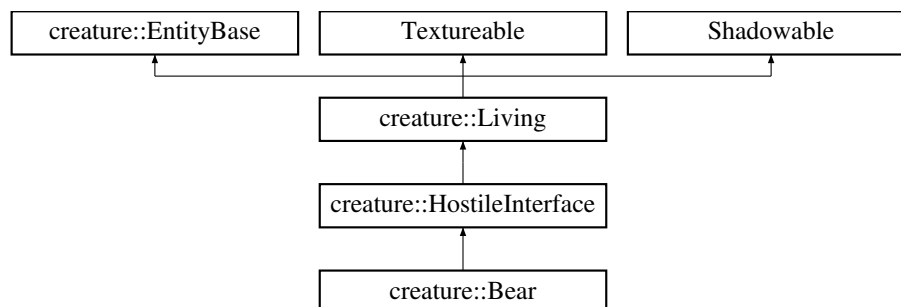
- src/creatures/humans/[AnglerMiner.h](#)
- src/creatures/humans/[AnglerMiner.cpp](#)

## 6.3. creature::Bear osztályreferencia

A medve osztály leírása.

```
#include <Bear.h>
```

A creature::Bear osztály származási diagramja:



### Publikus tagfüggvények

- [Bear](#) (int x, int y)  
*Idéz egy medvét egy pontos x és y koordinátára és beállítja az attribútumait.*
- [ENTITY\\_TYPE get\\_type](#) () const override  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- void [die](#) () override  
*Mi történjen, ha meghal az entitás.*
- void [update\\_logic](#) (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- void [draw\\_logic](#) (sf::RenderWindow &window, float deltaTime, int ofx, int ofy) override  
*Az entitás kirajzolási logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*
- void [select\\_target](#) (World &world) override  
*A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.*
- [~Bear](#) ()  
*A medve destruktora.*

## További örökölt tagok

### 6.3.1. Részletes leírás

A medve osztály leírása.

A medve egy agresszív állat, ami más medvéken kívül mindent támad. Gyorsan fut.

### 6.3.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.3.2.1. Bear()

```
creature::Bear::Bear (
    int x,
    int y )
```

Idéz egy medvét egy pontos x és y koordinátára és beállítja az attribútumait.

##### Paraméterek

x	Az x koordináta.
y	Az y koordináta.

#### 6.3.2.2. ~Bear()

```
creature::Bear::~~Bear ( )
```

A medve destruktora.

### 6.3.3. Tagfüggvények dokumentációja

#### 6.3.3.1. die()

```
void creature::Bear::die ( ) [override], [virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.3.3.2. draw\_logic()

```
void creature::Bear::draw_logic (
    sf::RenderWindow & window,
    float deltaTime,
    int offx,
    int offy ) [override], [virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítja a következőket: [creature::Living](#).

### 6.3.3.3. get\_type()

```
ENTITY_TYPE creature::Bear::get_type ( ) const [override], [virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

#### Visszatérési érték

A belső szimbólum.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.3.3.4. select\_target()

```
void creature::Bear::select_target (
    World & world ) [override], [virtual]
```

A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.

#### Paraméterek

<i>world</i>	A világ, amibe a célpontot kell választani.
--------------	---

Megvalósítja a következőket: [creature::HostileInterface](#).

### 6.3.3.5. update\_logic()

```
void creature::Bear::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

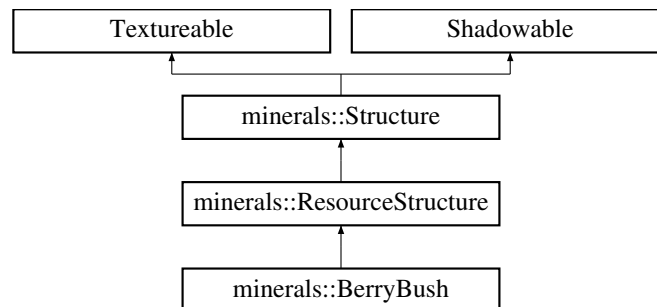
- [src/creatures/hostiles/Bear.h](#)
- [src/creatures/hostiles/Bear.cpp](#)

## 6.4. minerals::BerryBush osztályreferencia

A bokor osztály leírása. Ételt ad, ha kitermelik.

```
#include <BerryBush.h>
```

A minerals::BerryBush osztály származási diagramja:



### Publikus tagfüggvények

- [BerryBush](#) (int x, int y)  
*Konstruktor ami lerakja az erőforrást egy (x,y) pontra.*
- [MINERAL\\_TYPE get\\_type](#) () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void [update\\_logic](#) (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- bool [harvest](#) () override  
*virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.*
- void [play\\_destroy\\_sound](#) ([SoundPlayer](#) &sound\_player) const override

## További örökölt tagok

### 6.4.1. Részletes leírás

A bokr osztály leírása. Ételt ad, ha kitermelik.

### 6.4.2. Konstruktorok és destruktorok dokumentációja

#### 6.4.2.1. BerryBush()

```
minerals::BerryBush::BerryBush (
    int x,
    int y )
```

Konstruktor ami lerakja az erőforrást egy (x,y) pontra.

### 6.4.3. Tagfüggvények dokumentációja

#### 6.4.3.1. get\_type()

```
MINERAL_TYPE minerals::BerryBush::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.4.3.2. harvest()

```
bool minerals::BerryBush::harvest ( ) [override], [virtual]
```

virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.

Újraimplementált ősök: [minerals::ResourceStructure](#).

#### 6.4.3.3. play\_destroy\_sound()

```
void minerals::BerryBush::play_destroy_sound (
    SoundPlayer & sound_player ) const [override], [virtual]
```

Megvalósítja a következőket: [minerals::ResourceStructure](#).

#### 6.4.3.4. update\_logic()

```
void minerals::BerryBush::update_logic (
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.

## Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/world\\_object/BerryBush.h](#)
- [src/world\\_object/BerryBush.cpp](#)

## 6.5. sf::Bound osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus attribútumok

- `int width`
- `int height`

### 6.5.1. Adattagok dokumentációja

#### 6.5.1.1. height

```
int sf::Bound::height
```

#### 6.5.1.2. width

```
int sf::Bound::width
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

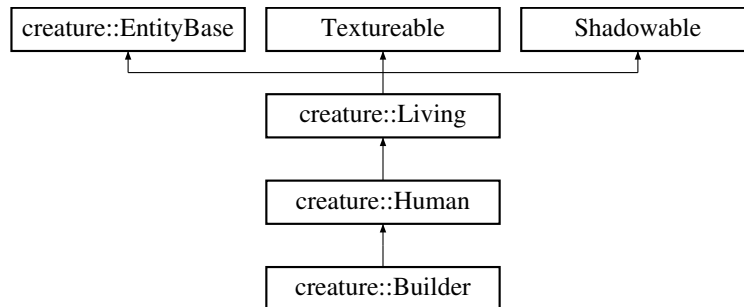
- [src/fake\\_sfml/fake\\_sfml.h](#)

## 6.6. creature::Builder osztályreferencia

Az építész szakmájú ember osztály leírása.

```
#include <Builder.h>
```

A creature::Builder osztály származási diagramja:



### Publikus tagfüggvények

- **Builder** (int x, int y, ENTITY\_GENDER gender\_modifier)  
*Inicializál egy építészt egy pontos x és y koordinátára és beállítja az attribútumait.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- **~Builder** ()  
*Az építész destruktora.*

### További örökölt tagok

#### 6.6.1. Részletes leírás

Az építész szakmájú ember osztály leírása.

Ez a szakmájú ember épületeket fejleszt magasabb szintekre. Ha nincs épület akkor épít még.

#### 6.6.2. Konstruktorok és destruktorok dokumentációja

##### 6.6.2.1. Builder()

```
creature::Builder::Builder (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy építészt egy pontos x és y koordinátára és beállítja az attribútumait.



## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	Az építész neme.

## 6.6.2.2. ~Builder()

```
creature::Builder::~~Builder ( )
```

Az építész destruktora.

## 6.6.3. Tagfüggvények dokumentációja

## 6.6.3.1. update\_logic()

```
void creature::Builder::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

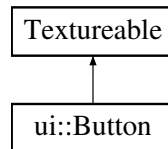
- [src/creatures/humans/Builder.h](#)
- [src/creatures/humans/Builder.cpp](#)

## 6.7. ui::Button osztályreferencia

A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak.

```
#include <button.h>
```

Az ui::Button osztály származási diagramja:



## Publikus tagfüggvények

- **Button** (int px, int py, int width, int height, const std::string &spritepath)  
*A konstruktor ami létrehozza a gombot megadott mérettel és képpel.*
- void **setCallback** (std::function< void()> func)  
*Beállítja, mi történjen, ha a gombra kattintanak.*
- void **try\_hover\_animation** (int mX, int mY)  
*Megnézi, hogy az egér kurzor rajta van-e a gombon.*
- void **onClick** (bool mc)  
*Megnézi, hogy kattintottak-e rá, ha igen akkor végrehajta a függvényt amit neki adtak be.*
- bool **setTexture** (const std::string &filename) override  
*Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- void **setPosition** (double x, double y) override  
*Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.*
- void **draw** (sf::RenderWindow &window) override  
*Kirajzolja az objektumot.*

### 6.7.1. Részletes leírás

A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak.

### 6.7.2. Konstruktorok és destruktorok dokumentációja

#### 6.7.2.1. Button()

```

ui::Button::Button (
    int px,
    int py,
    int width,
    int height,
    const std::string & spritepath )
  
```

A konstruktor ami létrehozza a gombot megadott mérettel és képpel.

#### Paraméterek

<i>px</i>	A gomb X koordinátája.
<i>py</i>	A gomb Y koordinátája.
<i>width</i>	A gomb szélessége.
<i>height</i>	A gomb magassága.
<i>spritepath</i>	A gomb képének elérési útvonala.

### 6.7.3. Tagfüggvények dokumentációja

#### 6.7.3.1. draw()

```
void ui::Button::draw (
    sf::RenderWindow & window ) [override], [virtual]
```

Kirajzolja az objektumot.

##### Paraméterek

<i>window</i>	Ahova ki kell rajzolni a textúrázható leszármazottat.
---------------	---

Megvalósítja a következőket: [Textureable](#).

#### 6.7.3.2. onClick()

```
void ui::Button::onClick (
    bool mc )
```

Megnézi, hogy kattintottak-e rá, ha igen akkor végrehajta a függvényt amit neki adtak be.

##### Paraméterek

<i>mc</i>	Le van-e nyomva az egér gomb.
-----------	-------------------------------

#### 6.7.3.3. setCallback()

```
void ui::Button::setCallback (
    std::function< void()> func )
```

Beállítja, mi történjen, ha a gombra kattintanak.

##### Paraméterek

<i>func</i>	A függvény, ami le fog futni.
-------------	-------------------------------

#### 6.7.3.4. setPosition()

```
void ui::Button::setPosition (
    double x,
    double y ) [override], [virtual]
```

Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.

##### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

Megvalósítja a következőket: [Textureable](#).

#### 6.7.3.5. setTexture()

```
bool ui::Button::setTexture (
    const std::string & filename ) [override], [virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

##### Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

##### Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítja a következőket: [Textureable](#).

#### 6.7.3.6. try\_hover\_animation()

```
void ui::Button::try_hover_animation (
    int mX,
    int mY )
```

Megnézi, hogy az egér kurzor rajta van-e a gombon.

##### Paraméterek

<i>mX</i>	A kurzor X koordinátája.
<i>mY</i>	A kurzor Y koordinátája.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

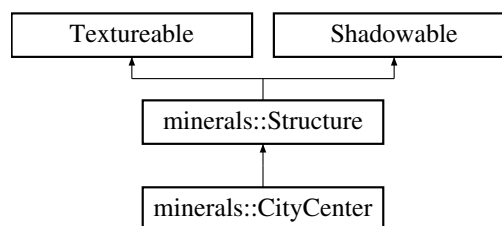
- [src/ui/button.h](#)
- [src/ui/button.cpp](#)

## 6.8. minerals::CityCenter osztályreferencia

A városközpont osztály leírása. E köré épülnek a házak.

```
#include <CityCenter.h>
```

A minerals::CityCenter osztály származási diagramja:



### Publikus tagfüggvények

- [CityCenter](#) (int x, int y)  
*Konstruktor ami lerakja a házat egy (x,y) pontra.*
- bool [is\\_there\\_room\\_for\\_housing](#) ()  
*Igazat ad vissza, ha lehet még házat építeni köré.*
- void [register\\_new\\_house](#) ()  
*Új házat vesz fel a városhoz.*
- [MINERAL\\_TYPE](#) [get\\_type](#) () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void [update\\_logic](#) (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- std::string [get\\_settlement\\_age](#) ()  
*String-ként adja vissza azt, hogy hány másodperces a város.*

### További örökölt tagok

#### 6.8.1. Részletes leírás

A városközpont osztály leírása. E köré épülnek a házak.

#### 6.8.2. Konstruktorok és destruktorok dokumentációja

#### 6.8.2.1. CityCenter()

```
minerals::CityCenter::CityCenter (
    int x,
    int y )
```

Konstruktor ami lerakja a házat egy (x,y) pontra.

### 6.8.3. Tagfüggvények dokumentációja

#### 6.8.3.1. get\_settlement\_age()

```
std::string minerals::CityCenter::get_settlement_age ( )
```

String-ként adja vissza azt, hogy hány másodperces a város.

#### 6.8.3.2. get\_type()

```
MINERAL\_TYPE minerals::CityCenter::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.8.3.3. is\_there\_room\_for\_housing()

```
bool minerals::CityCenter::is_there_room_for_housing ( )
```

Igazat ad vissza, ha lehet még házat építeni köré.

#### 6.8.3.4. register\_new\_house()

```
void minerals::CityCenter::register_new_house ( )
```

Új házat vesz fel a városhoz.

#### 6.8.3.5. update\_logic()

```
void minerals::CityCenter::update_logic (
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.

## Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

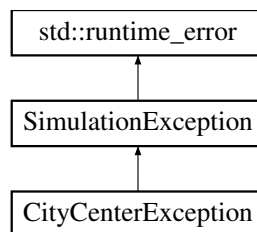
- `src/world_object/CityCenter.h`
- `src/world_object/CityCenter.cpp`

## 6.9. CityCenterException osztályreferencia

Akkor kell dobni, ha a városközpont hibásan működött.

```
#include <WorldExceptions.h>
```

A CityCenterException osztály származási diagramja:



### Publikus tagfüggvények

- [CityCenterException](#) (const std::string &msg)

#### 6.9.1. Részletes leírás

Akkor kell dobni, ha a városközpont hibásan működött.

#### 6.9.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.9.2.1. CityCenterException()

```
CityCenterException::CityCenterException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- `src/exceptions/WorldExceptions.h`

## 6.10. sf::Clock osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- void [restart](#) ()
- [ClockTime](#) & [getElapsedTime](#) ()

### 6.10.1. Tagfüggvények dokumentációja

#### 6.10.1.1. getElapsedTime()

```
ClockTime & sf::Clock::getElapsedTime ( )
```

#### 6.10.1.2. restart()

```
void sf::Clock::restart ( )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/fake\_sfml/[fake\\_sfml.h](#)
- src/fake\_sfml/[fake\\_sfml.cpp](#)

## 6.11. sf::ClockTime osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [ClockTime](#) (std::size\_t atime)
- [ClockTime](#) ()
- float [asSeconds](#) ()
- void [reset](#) ()
- void [increment](#) ()

### 6.11.1. Konstruktorkok és destruktorkok dokumentációja



#### 6.11.1.1. ClockTime() [1/2]

```
sf::ClockTime::ClockTime (
    std::size_t atime )
```

#### 6.11.1.2. ClockTime() [2/2]

```
sf::ClockTime::ClockTime ( )
```

### 6.11.2. Tagfüggvények dokumentációja

#### 6.11.2.1. asSeconds()

```
float sf::ClockTime::asSeconds ( )
```

#### 6.11.2.2. increment()

```
void sf::ClockTime::increment ( )
```

#### 6.11.2.3. reset()

```
void sf::ClockTime::reset ( )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.12. sf::Color osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [Color](#) ()
- [Color](#) (int \_r, int \_g, int \_b)
- [Color](#) (int \_r, int \_g, int \_b, int \_a)

## Publikus attribútumok

- int `r`
- int `g`
- int `b`
- int `a`

## Statikus publikus attribútumok

- static const `Color Black` = `Color(0, 0, 0)`
- static const `Color White` = `Color(255, 255, 255)`
- static const `Color Red` = `Color(255, 0, 0)`
- static const `Color Green` = `Color(0, 255, 0)`
- static const `Color Blue` = `Color(0, 0, 255)`
- static const `Color Transparent` = `Color(0, 0, 0, 0)`

## 6.12.1. Konstruktorkok és destruktorkok dokumentációja

### 6.12.1.1. `Color()` [1/3]

```
sf::Color::Color ( )
```

### 6.12.1.2. `Color()` [2/3]

```
sf::Color::Color (
    int _r,
    int _g,
    int _b )
```

### 6.12.1.3. `Color()` [3/3]

```
sf::Color::Color (
    int _r,
    int _g,
    int _b,
    int _a )
```

## 6.12.2. Adattagok dokumentációja

**6.12.2.1. a**

```
int sf::Color::a
```

**6.12.2.2. b**

```
int sf::Color::b
```

**6.12.2.3. Black**

```
const Color sf::Color::Black = Color(0, 0, 0) [static]
```

**6.12.2.4. Blue**

```
const Color sf::Color::Blue = Color(0, 0, 255) [static]
```

**6.12.2.5. g**

```
int sf::Color::g
```

**6.12.2.6. Green**

```
const Color sf::Color::Green = Color(0, 255, 0) [static]
```

**6.12.2.7. r**

```
int sf::Color::r
```

**6.12.2.8. Red**

```
const Color sf::Color::Red = Color(255, 0, 0) [static]
```

### 6.12.2.9. Transparent

```
const Color sf::Color::Transparent = Color(0, 0, 0, 0) [static]
```

### 6.12.2.10. White

```
const Color sf::Color::White = Color(255, 255, 255) [static]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

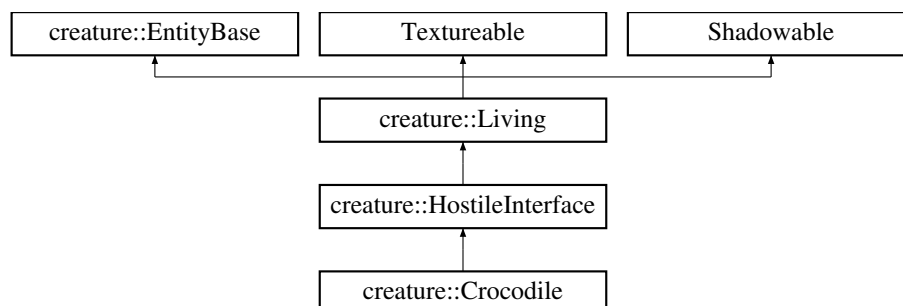
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.13. creature::Crocodile osztályreferencia

A krokodil osztály leírása.

```
#include <Crocodile.h>
```

A creature::Crocodile osztály származási diagramja:



### Publikus tagfüggvények

- [Crocodile](#) (int x, int y)  
*Idéz egy krokodilt egy pontos x és y koordinátára és beállítja az attribútumait.*
- [ENTITY\\_TYPE get\\_type](#) () const override  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- void [die](#) () override  
*Mi történjen, ha meghal az entitás.*
- void [update\\_logic](#) (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- void [draw\\_logic](#) (sf::RenderWindow &window, float deltaTime, int ofx, int ofy) override  
*Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*
- void [select\\_target](#) (World &world) override  
*A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.*
- [~Crocodile](#) ()  
*A krokodil destruktora.*

## További örökölt tagok

### 6.13.1. Részletes leírás

A krokodil osztály leírása.

A krokodil egy agresszív állat, ami mindent megeszik a robotokon kívül. Lassan mozog.

### 6.13.2. Konstruktorok és destruktorok dokumentációja

#### 6.13.2.1. Crocodile()

```
creature::Crocodile::Crocodile (
    int x,
    int y )
```

Idéz egy krokodilt egy pontos x és y koordinátára és beállítja az attribútumait.

##### Paraméterek

x	Az x koordináta.
y	Az y koordináta.

#### 6.13.2.2. ~Crocodile()

```
creature::Crocodile::~~Crocodile ( )
```

A krokodil destruktora.

### 6.13.3. Tagfüggvények dokumentációja

#### 6.13.3.1. die()

```
void creature::Crocodile::die ( ) [override], [virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.13.3.2. draw\_logic()

```
void creature::Crocodile::draw_logic (
    sf::RenderWindow & window,
    float deltaTime,
    int offx,
    int offy ) [override], [virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítja a következőket: [creature::Living](#).

### 6.13.3.3. get\_type()

```
ENTITY_TYPE creature::Crocodile::get_type ( ) const [override], [virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

#### Visszatérési érték

A belső szimbólum.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.13.3.4. select\_target()

```
void creature::Crocodile::select_target (
    World & world ) [override], [virtual]
```

A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.

#### Paraméterek

<i>world</i>	A világ, amibe a célpontot kell választani.
--------------	---

Megvalósítja a következőket: [creature::HostileInterface](#).

## 6.13.3.5. update\_logic()

```
void creature::Crocodile::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

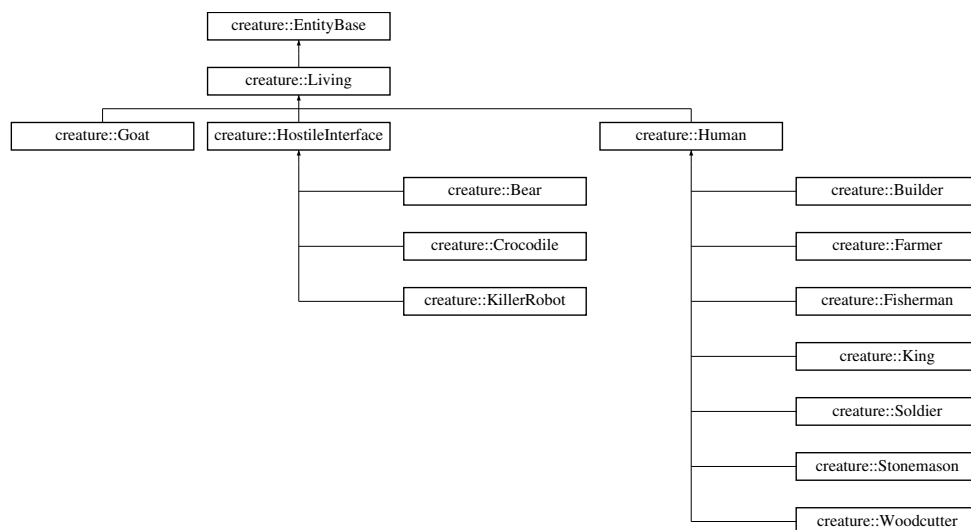
- [src/creatures/hostiles/Crocodile.h](#)
- [src/creatures/hostiles/Crocodile.cpp](#)

## 6.14. creature::EntityBase osztályreferencia

Egy alap, nem rajzolható entitás osztálya.

```
#include <EntityBase.h>
```

A creature::EntityBase osztály származási diagramja:



## Publikus tagfüggvények

- `const std::string & get_save_name () const`  
*Getter a mentés szimbólumra.*
- `void set_save_name (const std::string &s)`  
*Setter a mentés szimbólumra.*
- `ENTITY_GENDER get_gender () const`  
*Visszaadja az entitás nemét.*
- `LIVINGSTATE get_state () const`  
*Visszaadja az entitás belső állapotát.*
- `float get_death_timer () const`  
*Getter a death\_timer-re.*
- `virtual void set_state (LIVINGSTATE newstate)=0`
- `virtual ENTITY_TYPE get_type () const =0`  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- `void set_health (int amm)`  
*Beállítja az entitás életét ez bizonyos értékre.*
- `void apply_age ()`  
*Megnézi, hogy hány éves az entitás, ha már meg kell halnia akkor meghal.*
- `virtual void die ()=0`  
*Mi történjen, ha meghal az entitás.*
- `virtual ~EntityBase ()`  
*Virtuális destruktork.*
- `void set_idle_texture (std::string new_str)`  
*Frissíti az entitás "semmit nem csináló" textúráját.*
- `void set_attack_texture (std::string new_str)`  
*Frissíti az entitás "támadó" textúráját.*
- `void set_walk_texture (std::string new_str)`  
*Frissíti az entitás "sétáló" textúráját.*
- `void set_run_texture (std::string new_str)`  
*Frissíti az entitás "futó" textúráját.*
- `void set_death_texture (std::string new_str)`  
*Frissíti az entitás "elhalálozó" textúráját.*

## Publikus attribútumok

- `double posX`  
*Az entitás X pozíciója.*
- `double posy`  
*Az entitás Y pozíciója.*



## Védett attribútumok

- float `max_age`  
*Az entitás maximum életkora. Ha ezt eléri meghal.*
- `ENTITY_GENDER` `gender`  
*Az entitás neme.*
- `LIVINGSTATE` `state`  
*Az entitás belső állapota.*
- `FACING` `facing`  
*Jobbra vagy balra néz az entitás.*
- int `health`  
*Még mennyi élete maradt az entitásnak. Ha ez  $\leq 0$  akkor meghal az entitás.*
- float `hit_timer` = 0.0f  
*Ha megütik az entitást, akkor egy piros szín effektet kap, ez a változó mutatja, hogy még meddig legyen rajta ez az effekt.*
- float `inner_timer`  
*Az entitás születése óta eltelt idő.*
- float `speed`  
*Milyen gyorsan sétál az entitás (1 delta idő alatt).*
- float `run_speed_modifier`  
*Milyen gyorsan fut az entitás (1 delta idő alatt).*
- `LivingTexture` `texture_data`
- float `death_timer` = 0.1f  
*A halál animáció hátralévő idejét méri. Ha ez 0 akkor az entitás felszabadul és megsemmisül.*
- `std::string` `save_name` = "?"  
*Milyen szimbóluma legyen a mentés fájlba.*

### 6.14.1. Részletes leírás

Egy alap, nem rajzolható entitás osztálya.

Ebbe az alap entitás leírása van, kivéve a kirajzoláshoz való dolgok.

### 6.14.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.14.2.1. ~EntityBase()

```
creature::EntityBase::~~EntityBase ( ) [virtual]
```

Virtuális destruktork.

### 6.14.3. Tagfüggvények dokumentációja

#### 6.14.3.1. apply\_age()

```
void creature::EntityBase::apply_age ( )
```

Megnézi, hogy hány éves az entitás, ha már meg kell halnia akkor meghal.

#### 6.14.3.2. die()

```
virtual void creature::EntityBase::die ( ) [pure virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítják a következők: [creature::Human](#), [creature::KillerRobot](#), [creature::Crocodile](#), [creature::Bear](#) és [creature::Goat](#).

#### 6.14.3.3. get\_death\_timer()

```
float creature::EntityBase::get_death_timer ( ) const
```

Getter a death\_timer-re.

#### 6.14.3.4. get\_gender()

```
ENTITY_GENDER creature::EntityBase::get_gender ( ) const
```

Visszaadja az entitás nemét.

Visszatérési érték

Az entitás neve.

#### 6.14.3.5. get\_save\_name()

```
const std::string & creature::EntityBase::get_save_name ( ) const
```

Getter a mentés szimbólumra.

#### 6.14.3.6. get\_state()

```
LIVINGSTATE creature::EntityBase::get_state ( ) const
```

Visszaadja az entitás belső állapotát.

##### Visszatérési érték

Az entitás belső állapota.

#### 6.14.3.7. get\_type()

```
virtual ENTITY_TYPE creature::EntityBase::get_type ( ) const [pure virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

##### Visszatérési érték

A belső szimbólum.

Megvalósítják a következők: [creature::Human](#), [creature::KillerRobot](#), [creature::Crocodile](#), [creature::Bear](#) és [creature::Goat](#).

#### 6.14.3.8. set\_attack\_texture()

```
void creature::EntityBase::set_attack_texture (
    std::string new_str )
```

Frissíti az entitás "támadó" textúráját.

##### Paraméterek

<i>new_str</i>	Az új textúra elérési útvonala.
----------------	---------------------------------

#### 6.14.3.9. set\_death\_texture()

```
void creature::EntityBase::set_death_texture (
    std::string new_str )
```

Frissíti az entitás "elhalálozó" textúráját.

## Paraméterek

<i>new_str</i>	Az új textúra elérési útvonala.
----------------	---------------------------------

**6.14.3.10. set\_health()**

```
void creature::EntityBase::set_health (
    int amm )
```

Beállítja az entitás életét ez bizonyos értékre.

## Paraméterek

<i>amm</i>	Az új életpont szám.
------------	----------------------

**6.14.3.11. set\_idle\_texture()**

```
void creature::EntityBase::set_idle_texture (
    std::string new_str )
```

Frissíti az entitás "semmit nem csináló" textúráját.

## Paraméterek

<i>new_str</i>	Az új textúra elérési útvonala.
----------------	---------------------------------

**6.14.3.12. set\_run\_texture()**

```
void creature::EntityBase::set_run_texture (
    std::string new_str )
```

Frissíti az entitás "futó" textúráját.

## Paraméterek

<i>new_str</i>	Az új textúra elérési útvonala.
----------------	---------------------------------

#### 6.14.3.13. set\_save\_name()

```
void creature::EntityBase::set_save_name (
    const std::string & s )
```

Setter a mentés szimbólumra.

#### 6.14.3.14. set\_state()

```
virtual void creature::EntityBase::set_state (
    LIVINGSTATE newstate ) [pure virtual]
```

Megvalósítják a következők: [creature::Living](#).

#### 6.14.3.15. set\_walk\_texture()

```
void creature::EntityBase::set_walk_texture (
    std::string new_str )
```

Frissíti az entitás "sétáló" textúráját.

##### Paraméterek

<i>new_str</i>	Az új textúra elérési útvonala.
----------------	---------------------------------

### 6.14.4. Adattagok dokumentációja

#### 6.14.4.1. death\_timer

```
float creature::EntityBase::death_timer =0.1f [protected]
```

A halál animáció hátralévő idejét méri. Ha ez 0 akkor az entitás felszabadul és megsemmisül.

#### 6.14.4.2. facing

```
FACING creature::EntityBase::facing [protected]
```

Jobbra vagy balra néz az entitás.

#### 6.14.4.3. gender

```
ENTITY_GENDER creature::EntityBase::gender [protected]
```

Az entitás neme.

#### 6.14.4.4. health

```
int creature::EntityBase::health [protected]
```

Még mennyi élete maradt az entitásnak. Ha ez  $\leq 0$  akkor meghal az entitás.

#### 6.14.4.5. hit\_timer

```
float creature::EntityBase::hit_timer =0.0f [protected]
```

Ha megütik az entitást, akkor egy piros szín effektet kap, ez a változó mutatja, hogy még meddig legyen rajta ez az effekt.

#### 6.14.4.6. inner\_timer

```
float creature::EntityBase::inner_timer [protected]
```

Az entitás születése óta eltelt idő.

#### 6.14.4.7. max\_age

```
float creature::EntityBase::max_age [protected]
```

Az entitás maximum életkora. Ha ezt eléri meghal.

#### 6.14.4.8. posx

```
double creature::EntityBase::posx
```

Az entitás X pozíciója.

#### 6.14.4.9. posy

```
double creature::EntityBase::posy
```

Az entitás Y pozíciója.

#### 6.14.4.10. run\_speed\_modifier

```
float creature::EntityBase::run_speed_modifier [protected]
```

Milyen gyorsan fut az entitás (1 delta idő alatt).

#### 6.14.4.11. save\_name

```
std::string creature::EntityBase::save_name ="?" [protected]
```

Milyen szimbóluma legyen a mentés fájlba.

#### 6.14.4.12. speed

```
float creature::EntityBase::speed [protected]
```

Milyen gyorsan sétál az entitás (1 delta idő alatt).

#### 6.14.4.13. state

```
LIVINGSTATE creature::EntityBase::state [protected]
```

Az entitás belső állapota.

#### 6.14.4.14. texture\_data

```
LivingTexture creature::EntityBase::texture_data [protected]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/creatures/[EntityBase.h](#)
- src/creatures/[EntityBase.cpp](#)

## 6.15. EntityPlacer osztályreferencia

Az entitások a kattintással való lerakása.

```
#include <EntityPlacer.h>
```

### Publikus tagfüggvények

- [EntityPlacer](#) ()  
*Alap konstruktor, ami beállítja, hogy eleinte nem szabad lerakni semmit.*
- bool [try\\_place\\_entity](#) ([sf::Vector2i](#) &epos, [World](#) &world)  
*Megpróbál lerakni egy entitást a kurzor helyére, ha megteheti.*
- void [toggle\\_placing](#) ()  
*Ki-be kapcsolja a működést.*
- void [select\\_entity](#) (int new\_id)  
*Beállítja, hogy milyen entitást rakjon le index alapján.*
- bool [setup\\_factory](#) ()  
*Beállítja az alap idézési parancsokat.*
- void [reset\\_mouse](#) ()

### Publikus attribútumok

- bool [spacePreviouslyPressed](#)

#### 6.15.1. Részletes leírás

Az entitások a kattintással való lerakása.

#### 6.15.2. Konstruktorok és destruktorok dokumentációja

##### 6.15.2.1. EntityPlacer()

```
EntityPlacer::EntityPlacer ( )
```

Alap konstruktor, ami beállítja, hogy eleinte nem szabad lerakni semmit.

#### 6.15.3. Tagfüggvények dokumentációja



**6.15.3.1. reset\_mouse()**

```
void EntityPlacer::reset_mouse ( )
```

**6.15.3.2. select\_entity()**

```
void EntityPlacer::select_entity (
    int new_id )
```

Beállítja, hogy milyen entitást rakjon le index alapján.

**6.15.3.3. setup\_factory()**

```
bool EntityPlacer::setup_factory ( )
```

Beállítja az alap idézési parancsokat.

**6.15.3.4. toggle\_placing()**

```
void EntityPlacer::toggle_placing ( )
```

Ki-be kapcsolja a működést.

**6.15.3.5. try\_place\_entity()**

```
bool EntityPlacer::try_place_entity (
    sf::Vector2i & epos,
    World & world )
```

Megpróbál lerakni egy entitást a kurzor helyére, ha megteheti.

**6.15.4. Adattagok dokumentációja**

#### 6.15.4.1. spacePreviouslyPressed

```
bool EntityPlacer::spacePreviouslyPressed
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/EntityPlacer.h](#)
- [src/EntityPlacer.cpp](#)

### 6.16. sf::Event osztályreferencia

```
#include <fake_sfml.h>
```

#### Publikus típusok

- enum [EType](#) : char { [Closed](#) , [NoEvent](#) , [Invalid](#) }

#### Publikus tagfüggvények

- [Event](#) ()

#### Publikus attribútumok

- [EType](#) type

#### 6.16.1. Enumeráció-tagok dokumentációja

##### 6.16.1.1. EType

```
enum sf::Event::EType : char
```

Enumeráció-értékek

Closed	
NoEvent	
Invalid	

#### 6.16.2. Konstruktorkok és destruktorkok dokumentációja

### 6.16.2.1. Event()

```
sf::Event::Event ( )
```

## 6.16.3. Adattagok dokumentációja

### 6.16.3.1. type

```
EType sf::Event::type
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

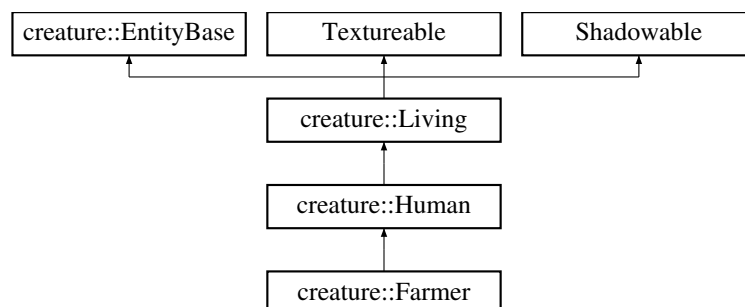
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.17. creature::Farmer osztályreferencia

A farmer szakmájú ember osztály leírása.

```
#include <Farmer.h>
```

A creature::Farmer osztály származási diagramja:



## Publikus tagfüggvények

- [Farmer](#) (int x, int y, [ENTITY\\_GENDER](#) gender\_modifier)  
*Inicializál egy farmert egy pontos x és y koordinátára és beállítja az attribútumait.*
- void [update\\_logic](#) ([World](#) &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- [~Farmer](#) ()  
*A farmer destruktora.*

## További örökölt tagok

### 6.17.1. Részletes leírás

A farmer szakmájú ember osztály leírása.

Ez a szakmájú ember bokrokat keres és kitermeli őket ezzel ételt szerez.

### 6.17.2. Konstruktorok és destruktorok dokumentációja

#### 6.17.2.1. Farmer()

```
creature::Farmer::Farmer (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy farmert egy pontos x és y koordinátára és beállítja az attribútumait.

##### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	A farmer neme.

#### 6.17.2.2. ~Farmer()

```
creature::Farmer::~~Farmer ( )
```

A farmer destruktora.

### 6.17.3. Tagfüggvények dokumentációja

#### 6.17.3.1. update\_logic()

```
void creature::Farmer::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

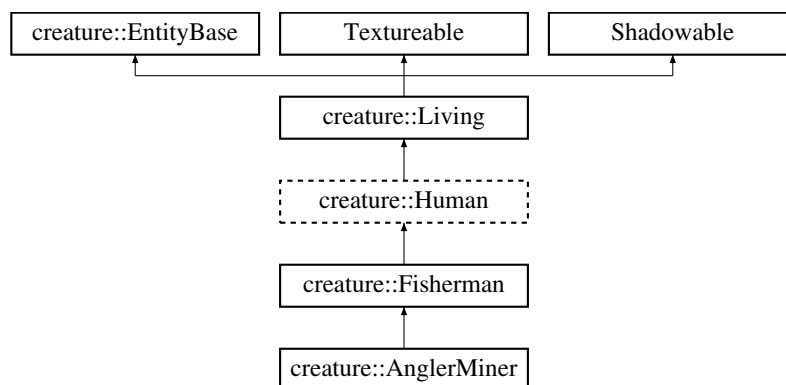
- src/creatures/humans/[Farmer.h](#)
- src/creatures/humans/[Farmer.cpp](#)

## 6.18. creature::Fisherman osztályreferencia

A halász szakmájú ember osztály leírása.

```
#include <Fisherman.h>
```

A creature::Fisherman osztály származási diagramja:



### Publikus tagfüggvények

- [Fisherman](#) (int x, int y, [ENTITY\\_GENDER](#) gender\_modifier)  
*Inicializál egy halászt egy pontos x és y koordinátára és beállítja az attribútumait.*
- void [update\\_logic](#) ([World](#) &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- [~Fisherman](#) ()  
*A halász destruktora.*

### Védett tagfüggvények

- void [try\\_fishing](#) ([World](#) &world)  
*Megpróbál tavat keresni, ahol halászhatsz.*

## Védett attribútumok

- bool `fishing`

*Halászni akar-e az ember jelenleg?*

## További örökölt tagok

### 6.18.1. Részletes leírás

A halász szakmájú ember osztály leírása.

Ez a szakmájú ember víz terepkockát keres és ott halászva ételt szerez.

### 6.18.2. Konstruktork és destruktorok dokumentációja

#### 6.18.2.1. Fisherman()

```
creature::Fisherman::Fisherman (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy halászt egy pontos x és y koordinátára és beállítja az attribútumait.

#### Paraméterek

<code>x</code>	Az x koordináta.
<code>y</code>	Az y koordináta.
<code>gender_modifier</code>	A halász neme.

#### 6.18.2.2. ~Fisherman()

```
creature::Fisherman::~~Fisherman ( )
```

A halász destruktor.

### 6.18.3. Tagfüggvények dokumentációja

### 6.18.3.1. try\_fishing()

```
void creature::Fisherman::try_fishing (
    World & world ) [protected]
```

Megpróbál tavat keresni, ahol halászhat.

#### Paraméterek

<i>world</i>	A világ, amibe tavat kell keresni.
--------------	------------------------------------

### 6.18.3.2. update\_logic()

```
void creature::Fisherman::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

## 6.18.4. Adattagok dokumentációja

### 6.18.4.1. fishing

```
bool creature::Fisherman::fishing [protected]
```

Halászni akar-e az ember jelenleg?

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/creatures/humans/Fisherman.h](#)
- [src/creatures/humans/Fisherman.cpp](#)

## 6.19. sf::FloatRect osztályreferencia

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- [FloatRect](#) ()
- [FloatRect](#) (float l, float t, float w, float h)
- bool [contains](#) (float x, float y) const

## Publikus attribútumok

- float [left](#)
- float [top](#)
- float [width](#)
- float [height](#)

## 6.19.1. Konstruktorkok és destruktorkok dokumentációja

### 6.19.1.1. FloatRect() [1/2]

```
sf::FloatRect::FloatRect ( )
```

### 6.19.1.2. FloatRect() [2/2]

```
sf::FloatRect::FloatRect (
    float l,
    float t,
    float w,
    float h )
```

## 6.19.2. Tagfüggvények dokumentációja

### 6.19.2.1. contains()

```
bool sf::FloatRect::contains (
    float x,
    float y ) const
```

## 6.19.3. Adattagok dokumentációja



#### 6.19.3.1. height

```
float sf::FloatRect::height
```

#### 6.19.3.2. left

```
float sf::FloatRect::left
```

#### 6.19.3.3. top

```
float sf::FloatRect::top
```

#### 6.19.3.4. width

```
float sf::FloatRect::width
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- `src/fake_sfml/fake_sfml.h`
- `src/fake_sfml/fake_sfml.cpp`

## 6.20. GameConfig osztályreferencia

A világ szimulációjának leírása.

```
#include <GameConfig.h>
```

## Publikus tagfüggvények

- `GameConfig` (const `GameConfig` &)=delete  
*Nem szükséges a singleton pattern miatt.*
- `GameConfig` & `operator=` (const `GameConfig` &)=delete  
*Nem szükséges a singleton pattern miatt.*
- bool `read_from_config_file` (const std::string &filepath)  
*Beolvassa a filepath elérési útvonalról a konfigurációt.*
- int `get_config_level` () const  
*Visszaadja, hogy mennyire bőbeszédű legyen a hibakezelés / figyelmeztetések.*
- int `get_target_fps` () const  
*Visszaadja az elérni kívánt FPS értékét.*
- int `get_screen_width` () const  
*Visszaadja az ablak szélességét.*
- int `get_screen_height` () const  
*Visszaadja az ablak magasságát.*
- void `set_config_level` (int n\_flag)  
*Beállítható, hogy mennyire bőbeszédű legyen a hibakezelés / figyelmeztetések.*
- int `get_world_size` () const  
*Visszaadja a világ konfigurált méretét.*
- void `set_world_size` (int newsize)  
*Beállítja a világ konfigurált méretét.*
- int `get_max_spawn_tries` () const
- int `get_resource_scarcity` () const
- int `get_hostiles_count` () const
- bool `is_noise` () const
- bool `is_chromatic_aberration` () const
- `Language` `get_lang` () const
- `Language` `get_sfml_lang` () const

## Statikus publikus tagfüggvények

- static `GameConfig` & `get_instance` ()  
*Visszaad egy referenciát erre az osztály-ra.*

## Publikus attribútumok

- float `day_length` =400.0  
*A napok hossza másodpercben.*

### 6.20.1. Részletes leírás

A világ szimulációjának leírása.

Tárolja azokat az értékeket, amiktől függ az, hogy mi mikor és hogyan történik a világba.

### 6.20.2. Konstruktorkok és destruktorkok dokumentációja

### 6.20.2.1. GameConfig()

```
GameConfig::GameConfig (
    const GameConfig & ) [delete]
```

Nem szükséges a singleton pattern miatt.

## 6.20.3. Tagfüggvények dokumentációja

### 6.20.3.1. get\_config\_level()

```
int GameConfig::get_config_level ( ) const
```

Visszaadja, hogy mennyire bőbeszédű legyen a hibakezelés / figyelmeztetések.

### 6.20.3.2. get\_hostiles\_count()

```
int GameConfig::get_hostiles_count ( ) const
```

### 6.20.3.3. get\_instance()

```
GameConfig & GameConfig::get_instance ( ) [static]
```

Visszaad egy referenciát erre az osztály-ra.

Visszatérési érték

A singleton-hoz egy referencia.

### 6.20.3.4. get\_lang()

```
Language GameConfig::get_lang ( ) const
```

### 6.20.3.5. get\_max\_spawn\_tries()

```
int GameConfig::get_max_spawn_tries ( ) const
```

**6.20.3.6. get\_resource\_scarcity()**

```
int GameConfig::get_resource_scarcity ( ) const
```

**6.20.3.7. get\_screen\_height()**

```
int GameConfig::get_screen_height ( ) const
```

Visszaadja az ablak magasságát.

**6.20.3.8. get\_screen\_width()**

```
int GameConfig::get_screen_width ( ) const
```

Visszaadja az ablak szélességét.

**6.20.3.9. get\_sfml\_lang()**

```
Language GameConfig::get_sfml_lang ( ) const
```

**6.20.3.10. get\_target\_fps()**

```
int GameConfig::get_target_fps ( ) const
```

Visszaadja az elérni kívánt FPS értékét.

**6.20.3.11. get\_world\_size()**

```
int GameConfig::get_world_size ( ) const
```

Visszaadja a világ konfigurált méretét.

**6.20.3.12. is\_chromatic\_aberration()**

```
bool GameConfig::is_chromatic_aberration ( ) const
```

**6.20.3.13. is\_noise()**

```
bool GameConfig::is_noise ( ) const
```

**6.20.3.14. operator=()**

```
GameConfig& GameConfig::operator= (
    const GameConfig & ) [delete]
```

Nem szükséges a singleton pattern miatt.

**6.20.3.15. read\_from\_config\_file()**

```
bool GameConfig::read_from_config_file (
    const std::string & filepath )
```

Beolvassa a filepath elérési útvonalról a konfigurációt.

**6.20.3.16. set\_config\_level()**

```
void GameConfig::set_config_level (
    int n_flag )
```

Beállítható, hogy mennyire bőbeszédű legyen a hibakezelés / figyelmeztetések.

**6.20.3.17. set\_world\_size()**

```
void GameConfig::set_world_size (
    int newsize )
```

Beállítja a világ konfigurált méretét.

**6.20.4. Adattagok dokumentációja**

#### 6.20.4.1. day\_length

```
float GameConfig::day_length =400.0
```

A napok hossza másodpercben.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/GameConfig.h
- src/GameConfig.cpp

### 6.21. GameManager osztályreferencia

A világ szimulálásáért és a kirazolás irányításáért felelős osztály.

```
#include <GameManager.h>
```

#### Publikus tagfüggvények

- [GameManager](#) ()  
*A konstruktorba létrejön az ablak és az alap változók beállításra kerülnek.*
- void [run](#) ()  
*Elindítja a szimulációt és innentől kirajolja a világot, gombokat.*
- void [game\\_loop](#) ()  
*A szimuláció loopolását indítja el.*
- void [setup\\_buttons](#) ()  
*A gombokat létrehozza, textúrájukat, viselkedésüket betölti.*
- void [update\\_buttons](#) ()  
*Frissíti a gombokat, ha 1-re rákattintottak.*
- void [draw\\_buttons](#) ()  
*Kirajolja a gombokat.*
- bool [is\\_valid](#) () const  
*Megadja, hogy sikeres lett-e a szimulációs elemek inicializálása.*
- float [get\\_elapsed\\_time](#) () const  
*Megadja az eltelt időt, ami eltelt a szimulációba.*
- void [simulate\\_tick](#) (float e\_time)  
*Szimulál T idő egységnyi időt.*
- bool [handle\\_unit\\_placement](#) ()  
*Igazat ad vissza, ha idéztek le entitást, ha nem akkor hamis.*
- [~GameManager](#) ()  
*Felszabadítja a világot, gombokat, hangot, textúrákat, render ablakot. Mindent, ami a program tartalmaz.*

#### 6.21.1. Részletes leírás

A világ szimulálásáért és a kirazolás irányításáért felelős osztály.

Tárolja a világot, a render ablakot, a kamera adatait, a gombokat és a zene lejátszót. Végül mindent ez az osztály szabadít fel.

## 6.21.2. Konstruktorkok és destruktorkok dokumentációja

### 6.21.2.1. GameManager()

```
GameManager::GameManager ( )
```

A konstruktorba létrejön az ablak és az alap változók beállításra kerülnek.

### 6.21.2.2. ~GameManager()

```
GameManager::~~GameManager ( )
```

Felszabadítja a világot, gombokat, hangot, textúrákat, render ablakot. Mindent, ami a program tartalmaz.

## 6.21.3. Tagfüggvények dokumentációja

### 6.21.3.1. draw\_buttons()

```
void GameManager::draw_buttons ( )
```

Kirajzolja a gombokat.

### 6.21.3.2. game\_loop()

```
void GameManager::game_loop ( )
```

A szimuláció loopolását indítja el.

### 6.21.3.3. get\_elapsed\_time()

```
float GameManager::get_elapsed_time ( ) const
```

Megadja az eltelt időt, ami eltelt a szimulációba.

#### 6.21.3.4. `handle_unit_placement()`

```
bool GameManager::handle_unit_placement ( )
```

Igazat ad vissza, ha idéztek le entitást, ha nem akkor hamis.

#### 6.21.3.5. `is_valid()`

```
bool GameManager::is_valid ( ) const
```

Megadja, hogy sikeres lett-e a szimulációs elemek inicializálása.

#### 6.21.3.6. `run()`

```
void GameManager::run ( )
```

Elindítja a szimulációt és innentől kirajolja a világot, gombokat.

#### 6.21.3.7. `setup_buttons()`

```
void GameManager::setup_buttons ( )
```

A gombokat létrehozza, textúrájukat, viselkedésüket betölti.

#### 6.21.3.8. `simulate_tick()`

```
void GameManager::simulate_tick (
    float e_time )
```

Szimulál T idő egységnyi időt.

#### 6.21.3.9. `update_buttons()`

```
void GameManager::update_buttons ( )
```

Frissíti a gombokat, ha 1-re rákattintottak.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/GameManager.h](#)
- [src/GameManager.cpp](#)

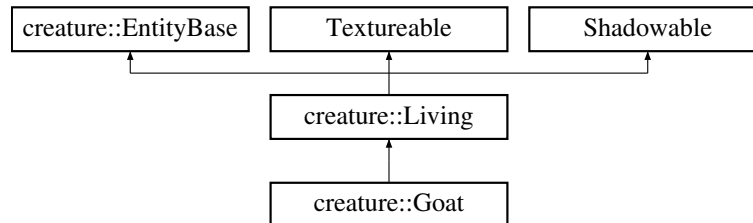


## 6.22. creature::Goat osztályreferencia

A kecske osztály leírása.

```
#include <Goat.h>
```

A creature::Goat osztály származási diagramja:



### Publikus tagfüggvények

- **Goat** (int x, int y)  
*Idéz egy kecskét egy pontos x és y koordinátára és beállítja az attribútumait.*
- **ENTITY\_TYPE get\_type** () const override  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- void **die** () override  
*Mi történjen, ha meghal az entitás.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- void **draw\_logic** (sf::RenderWindow &window, float deltaTime, int offx, int offy) override  
*Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*
- virtual **~Goat** ()  
*Virtuális destruktork.*

### További örökölt tagok

#### 6.22.1. Részletes leírás

A kecske osztály leírása.

A kecske egy passzív, nem támadó állat, amit ha az emberek megölnek, ételt ad.

#### 6.22.2. Konstruktorok és destruktork dokumentációja

##### 6.22.2.1. Goat()

```
creature::Goat::Goat (
    int x,
    int y )
```

Idéz egy kecskét egy pontos x és y koordinátára és beállítja az attribútumait.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

**6.22.2.2. ~Goat()**

```
creature::Goat::~~Goat ( ) [virtual]
```

Virtuális destruktork.

**6.22.3. Tagfüggvények dokumentációja****6.22.3.1. die()**

```
void creature::Goat::die ( ) [override], [virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítja a következőket: [creature::EntityBase](#).

**6.22.3.2. draw\_logic()**

```
void creature::Goat::draw_logic (
    sf::RenderWindow & window,
    float deltaTime,
    int offx,
    int offy ) [override], [virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

## Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítja a következőket: [creature::Living](#).

### 6.22.3.3. get\_type()

```
ENTITY_TYPE creature::Goat::get_type ( ) const [override], [virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

#### Visszatérési érték

A belső szimbólum.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.22.3.4. update\_logic()

```
void creature::Goat::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

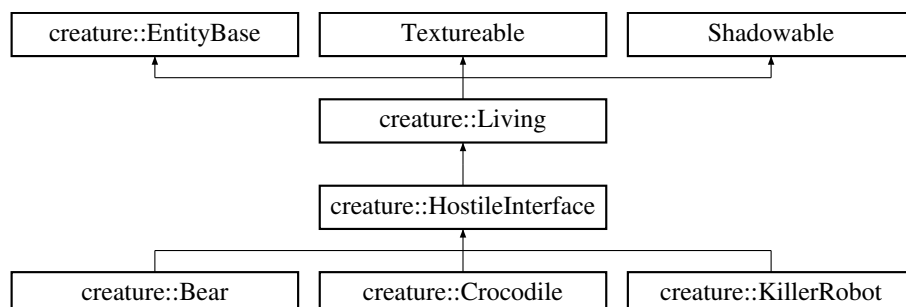
- [src/creatures/Goat.h](#)
- [src/creatures/Goat.cpp](#)

## 6.23. creature::HostileInterface osztályreferencia

A vadállat entitások interface leírása.

```
#include <HostileInterface.h>
```

A creature::HostileInterface osztály származási diagramja:



## Publikus tagfüggvények

- void `set_hostile_config` (int newdamage, float newattackspeed)  
*Beállítja a vadállat támadási sebességét és sebzését.*
- virtual void `select_target` (World &world)=0  
*A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.*
- virtual `~HostileInterface` ()=default  
*Virtuális destruktork.*
- void `retarget` (Living \*new\_target) override  
*Felidegesíti az entitást a kapott entításra.*
- Living \* `check_aggroed` () const override  
*Visszaadja, hogy kire "ideges az entitás". Ez lehet az, hogy ki ütötte meg vagy hogy kit akar megenni.*

## Védett tagfüggvények

- void `try_attack` ()  
*Megnézi, hogy milyen közel van a célpontja, ha elég közel van, akkor támad.*
- void `hostile_run` (float deltaTime)  
*Egységes futás logika. Addíg fut a célpont felé míg az vagy meghal vagy elég közel lesz.*
- void `hostile_walk` (float deltaTime)  
*Egységes séta logika. Addíg sétál a célpont felé míg az vagy meghal vagy elég közel lesz. Ezt használja a krokodil, ha ebbe a fázisba meghal a célpont, akkor futás módba vált és egyből keres egy új célpontot.*

## Védett attribútumok

- sf::Vector2f `goal`  
*Az célpont entitás pozíciója.*
- int `damage`  
*A vadállat sebzése.*
- float `attack_speed`  
*A vadállat támadási sebessége.*
- Living \* `target`  
*A vadállat célpontja.*

## További örökölt tagok

### 6.23.1. Részletes leírás

A vadállat entitások interface leírása.

Ebbe minden deklarálva van, ami ahhoz kell, hogy egy entitás agresszív legyen. Van célpontjuk, egységes támadási módszereik és sebzésük.

### 6.23.2. Konstruktorkok és destruktorkok dokumentációja

### 6.23.2.1. ~HostileInterface()

```
virtual creature::HostileInterface::~~HostileInterface ( ) [virtual], [default]
```

Virtuális destruktork.

## 6.23.3. Tagfüggvények dokumentációja

### 6.23.3.1. check\_aggroed()

```
Living * creature::HostileInterface::check_aggroed ( ) const [override], [virtual]
```

Visszaadja, hogy mire "ideges az entitás". Ez lehet az, hogy ki ütötte meg vagy hogy kit akar megenni.

Visszatérési érték

Az entitás, akire ideges. Nullpointer, ha nincs ilyen entitás.

Újraimplementált ősök: [creature::Living](#).

### 6.23.3.2. hostile\_run()

```
void creature::HostileInterface::hostile_run (
    float deltaTime ) [protected]
```

Egységes futás logika. Addíg fut a célpont felé míg az vagy meghal vagy elég közel lesz.

### 6.23.3.3. hostile\_walk()

```
void creature::HostileInterface::hostile_walk (
    float deltaTime ) [protected]
```

Egységes séta logika. Addíg sétál a célpont felé míg az vagy meghal vagy elég közel lesz. Ezt használja a krokodil, ha ebbe a fázisba meghal a célpont, akkor futás módba vált és egyből keres egy új célpontot.

### 6.23.3.4. retarget()

```
void creature::HostileInterface::retarget (
    Living * new_target ) [override], [virtual]
```

Felidegesíti az entitást a kapott entításra.

## Paraméterek

<i>new_target</i>	Az entitás, akire dühösnek kell lennie.
-------------------	---

Újrimplementált ősök: [creature::Living](#).

**6.23.3.5. select\_target()**

```
virtual void creature::HostileInterface::select_target (
    World & world ) [pure virtual]
```

A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.

## Paraméterek

<i>world</i>	A világ, amibe a célpontot kell választani.
--------------	---

Megvalósítják a következők: [creature::KillerRobot](#), [creature::Crocodile](#) és [creature::Bear](#).

**6.23.3.6. set\_hostile\_config()**

```
void creature::HostileInterface::set_hostile_config (
    int newdamage,
    float newattackspeed )
```

Beállítja a vadállat támadási sebességét és sebzését.

## Paraméterek

<i>newdamage</i>	Az új beállított sebzés.
<i>newattackspeed</i>	Az új beállított sebzési sebesség.

**6.23.3.7. try\_attack()**

```
void creature::HostileInterface::try_attack ( ) [protected]
```

Megnézi, hogy milyen közel van a célpontja, ha elég közel van, akkor támad.

**6.23.4. Adattagok dokumentációja**

#### 6.23.4.1. attack\_speed

```
float creature::HostileInterface::attack_speed [protected]
```

A vadállat támadási sebessége.

#### 6.23.4.2. damage

```
int creature::HostileInterface::damage [protected]
```

A vadállat sebzése.

#### 6.23.4.3. goal

```
sf::Vector2f creature::HostileInterface::goal [protected]
```

Az célpont entitás pozíciója.

#### 6.23.4.4. target

```
Living* creature::HostileInterface::target [protected]
```

A vadállat célpontja.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

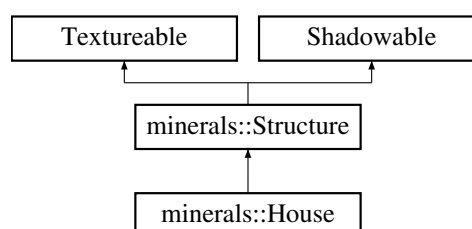
- [src/creatures/HostileInterface.h](#)
- [src/creatures/HostileInterface.cpp](#)

## 6.24. minerals::House osztályreferencia

A ház osztály leírása. Szinttől függően idéz embereket.

```
#include <House.h>
```

A minerals::House osztály származási diagramja:



## Publikus tagfüggvények

- `House` (int x, int y)  
*Konstruktor ami lerakja a házat egy (x,y) pontra.*
- `MINERAL_TYPE get_type` () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void `update_logic` (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- bool `try_upgrade` (const `HumanResources` &hr)
- int `get_level` () const
- void `set_level` (int new\_level)
- void `upgrade_house` ()

## További örökölt tagok

### 6.24.1. Részletes leírás

A ház osztály leírása. Szinttől függően idéz embereket.

### 6.24.2. Konstruktorok és destruktorok dokumentációja

#### 6.24.2.1. `House()`

```
minerals::House::House (  
    int x,  
    int y )
```

Konstruktor ami lerakja a házat egy (x,y) pontra.

### 6.24.3. Tagfüggvények dokumentációja

#### 6.24.3.1. `get_level()`

```
int minerals::House::get_level ( ) const
```



#### 6.24.3.2. get\_type()

```
MINERAL_TYPE minerals::House::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.24.3.3. set\_level()

```
void minerals::House::set_level (
    int new_level )
```

#### 6.24.3.4. try\_upgrade()

```
bool minerals::House::try_upgrade (
    const HumanResources & hr )
```

#### 6.24.3.5. update\_logic()

```
void minerals::House::update_logic (
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.

##### Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.24.3.6. upgrade\_house()

```
void minerals::House::upgrade_house ( )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

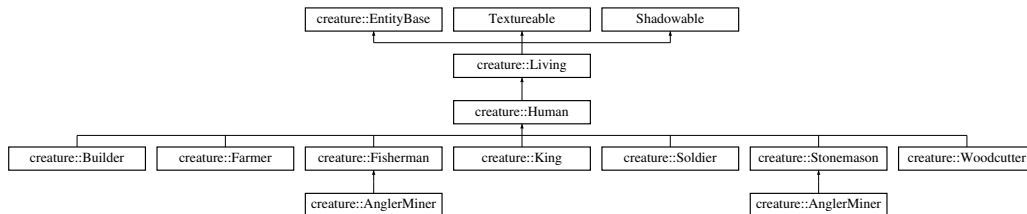
- [src/world\\_object/House.h](#)
- [src/world\\_object/House.cpp](#)

## 6.25. creature::Human osztályreferencia

Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik.

```
#include <Human.h>
```

A creature::Human osztály származási diagramja:



### Publikus tagfüggvények

- **Human** (int x, int y)  
*Az alap konstruktor ami leidézi az embert egy x és y koordinátára.*
- **Human** (int x, int y, ENTITY\_GENDER const\_gender)  
*Az alap konstruktor ami leidézi az embert egy x és y koordinátára egy megadott nemmel.*
- **ENTITY\_TYPE get\_type** () const override  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- void **die** () override  
*Mi történjen, ha meghal az entitás.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- void **draw\_logic** (sf::RenderWindow &window, float deltaTime, int ofx, int ofy) override  
*Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*
- virtual **~Human** ()  
*Virtuális destruktor, felszabadítja a szakma ikon pointert is.*
- void **initialize** (int x, int y)  
*Beállítja az ember tulajdonságait: életpontok, max életkor, nem.*
- void **select\_texture** (int x, int y, int gender\_selector)  
*Beállít egy textúrát ami nagyon különböző lehet emberenélként és egyből beállítja, hogy az embert a saját (x,y) koordinátára rajzolja ki.*
- std::string **get\_profession\_string** ()  
*Lekérhető az ember szakmájának szöveggé alakított szimbóluma. Ez fontos a fájlba tároláshoz.*

### Publikus attribútumok

- bool **needs\_to\_be\_royal**  
*Kell-e királyá koronázni?*
- bool **needs\_promotion**  
*Kell-e neki egy új szakma? Csak akkor igaz, ha már van város.*

## Védett tagfüggvények

- virtual bool [humanoid\\_run](#) (float deltaTime)  
*Az emberszabású futás függvény.*
- virtual bool [humanoid\\_walk](#) (float deltaTime)  
*Az emberszabású mozgás függvény.*

## Védett attribútumok

- [Profession](#) \* [profession](#) =nullptr  
*A szakma ikon pointere.*
- [sf::Vector2f](#) [goal](#)  
*A cselekvésének a célpontja.*

## További örökölt tagok

### 6.25.1. Részletes leírás

Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik.

Tárolja az ember szakma címerét, célkoordinátáját is.

### 6.25.2. Konstruktorkok és destruktorok dokumentációja

#### 6.25.2.1. Human() [1/2]

```
creature::Human::Human (
    int x,
    int y )
```

Az alap konstruktor ami leidezi az embert egy x és y koordinátára.

#### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

#### 6.25.2.2. Human() [2/2]

```
creature::Human::Human (
    int x,
```

```
int y,
ENTITY_GENDER const_gender )
```

Az alap konstruktor ami leidezi az embert egy x és y koordinátára egy megadott nemmel.

#### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>const_gender</i>	Az ember neme.

#### 6.25.2.3. ~Human()

```
creature::Human::~~Human ( ) [virtual]
```

Virtuális destruktork, felszabadítja a szakma ikon pointert is.

### 6.25.3. Tagfüggvények dokumentációja

#### 6.25.3.1. die()

```
void creature::Human::die ( ) [override], [virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítja a következőket: [creature::EntityBase](#).

#### 6.25.3.2. draw\_logic()

```
void creature::Human::draw_logic (
    sf::RenderWindow & window,
    float deltaTime,
    int offx,
    int offy ) [override], [virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítja a következőket: [creature::Living](#).

#### 6.25.3.3. get\_profession\_string()

```
std::string creature::Human::get_profession_string ( )
```

Lekérhető az ember szakmájának szöveggé alakított szimbóluma. Ez fontos a fájlba tároláshoz.

##### Visszatérési érték

Az ember szakmájának szimbóluma.

#### 6.25.3.4. get\_type()

```
ENTITY_TYPE creature::Human::get_type ( ) const [override], [virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

##### Visszatérési érték

A belső szimbólum.

Megvalósítja a következőket: [creature::EntityBase](#).

#### 6.25.3.5. humanoid\_run()

```
bool creature::Human::humanoid_run (
    float deltaTime ) [protected], [virtual]
```

Az emberszabású futás függvény.

#### 6.25.3.6. humanoid\_walk()

```
bool creature::Human::humanoid_walk (
    float deltaTime ) [protected], [virtual]
```

Az emberszabású mozgás függvény.

#### 6.25.3.7. initialize()

```
void creature::Human::initialize (
    int x,
    int y )
```

Beállítja az ember tulajdonságait: életpontok, max életkor, nem.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

**6.25.3.8. select\_texture()**

```
void creature::Human::select_texture (
    int x,
    int y,
    int gender_selector )
```

Beállít egy textúrát ami nagyon különböző lehet emberenéknt és egyből beállítja, hogy az embert a saját (x,y) koordinátára rajzolják.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_selector</i>	Egy véletlen szám. Ettől függ a textúra variáció.

**6.25.3.9. update\_logic()**

```
void creature::Human::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

< Még mindig nincs?

Megvalósítja a következőket: [creature::Living](#).

Újraimplementáló leszármazottak: [creature::Woodcutter](#), [creature::Stonemason](#), [creature::Soldier](#) és [creature::King](#).

**6.25.4. Adattagok dokumentációja**

#### 6.25.4.1. goal

```
sf::Vector2f creature::Human::goal [protected]
```

A cselekvésének a célpontja.

#### 6.25.4.2. needs\_promotion

```
bool creature::Human::needs_promotion
```

Kell-e neki egy új szakma? Csak akkor igaz, ha már van város.

#### 6.25.4.3. needs\_to\_be\_royal

```
bool creature::Human::needs_to_be_royal
```

Kell-e királyá koronázni?

#### 6.25.4.4. profession

```
Profession* creature::Human::profession =nullptr [protected]
```

A szakma ikon pointerre.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/creatures/humans/[Human.h](#)
- src/creatures/humans/[Human.cpp](#)

## 6.26. HumanResources osztályreferencia

Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva.

```
#include <HumanResources.h>
```

## Publikus tagfüggvények

- void `add_resources` (const std::string &what, int amount)  
*Az emberek által gyűjtött erőforrásokhoz hozzáad egy típusból valamennyit.*
- void `remove_resources` (const std::string &what, int amount)  
*Az emberek által gyűjtött erőforrásokból kised egy típusból valamennyit.*
- bool `is_there_enough_resource` (const std::string &from\_what, int needed\_amount) const  
*Megnézi, hogy az emberek már szedtek-e elég erőforrást valamiből.*
- void `set_resources` (const std::string &what, int amount)  
*Beállítja egy erőforrás számát fix értékre.*
- int `get_count_from` (const std::string &what) const  
*Visszaadja, hogy mennyi erőforrás van egy bizonyos típusból.*

### 6.26.1. Részletes leírás

Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva.

### 6.26.2. Tagfüggvények dokumentációja

#### 6.26.2.1. `add_resources()`

```
void HumanResources::add_resources (
    const std::string & what,
    int amount )
```

Az emberek által gyűjtött erőforrásokhoz hozzáad egy típusból valamennyit.

##### Paraméterek

<i>what</i>	Mit adjon hozzá.
<i>amount</i>	Mennyit adjon hozzá.

#### 6.26.2.2. `get_count_from()`

```
int HumanResources::get_count_from (
    const std::string & what ) const
```

Visszaadja, hogy mennyi erőforrás van egy bizonyos típusból.



### 6.26.2.3. is\_there\_enough\_resource()

```
bool HumanResources::is_there_enough_resource (
    const std::string & from_what,
    int needed_amount ) const
```

Megnézi, hogy az emberek már szedtek-e elég erőforrást valamiből.

#### Paraméterek

<i>from_what</i>	Miből kell.
<i>needed_amount</i>	Mennyi kell, hogy legyen.

#### Visszatérési érték

Ha van elég, akkor igaz, ha nincs akkor hamis.

### 6.26.2.4. remove\_resources()

```
void HumanResources::remove_resources (
    const std::string & what,
    int amount )
```

Az emberek által gyűjtött erőforrásokból kised egy típusból valamennyit.

#### Paraméterek

<i>what</i>	Mit vesz el.
-------------	--------------

#### Visszatérési érték

Mennyit vegyen el.

### 6.26.2.5. set\_resources()

```
void HumanResources::set_resources (
    const std::string & what,
    int amount )
```

Beállítja egy erőforrás számát fix értékre.

#### Paraméterek

<i>what</i>	Miből kell.
<i>amount</i>	Mennyi kell, hogy legyen.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

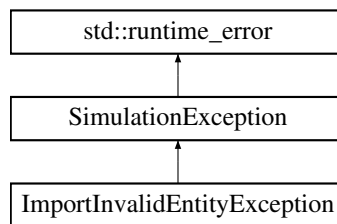
- [src/HumanResources.h](#)
- [src/HumanResources.cpp](#)

## 6.27. ImportInvalidEntityException osztályreferencia

Akkor kell dobni, ha egy entitás hibásan lett beolvasva.

```
#include <FileExceptions.h>
```

Az ImportInvalidEntityException osztály származási diagramja:



### Publikus tagfüggvények

- [ImportInvalidEntityException](#) (const std::string &msg)

#### 6.27.1. Részletes leírás

Akkor kell dobni, ha egy entitás hibásan lett beolvasva.

#### 6.27.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.27.2.1. ImportInvalidEntityException()

```
ImportInvalidEntityException::ImportInvalidEntityException (  
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

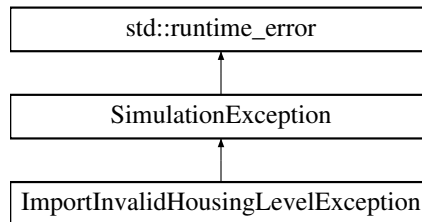
- [src/exceptions/FileExceptions.h](#)

## 6.28. ImportInvalidHousingLevelException osztályreferencia

Akkor kell dobni, ha egy ház hibásan lett beolvasva.

```
#include <FileExceptions.h>
```

Az ImportInvalidHousingLevelException osztály származási diagramja:



### Publikus tagfüggvények

- [ImportInvalidHousingLevelException](#) (const std::string &msg)

#### 6.28.1. Részletes leírás

Akkor kell dobni, ha egy ház hibásan lett beolvasva.

#### 6.28.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.28.2.1. ImportInvalidHousingLevelException()

```
ImportInvalidHousingLevelException::ImportInvalidHousingLevelException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

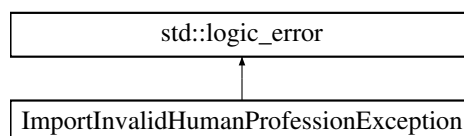
- src/exceptions/[FileExceptions.h](#)

## 6.29. ImportInvalidHumanProfessionException osztályreferencia

Akkor kell dobni, ha egy szakma hibásan lett beolvasva.

```
#include <FileExceptions.h>
```

Az ImportInvalidHumanProfessionException osztály származási diagramja:



## Publikus tagfüggvények

- [ImportInvalidHumanProfessionException](#) (const std::string &msg)

### 6.29.1. Részletes leírás

Akkor kell dobni, ha egy szakma hibásan lett beolvasva.

### 6.29.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.29.2.1. ImportInvalidHumanProfessionException()

```
ImportInvalidHumanProfessionException::ImportInvalidHumanProfessionException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

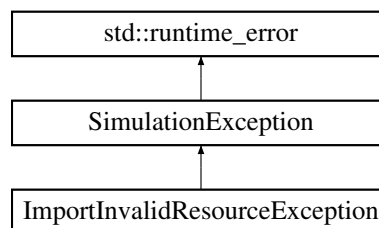
- src/exceptions/[FileExceptions.h](#)

## 6.30. ImportInvalidResourceException osztályreferencia

Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva.

```
#include <FileExceptions.h>
```

Az ImportInvalidResourceException osztály származási diagramja:



## Publikus tagfüggvények

- [ImportInvalidResourceException](#) (const std::string &msg)

### 6.30.1. Részletes leírás

Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva.

## 6.30.2. Konstruktorok és destruktorok dokumentációja

### 6.30.2.1. ImportInvalidResourceException()

```
ImportInvalidResourceException::ImportInvalidResourceException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- src/exceptions/[FileExceptions.h](#)

## 6.31. sf::IntRect osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [IntRect](#) ()
- [IntRect](#) (int l, int t, int w, int h)

### Publikus attribútumok

- int [left](#)
- int [top](#)
- int [width](#)
- int [height](#)

## 6.31.1. Konstruktorok és destruktorok dokumentációja

### 6.31.1.1. IntRect() [1/2]

```
sf::IntRect::IntRect ( )
```

### 6.31.1.2. IntRect() [2/2]

```
sf::IntRect::IntRect (
    int l,
    int t,
    int w,
    int h )
```

## 6.31.2. Adattagok dokumentációja

### 6.31.2.1. height

```
int sf::IntRect::height
```

### 6.31.2.2. left

```
int sf::IntRect::left
```

### 6.31.2.3. top

```
int sf::IntRect::top
```

### 6.31.2.4. width

```
int sf::IntRect::width
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

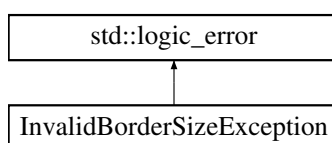
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.32. InvalidBorderSizeException osztályreferencia

Akkor kell dobni, ha egy világhatárnak nem jó értéket akarnak beállítani.

```
#include <WorldExceptions.h>
```

Az InvalidBorderSizeException osztály származási diagramja:



## Publikus tagfüggvények

- [InvalidBorderSizeException](#) (const std::string &msg)

### 6.32.1. Részletes leírás

Akkor kell dobni, ha egy világhatárnak nem jó értéket akarnak beállítani.

### 6.32.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.32.2.1. InvalidBorderSizeException()

```
InvalidBorderSizeException::InvalidBorderSizeException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

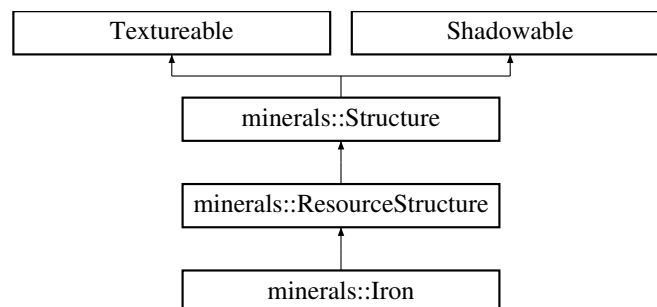
- src/exceptions/[WorldExceptions.h](#)

## 6.33. minerals::Iron osztályreferencia

A vasérc osztály leírása. Vasat ad, amikor kitermelik.

```
#include <Iron.h>
```

A minerals::Iron osztály származási diagramja:



## Publikus tagfüggvények

- [Iron](#) (int x, int y)  
*Konstruktor ami lerakja a házat egy (x,y) pontra.*
- [MINERAL\\_TYPE get\\_type](#) () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void [update\\_logic](#) (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- void [play\\_destroy\\_sound](#) ([SoundPlayer](#) &sound\_player) const override

## További örökölt tagok

### 6.33.1. Részletes leírás

A vasérc osztály leírása. Vasat ad, amikor kitermelik.

### 6.33.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.33.2.1. Iron()

```
minerals::Iron::Iron (
    int x,
    int y )
```

Konstruktork ami lerakja a házat egy (x,y) pontra.

### 6.33.3. Tagfüggvények dokumentációja

#### 6.33.3.1. get\_type()

```
MINERAL_TYPE minerals::Iron::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.33.3.2. play\_destroy\_sound()

```
void minerals::Iron::play_destroy_sound (
    SoundPlayer & sound_player ) const [override], [virtual]
```

Megvalósítja a következőket: [minerals::ResourceStructure](#).

#### 6.33.3.3. update\_logic()

```
void minerals::Iron::update_logic (
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.



## Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/world\\_object/Iron.h](#)
- [src/world\\_object/Iron.cpp](#)

## 6.34. sf::Keyboard osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus típusok

- enum [Key](#) : char {  
[Right](#) , [Left](#) , [Down](#) , [Up](#) ,  
[Space](#) , [Num1](#) , [Num2](#) , [Num3](#) ,  
[Num4](#) , [Num5](#) , [Num6](#) , [Num7](#) ,  
[Num8](#) , [Num9](#) , [Num0](#) }

### Statikus publikus tagfüggvények

- static bool [isKeyPressed](#) ([Key](#) key)
- static void [simulate\\_key\\_press](#) ([Key](#) key)
- static void [simulate\\_key\\_release](#) ([Key](#) key)

### 6.34.1. Enumeráció-tagok dokumentációja

#### 6.34.1.1. Key

```
enum sf::Keyboard::Key : char
```

#### Enumeráció-értékek

Right	
Left	
Down	
Up	
Space	
Num1	
Num2	
Num3	
Num4	
Num5	
Num6	
Num7	

## 6.34.2. Tagfüggvények dokumentációja

### 6.34.2.1. isKeyPressed()

```
bool sf::Keyboard::isKeyPressed (
    Key key ) [static]
```

### 6.34.2.2. simulate\_key\_press()

```
void sf::Keyboard::simulate_key_press (
    Key key ) [static]
```

### 6.34.2.3. simulate\_key\_release()

```
void sf::Keyboard::simulate_key_release (
    Key key ) [static]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

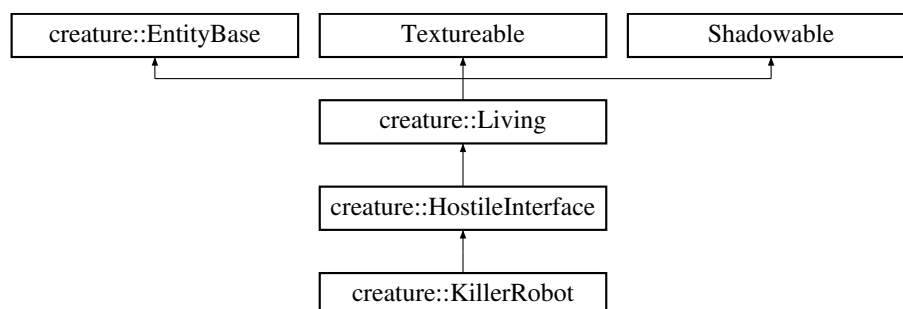
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.35. creature::KillerRobot osztályreferencia

A gyilkos robot osztály leírása.

```
#include <KillerRobot.h>
```

A creature::KillerRobot osztály származási diagramja:



## Publikus tagfüggvények

- `KillerRobot` (int x, int y)  
*Idéz egy gyilkos robotot egy pontos x és y koordinátára és beállítja az attribútumait.*
- `ENTITY_TYPE get_type ()` const override  
*Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.*
- void `die ()` override  
*Mi történjen, ha meghal az entitás.*
- void `update_logic (World &world, float deltaTime)` override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- void `draw_logic (sf::RenderWindow &window, float deltaTime, int ofx, int ofy)` override  
*Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*
- void `select_target (World &world)` override  
*A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.*
- `~KillerRobot ()`  
*Virtuális destruktork.*

## További örökölt tagok

### 6.35.1. Részletes leírás

A gyilkos robot osztály leírása.

A gyilkos robot egy ritka ellenség, aminek az az egy célja, hogy kiirtsa az emberiséget, majdnem egy évezredig él (999 évig pontosan), így vagy ő marad, vagy az emberiség.

### 6.35.2. Konstruktorok és destruktork dokumentációja

#### 6.35.2.1. KillerRobot()

```
creature::KillerRobot::KillerRobot (
    int x,
    int y )
```

Idéz egy gyilkos robotot egy pontos x és y koordinátára és beállítja az attribútumait.

#### Paraméterek

x	Az x koordináta.
y	Az y koordináta.

### 6.35.2.2. ~KillerRobot()

```
creature::KillerRobot::~~KillerRobot ( )
```

Virtuális destruktorkor.

## 6.35.3. Tagfüggvények dokumentációja

### 6.35.3.1. die()

```
void creature::KillerRobot::die ( ) [override], [virtual]
```

Mi történjen, ha meghal az entitás.

Megvalósítja a következőket: [creature::EntityBase](#).

### 6.35.3.2. draw\_logic()

```
void creature::KillerRobot::draw_logic (
    sf::RenderWindow & window,
    float deltaTime,
    int offx,
    int offy ) [override], [virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítja a következőket: [creature::Living](#).

### 6.35.3.3. get\_type()

```
ENTITY_TYPE creature::KillerRobot::get_type ( ) const [override], [virtual]
```

Visszaadja az entitás belső szimbólumát. Ez abba segít, hogy a vadállatok például csak embert támadjanak, Az emberek csak vadállatot.

## Visszatérési érték

A belső szimbólum.

Megvalósítja a következőket: [creature::EntityBase](#).

## 6.35.3.4. select\_target()

```
void creature::KillerRobot::select_target (
    World & world ) [override], [virtual]
```

A logikát írja le, ahogy az entitás a világba kiválasztja magának a célpontot.

## Paraméterek

<i>world</i>	A világ, amibe a célpontot kell választani.
--------------	---

Megvalósítja a következőket: [creature::HostileInterface](#).

## 6.35.3.5. update\_logic()

```
void creature::KillerRobot::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítja a következőket: [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

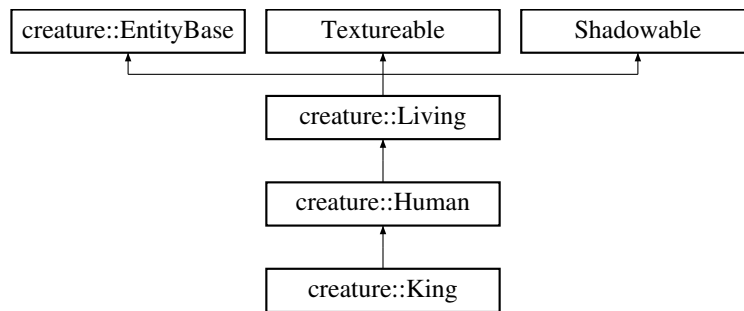
- [src/creatures/hostiles/KillerRobot.h](#)
- [src/creatures/hostiles/KillerRobot.cpp](#)

## 6.36. creature::King osztályreferencia

A király szakmájú ember osztály leírása.

```
#include <King.h>
```

A creature::King osztály származási diagramja:



## Publikus tagfüggvények

- **King** (int x, int y, ENTITY\_GENDER gender\_modifier)  
*Inicializál egy királyt egy pontos x és y koordinátára és beállítja az attribútumait.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- **~King** ()  
*A király destruktora.*

## További örökölt tagok

### 6.36.1. Részletes leírás

A király szakmájú ember osztály leírása.

Ez a szakmájú ember nem sokat csinál. A király szakma csak indikálja, hogy ő alapította a várost. Alapítást után csak örülten bolyong a világba.

### 6.36.2. Konstruktorok és destruktorkok dokumentációja

#### 6.36.2.1. King()

```

creature::King::King (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
  
```

Inicializál egy királyt egy pontos x és y koordinátára és beállítja az attribútumait.

#### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	A király neve.

### 6.36.2.2. ~King()

```
creature::King::~~King ( )
```

A király destruktora.

## 6.36.3. Tagfüggvények dokumentációja

### 6.36.3.1. update\_logic()

```
void creature::King::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

< Még mindig nincs?

Újrimplementált ősök: [creature::Human](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

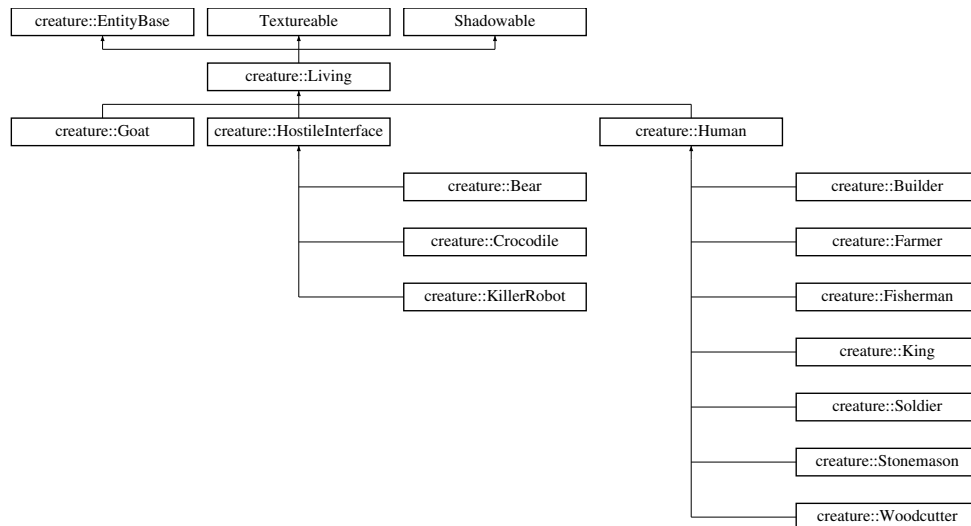
- [src/creatures/humans/King.h](#)
- [src/creatures/humans/King.cpp](#)

## 6.37. creature::Living osztályreferencia

Az élő entitások interface leírása.

```
#include <Living.h>
```

A creature::Living osztály származási diagramja:



## Publikus tagfüggvények

- void `look_left` ()  
*Balra nézeti az entitást.*
- void `look_right` ()  
*Jobbra nézeti az entitást.*
- void `damage` (Living \*dam\_by, int amm)  
*Ez a függvény jelzi, hogy megsebezték az entitást és azt, hogy ki sebezte meg.*
- void `set_state` (LIVINGSTATE newstate) override  
*Beállítja az entitás belső állapotát egy új értékre.*
- void `init_spritesheet_data` (int maxframes, double animspeed)  
*Beállítja az entitásnak azt, hogy hány képkockás animációja legyen és az milyen gyors legyen.*
- bool `setTexture` (const std::string &filename) override  
*Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- bool `setTheShadow` (const std::string &filename)  
*Beállítja az entitás árnyék textúráját.*
- void `setPosition` (double x, double y) override  
*Beállítja, hogy hova kell kirajzolni az entitást.*
- void `update_spritesheet` (float deltaTime)  
*Frissíti az entitás animációját az idő függvényében.*
- void `draw` (sf::RenderWindow &window) override  
*Kirajolja az élő entitást a render screen-re.*
- bool `needs_drawn` ()  
*Megnézi, hogy a felhasználó látja-e az entitást.*
- int `get_width` () const  
*Visszaadja az entitás vastagságát.*
- virtual Living \* `check_aggroed` () const  
*Visszaadja, hogy kire "ideges az entitás". Ez lehet az, hogy ki ütötte meg vagy hogy kit akar megenni.*
- virtual void `retarget` (Living \*new\_target)  
*Felidegesíti az entitást a kapott entitásra.*
- virtual void `update_logic` (World &world, float deltaTime)=0  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- virtual void `draw_logic` (sf::RenderWindow &window, float deltaTime, int offx, int offy)=0  
*Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.*



- void `shadow_logic` (`sf::RenderWindow` &window, float elapsed\_time, int offx, int offy)  
*Az entitás árnyékolás logikája, itt állítódik be az árnyék fázisa.*
- virtual `~Living` ()  
*Virtuális destruktork.*

## Védett attribútumok

- `Living * damaged_by` =nullptr  
*Arra az entitásra pointer, ami utoljára megsebezte.*

## Statikus védett attribútumok

- static constexpr int `MAX_CREATURE_SIZE` =64  
*Mekkora a maximum entitás, amit még a kamera culling nélkül kirajzol, akkor is ha annak a középpontja nincs benne a látótérbe.*

## További örökölt tagok

### 6.37.1. Részletes leírás

Az élő entitások interface leírása.

Ebbe minden deklarálva van, amire egy entitásnak szüksége van. Tud fordulni, animált képet rajzolni, futni, mozogni, támadni, meghalni, "csinálni a dolgát". Eltárolja, hogy melyik entitás sebezte meg utoljára.

### 6.37.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.37.2.1. `~Living()`

```
creature::Living::~~Living ( ) [virtual]
```

Virtuális destruktork.

### 6.37.3. Tagfüggvények dokumentációja

#### 6.37.3.1. check\_aggroed()

```
Living * creature::Living::check_aggroed ( ) const [virtual]
```

Visszaadja, hogy kire "ideges az entitás". Ez lehet az, hogy ki ütötte meg vagy hogy kit akar megenni.

##### Visszatérési érték

Az entitás, akire ideges. Nullpointer, ha nincs ilyen entitás.

Újrimplementáló leszármazottak: [creature::HostileInterface](#).

#### 6.37.3.2. damage()

```
void creature::Living::damage (
    Living * dam_by,
    int amm )
```

Ez a függvény jelzi, hogy megsebeztek az entitást és azt, hogy ki sebezte meg.

##### Paraméterek

<i>dam_by</i>	Az entitás, aki megsebezte.
<i>amm</i>	Mennyi sebzést kapott. Ezt levonja a metódus az entitás életéből.

#### 6.37.3.3. draw()

```
void creature::Living::draw (
    sf::RenderWindow & window ) [override], [virtual]
```

Kirajzolja az élő entitást a render screen-re.

##### Paraméterek

<i>window</i>	A render ablak.
---------------	-----------------

Megvalósítja a következőket: [Textureable](#).

#### 6.37.3.4. draw\_logic()

```
virtual void creature::Living::draw_logic (
    sf::RenderWindow & window,
```

```
float deltaTime,
int offx,
int offy ) [pure virtual]
```

Az entitás kirajzolás logikája, például az ember az árnyékát és képét rajzolja ki, de a robot csak a képét.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

Megvalósítják a következők: [creature::Human](#), [creature::KillerRobot](#), [creature::Crocodile](#), [creature::Bear](#) és [creature::Goat](#).

#### 6.37.3.5. get\_width()

```
int creature::Living::get_width ( ) const
```

Visszaadja az entitás vastagságát.

#### Visszatérési érték

Az entitás vastagsága.

#### 6.37.3.6. init\_spritesheet\_data()

```
void creature::Living::init_spritesheet_data (
    int maxframes,
    double animspeed )
```

Beállítja az entitásnak azt, hogy hány képkockás animációja legyen és az milyen gyors legyen.

#### Paraméterek

<i>maxframes</i>	A képkockák száma.
<i>animspeed</i>	Az animáció gyorsasága.

#### 6.37.3.7. look\_left()

```
void creature::Living::look_left ( )
```

Balra nézeti az entitást.

#### 6.37.3.8. look\_right()

```
void creature::Living::look_right ( )
```

Jobbra nézeti az entitást.

#### 6.37.3.9. needs\_drawn()

```
bool creature::Living::needs_drawn ( )
```

Megnézi, hogy a felhasználó látja-e az entitást.

##### Visszatérési érték

Benne van-e a látótérbe.

#### 6.37.3.10. retarget()

```
void creature::Living::retarget (
    Living * new_target ) [virtual]
```

Felidegesíti az entitást a kapott entitásra.

##### Paraméterek

<i>new_target</i>	Az entitás, akire dühösnek kell lennie.
-------------------	---

Újraimplementáló leszármazottak: [creature::HostileInterface](#).

#### 6.37.3.11. set\_state()

```
void creature::Living::set_state (
    LIVINGSTATE newstate ) [override], [virtual]
```

Beállítja az entitás belső állapotát egy új értékre.

## Paraméterek

<i>newstate</i>	Az új belső állapot.
-----------------	----------------------

Megvalósítja a következőket: [creature::EntityBase](#).

**6.37.3.12. setPosition()**

```
void creature::Living::setPosition (
    double x,
    double y ) [override], [virtual]
```

Beállítja, hogy hova kell kirajzolni az entitást.

Megvalósítja a következőket: [Textureable](#).

**6.37.3.13. setTexture()**

```
bool creature::Living::setTexture (
    const std::string & filename ) [override], [virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

## Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

## Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítja a következőket: [Textureable](#).

**6.37.3.14. setTheShadow()**

```
bool creature::Living::setTheShadow (
    const std::string & filename )
```

Beállítja az entitás árnyék textúráját.

## Paraméterek

<i>filename</i>	Az árnyék textúra elérési útvonala.
-----------------	-------------------------------------

### 6.37.3.15. shadow\_logic()

```
void creature::Living::shadow_logic (
    sf::RenderWindow & window,
    float elapsed_time,
    int offx,
    int offy )
```

Az entitás árnyékolás logikája, itt állítódik be az árnyék fázisa.

#### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>elapsed_time</i>	A szimuláció kezdete óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

### 6.37.3.16. update\_logic()

```
virtual void creature::Living::update_logic (
    World & world,
    float deltaTime ) [pure virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

#### Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

Megvalósítják a következők: [creature::Woodcutter](#), [creature::Stonemason](#), [creature::Soldier](#), [creature::King](#), [creature::Human](#), [creature::Fisherman](#), [creature::Farmer](#), [creature::Builder](#), [creature::AnglerMiner](#), [creature::KillerRobot](#), [creature::Crocodile](#), [creature::Bear](#) és [creature::Goat](#).

### 6.37.3.17. update\_spritesheet()

```
void creature::Living::update_spritesheet (
    float deltaTime )
```

Frissíti az entitás animációját az idő függvényében.

## Paraméterek

<code>deltaTime</code>	Az előző frissítés óta eltelt idő.
------------------------	------------------------------------

### 6.37.4. Adattagok dokumentációja

#### 6.37.4.1. damaged\_by

```
Living* creature::Living::damaged_by =nullptr [protected]
```

Arra az entitásra pointer, ami utoljára megsebezte.

#### 6.37.4.2. MAX\_CREATURE\_SIZE

```
constexpr int creature::Living::MAX_CREATURE_SIZE =64 [static], [constexpr], [protected]
```

Mekkora a maximum entitás, amit még a kamera culling nélkül kirajzol, akkor is ha annak a középpontja nincs benne a látótérbe.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/creatures/Living.h](#)
- [src/creatures/Living.cpp](#)

## 6.38. creature::LivingTexture osztályreferencia

Az élő entitások kinézetének adatai.

```
#include <EntityUtils.h>
```

### Publikus attribútumok

- `std::string idle_texture_path`  
*Az entitás semmit nem csinálás képének az elérési útvonala.*
- `std::string attack_texture_path`  
*Az entitás támadás képének az elérési útvonala.*
- `std::string walk_texture_path`  
*Az entitás sétálás képének az elérési útvonala.*
- `std::string run_texture_path`  
*Az entitás futás képének az elérési útvonala.*
- `std::string death_texture`  
*Az entitás meghalás képének az elérési útvonala.*
- `int frame_count`  
*Hány képkockából áll egy animáció.*
- `double animation_speed`  
*Milyen gyorsan változzon az animáció.*
- `double current_animation_time`  
*A jelenlegi animáció időt tárolja és ez alapján választja ki a kirajzolt képkockát.*

### 6.38.1. Részletes leírás

Az élő entíások kinézetének adatai.

### 6.38.2. Adattagok dokumentációja

#### 6.38.2.1. animation\_speed

```
double creature::LivingTexture::animation_speed
```

Milyen gyorsan változzon az animáció.

#### 6.38.2.2. attack\_texture\_path

```
std::string creature::LivingTexture::attack_texture_path
```

Az entitás támadás képének az elérési útvonala.

#### 6.38.2.3. current\_animation\_time

```
double creature::LivingTexture::current_animation_time
```

A jelenlegi animáció időt tárolja és ez alapján választja ki a kirajzolt képkockát.

#### 6.38.2.4. death\_texture

```
std::string creature::LivingTexture::death_texture
```

Az entitás meghalás képének az elérési útvonala.

#### 6.38.2.5. frame\_count

```
int creature::LivingTexture::frame_count
```

Hány képkockából áll egy animáció.



#### 6.38.2.6. idle\_texture\_path

```
std::string creature::LivingTexture::idle_texture_path
```

Az entitás semmit nem csinálás képének az elérési útvonala.

#### 6.38.2.7. run\_texture\_path

```
std::string creature::LivingTexture::run_texture_path
```

Az entitás futás képének az elérési útvonala.

#### 6.38.2.8. walk\_texture\_path

```
std::string creature::LivingTexture::walk_texture_path
```

Az entitás sétálás képének az elérési útvonala.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- src/creatures/[EntityUtils.h](#)

## 6.39. sf::Mouse osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus típusok

- enum [Mousedowntype](#) : char { [Right](#) , [Left](#) }

### Statikus publikus tagfüggvények

- static bool [isButtonPressed](#) ([Mousedowntype](#) key)
- static [Vector2i](#) [getPosition](#) ([RenderWindow](#) &window)
- static void [simulate\\_key\\_press](#) ([Mousedowntype](#) key)
- static void [simulate\\_key\\_release](#) ([Mousedowntype](#) key)

### 6.39.1. Enumeráció-tagok dokumentációja

#### 6.39.1.1. Mousedowntype

```
enum sf::Mouse::Mousedowntype : char
```

## Enumeráció-értékek

Right	
Left	

## 6.39.2. Tagfüggvények dokumentációja

### 6.39.2.1. getPosition()

```
Vector2i sf::Mouse::getPosition (
    RenderWindow & window ) [static]
```

### 6.39.2.2. isButtonPressed()

```
bool sf::Mouse::isButtonPressed (
    Mousedowntype key ) [static]
```

### 6.39.2.3. simulate\_key\_press()

```
void sf::Mouse::simulate_key_press (
    Mousedowntype key ) [static]
```

### 6.39.2.4. simulate\_key\_release()

```
void sf::Mouse::simulate_key_release (
    Mousedowntype key ) [static]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.40. sf::Music osztályreferencia

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- [Music](#) ()
- void [setLoop](#) (bool newval)
- void [setVolume](#) (double new\_db)
- [SoundSource::SoundSourceType](#) [getStatus](#) ()
- void [play](#) ()
- void [stop](#) ()
- bool [openFromFile](#) (const std::string &filepath)

### 6.40.1. Konstruktorkok és destruktorkok dokumentációja

#### 6.40.1.1. Music()

```
sf::Music::Music ( )
```

### 6.40.2. Tagfüggvények dokumentációja

#### 6.40.2.1. getStatus()

```
SoundSource::SoundSourceType sf::Music::getStatus ( )
```

#### 6.40.2.2. openFromFile()

```
bool sf::Music::openFromFile (
    const std::string & filepath )
```

#### 6.40.2.3. play()

```
void sf::Music::play ( )
```

#### 6.40.2.4. setLoop()

```
void sf::Music::setLoop (
    bool newval )
```

#### 6.40.2.5. setVolume()

```
void sf::Music::setVolume (
    double new_db )
```

#### 6.40.2.6. stop()

```
void sf::Music::stop ( )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

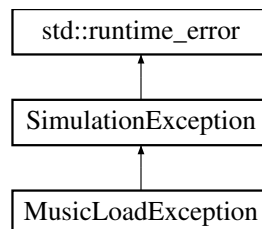
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

### 6.41. MusicLoadException osztályreferencia

Akkor kell dobni, ha egy zene hibásan lett lejátszva vagy hibás a beolvasott zene.

```
#include <MusicLoadException.h>
```

A MusicLoadException osztály származási diagramja:



#### Publikus tagfüggvények

- [MusicLoadException](#) (const std::string &msg)

#### 6.41.1. Részletes leírás

Akkor kell dobni, ha egy zene hibásan lett lejátszva vagy hibás a beolvasott zene.

#### 6.41.2. Konstruktorkok és destruktorkok dokumentációja

### 6.41.2.1. MusicLoadException()

```
MusicLoadException::MusicLoadException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [src/exceptions/MusicLoadException.h](#)

## 6.42. MusicPlayer osztályreferencia

A zene játsszó osztály leírása.

```
#include <MusicPlayer.h>
```

### Publikus tagfüggvények

- [MusicPlayer](#) ()  
*Alap konstruktor, beállítja a toggled és load\_music értéket hamisra.*
- void [load\\_music](#) (const std::string &filename)  
*Betölti az elérési útvonal végén lévő fájlból a zenét.*
- void [toggle\\_music](#) ()  
*Ki-be kapcsolja a zenét.*
- void [set\\_volume](#) (float vol)  
*Beállítja a hangerőt X decibelre.*
- [~MusicPlayer](#) ()  
*Destruktor, ami megállítja a zenét. Ez kiküszöböli a sound blasting-et, ami e-nélkül lenne.*

### 6.42.1. Részletes leírás

A zene játsszó osztály leírása.

Képes zenét betölteni, ki-be kapcsolni és lejátszani megadott hangerőn.

### 6.42.2. Konstruktorok és destruktorok dokumentációja

#### 6.42.2.1. MusicPlayer()

```
MusicPlayer::MusicPlayer ( )
```

Alap konstruktor, beállítja a toggled és load\_music értéket hamisra.

#### 6.42.2.2. ~MusicPlayer()

```
MusicPlayer::~~MusicPlayer ( )
```

Destruktor, ami megállítja a zenét. Ez kiküszöböli a sound blasting-et, ami e-nélkül lenne.

### 6.42.3. Tagfüggvények dokumentációja

#### 6.42.3.1. load\_music()

```
void MusicPlayer::load_music (
    const std::string & filename )
```

Betölti az elérési útvonal végén lévő fájlból a zenét.

##### Paraméterek

<i>filename</i>	A fájl elérési útvonala.
-----------------	--------------------------

#### 6.42.3.2. set\_volume()

```
void MusicPlayer::set_volume (
    float vol )
```

Beállítja a hangerőt X decibelre.

##### Paraméterek

<i>vol</i>	Mekkora decibel.
------------	------------------

#### 6.42.3.3. toggle\_music()

```
void MusicPlayer::toggle_music ( )
```

Ki-be kapcsolja a zenét.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/MusicPlayer.h](#)
- [src/MusicPlayer.cpp](#)

## 6.43. ObjectRegistry osztályreferencia

Az entitások és más világ objektumok lerakásának intézéséért felelős osztály.

```
#include <EntityPlacer.h>
```

### Publikus tagfüggvények

- void [register\\_type](#) (int id, SpawnFunc func)  
*Felvesz egy új idézés parancsot egy bizonyos gomb lenyomásra.*
- bool [spawn](#) (int id, [World](#) &world, [sf::Vector2i](#) &epos) const  
*Szimulál egy idézést a megadott gomb lenyomásra.*

#### 6.43.1. Részletes leírás

Az entitások és más világ objektumok lerakásának intézéséért felelős osztály.

#### 6.43.2. Tagfüggvények dokumentációja

##### 6.43.2.1. register\_type()

```
void ObjectRegistry::register_type (
    int id,
    SpawnFunc func )
```

Felvesz egy új idézés parancsot egy bizonyos gomb lenyomásra.

##### 6.43.2.2. spawn()

```
bool ObjectRegistry::spawn (
    int id,
    World & world,
    sf::Vector2i & epos ) const
```

Szimulál egy idézést a megadott gomb lenyomásra.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/EntityPlacer.h](#)
- [src/EntityPlacer.cpp](#)

## 6.44. gtest\_lite::ostreamRedir osztályreferencia

```
#include <modified_gtest_lite.h>
```

### Publikus tagfüggvények

- [ostreamRedir](#) (std::ostream &src, std::ostream &dst)
- [~ostreamRedir](#) ()

### 6.44.1. Részletes leírás

Segédsablon ostream átirányításához A destruktorkat visszaállít

### 6.44.2. Konstruktorkat és destruktorkat dokumentációja

#### 6.44.2.1. ostreamRedir()

```
gtest_lite::ostreamRedir::ostreamRedir (
    std::ostream & src,
    std::ostream & dst ) [inline]
```

#### 6.44.2.2. ~ostreamRedir()

```
gtest_lite::ostreamRedir::~~ostreamRedir ( ) [inline]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- src/external/[modified\\_gtest\\_lite.h](#)

## 6.45. PostProcessor osztályreferencia

A grafikus szépítő osztály leírása.

```
#include <PostProcessor.h>
```



## Publikus tagfüggvények

- `PostProcessor` ()  
*A konstruktor, a használt textúrák betöltése itt történik.*
- void `toggle_vignette` (bool newval)  
*Ki-be kapcsolja a vignettát.*
- void `toggle_noise` (bool newval)  
*Ki-be kapcsolja a zajt.*
- void `toggle_chromatic_aberration` (bool newval)  
*Ki-be kapcsolja a Chromatic aberration-t.*
- bool `setTextureFor` (sf::Sprite &what, const std::string &filename)  
*Beállít egy képnek egy új textúrát.*
- void `setRenderSize` (double x, double y)  
*Beállítja azt a négyzetet (0,0) (x,y)-ig, ahol a szépítő osztály dolgozni fog.*
- void `draw` (sf::RenderWindow &window)  
*Kirajzolódik az osztály.*
- void `setColorOverlay` (int r, int g, int b, int a)  
*Beállítja az új szín réteget.*

### 6.45.1. Részletes leírás

A grafikus szépítő osztály leírása.

Különböző szépítések beállíthatóak: Zaj, Szín, Chromatic aberration, Vignette.

### 6.45.2. Konstruktorok és destruktorok dokumentációja

#### 6.45.2.1. PostProcessor()

```
PostProcessor::PostProcessor ( )
```

A konstruktor, a használt textúrák betöltése itt történik.

### 6.45.3. Tagfüggvények dokumentációja

#### 6.45.3.1. draw()

```
void PostProcessor::draw (
    sf::RenderWindow & window )
```

Kirajzolódik az osztály.

## Paraméterek

<i>window</i>	Az ablak, amire rajzolódik.
---------------	-----------------------------

**6.45.3.2. setColorOverlay()**

```
void PostProcessor::setColorOverlay (
    int r,
    int g,
    int b,
    int a )
```

Beállítja az új szín réteget.

## Paraméterek

<i>r</i>	Piros komponens.
<i>g</i>	Zöld komponens.
<i>b</i>	Kék komponens.
<i>a</i>	Alfa komponens.

**6.45.3.3. setRenderSize()**

```
void PostProcessor::setRenderSize (
    double x,
    double y )
```

Beállítja azt a négyzetet (0,0) (x,y)-ig, ahol a szépítő osztály dolgozni fog.

## Paraméterek

<i>x</i>	A szélesség.
<i>y</i>	A magasság.

**6.45.3.4. setTextureFor()**

```
bool PostProcessor::setTextureFor (
    sf::Sprite & what,
    const std::string & filename )
```

Beállít egy képnek egy új textúrát.

## Paraméterek

<i>what</i>	Azt a képet, amit be kell állítani.
<i>filename</i>	Az új textúra elérési útvonala.

## Visszatérési érték

Sikerült-e.

**6.45.3.5. toggle\_chromatic\_aberration()**

```
void PostProcessor::toggle_chromatic_aberration (
    bool newval )
```

Ki-be kapcsolja a Chromatic aberration-t.

## Paraméterek

<i>newval</i>	Ha igaz, akkor ezen túl ki lesz rajzolva, különben nem lesz.
---------------	--

**6.45.3.6. toggle\_noise()**

```
void PostProcessor::toggle_noise (
    bool newval )
```

Ki-be kapcsolja a zajt.

## Paraméterek

<i>newval</i>	Ha igaz, akkor ezen túl ki lesz rajzolva, különben nem lesz.
---------------	--

**6.45.3.7. toggle\_vignette()**

```
void PostProcessor::toggle_vignette (
    bool newval )
```

Ki-be kapcsolja a vignettát.

## Paraméterek

<i>newval</i>	Ha igaz, akkor ezen túl ki lesz rajzolva, különben nem lesz.
---------------	--

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

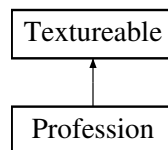
- src/[PostProcessor.h](#)
- src/[PostProcessor.cpp](#)

## 6.46. Profession osztályreferencia

A szakma osztály leírása.

```
#include <Profession.h>
```

A Profession osztály származási diagramja:



### Publikus tagfüggvények

- [Profession](#) (const std::string &intype)  
*A konstruktor, ami egy szimbólum alapján betölti az ikon képet.*
- bool [setTexture](#) (const std::string &filename) override  
*Beállítja egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- void [setPosition](#) (double x, double y) override  
*Beállítja, hogy hova kell rajzolni a textúrázható leszármozottat.*
- void [draw](#) (sf::RenderWindow &window) override  
*Kirajzolja az objektumot.*
- void [load\\_profession](#) (const std::string &new\_profession)  
*Egy szimbólum alapján betölti az ikon képet.*
- std::string [to\\_string](#) ()  
*Egy getter a szakma szimbólumához.*

### 6.46.1. Részletes leírás

A szakma osztály leírása.

Tárolja a szakma ikonját és szimbólumát is.

### 6.46.2. Konstruktorok és destruktorok dokumentációja

#### 6.46.2.1. Profession()

```
Profession::Profession (  
    const std::string & intype )
```

A konstruktor, ami egy szimbólum alapján betölti az ikon képet.

## Paraméterek

<i>intype</i>	A szakma szimbólum.
---------------	---------------------

### 6.46.3. Tagfüggvények dokumentációja

#### 6.46.3.1. draw()

```
void Profession::draw (
    sf::RenderWindow & window ) [override], [virtual]
```

Kirajzolja az objektumot.

## Paraméterek

<i>window</i>	Ahova ki kell rajzolni a textúrázható leszármozottat.
---------------	---

Megvalósítja a következőket: [Textureable](#).

#### 6.46.3.2. load\_profession()

```
void Profession::load_profession (
    const std::string & new_profession )
```

Egy szimbólum alapján betölti az ikon képet.

## Paraméterek

<i>intype</i>	A szakma szimbólum.
---------------	---------------------

#### 6.46.3.3. setPosition()

```
void Profession::setPosition (
    double x,
    double y ) [override], [virtual]
```

Beállítja, hogy hova kell rajzolni a textúrázható leszármozottat.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

Megvalósítja a következőket: [Textureable](#).

**6.46.3.4. setTexture()**

```
bool Profession::setTexture (
    const std::string & filename ) [override], [virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

## Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

## Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítja a következőket: [Textureable](#).

**6.46.3.5. to\_string()**

```
std::string Profession::to_string ( )
```

Egy getter a szakma szimbólumához.

## Visszatérési érték

A szakma szimbóluma szöveggént.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/[Profession.h](#)
- src/[Profession.cpp](#)

**6.47. RandomGenerator osztályreferencia**

Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály.

```
#include <Random_Gen.h>
```

## Publikus tagfüggvények

- `RandomGenerator (const RandomGenerator &)=delete`  
*A singleton pattern miatt törölve.*
- `RandomGenerator & operator= (const RandomGenerator &)=delete`  
*A singleton pattern miatt törölve.*
- `int get_random_int (int max)`  
*0 és a max-1 számok között visszaad egy véletlen számot.*

## Statikus publikus tagfüggvények

- `static RandomGenerator & get_instance ()`  
*Visszaad egy referenciát erre az osztály-ra.*

### 6.47.1. Részletes leírás

Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály.

Singleton pattern-t használ.

### 6.47.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.47.2.1. RandomGenerator()

```
RandomGenerator::RandomGenerator (  
    const RandomGenerator & ) [delete]
```

A singleton pattern miatt törölve.

### 6.47.3. Tagfüggvények dokumentációja

#### 6.47.3.1. get\_instance()

```
RandomGenerator & RandomGenerator::get_instance ( ) [static]
```

Visszaad egy referenciát erre az osztály-ra.

##### Visszatérési érték

A singleton-hoz egy referencia.

#### 6.47.3.2. get\_random\_int()

```
int RandomGenerator::get_random_int (  
    int max )
```

0 és a max-1 számok között visszaad egy véletlen számot.

## Paraméterek

<i>max</i>	A maximum érték, aminél már csak kisebb számokat ad vissza.
------------	---

## Visszatérési érték

A határ mérete.

**6.47.3.3. operator=()**

```
RandomGenerator& RandomGenerator::operator= (
    const RandomGenerator & ) [delete]
```

A singleton pattern miatt törölve.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

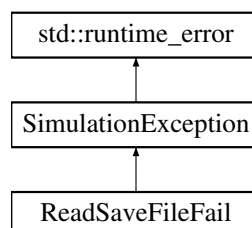
- [src/Random\\_Gen.h](#)
- [src/Random\\_Gen.cpp](#)

**6.48. ReadSaveFileFail osztályreferencia**

Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás.

```
#include <FileExceptions.h>
```

A ReadSaveFileFail osztály származási diagramja:

**Publikus tagfüggvények**

- [ReadSaveFileFail](#) (const std::string &msg)

**6.48.1. Részletes leírás**

Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás.



## 6.48.2. Konstruktorok és destruktorok dokumentációja

### 6.48.2.1. ReadSaveFileFail()

```
ReadSaveFileFail::ReadSaveFileFail (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- src/exceptions/[FileExceptions.h](#)

## 6.49. sf::RectangleShape osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- void [setFillColor](#) ([Color](#) new\_color)
- void [setSize](#) ([Vector2f](#) newsize)
- void [setPosition](#) ([Vector2f](#) newsize)

### Publikus attribútumok

- [Vector2f](#) position

## 6.49.1. Tagfüggvények dokumentációja

### 6.49.1.1. setFillColor()

```
void sf::RectangleShape::setFillColor (
    Color new_color )
```

### 6.49.1.2. setPosition()

```
void sf::RectangleShape::setPosition (
    Vector2f newsize )
```

### 6.49.1.3. setSize()

```
void sf::RectangleShape::setSize (
    Vector2f newsize )
```

## 6.49.2. Adattagok dokumentációja

### 6.49.2.1. position

```
Vector2f sf::RectangleShape::position
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.50. sf::RenderStates osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [RenderStates](#) ()
- void [setBlendMode](#) ([BlendMode](#) mode)
- void [setTransform](#) (const float newTransform[4][4])

### Publikus attribútumok

- [Transform](#) transform
- [BlendMode](#) blendMode

## 6.50.1. Konstruktorkok és destruktorok dokumentációja

### 6.50.1.1. RenderStates()

```
sf::RenderStates::RenderStates ( )
```

## 6.50.2. Tagfüggvények dokumentációja

### 6.50.2.1. setBlendMode()

```
void sf::RenderStates::setBlendMode (
    BlendMode mode )
```

### 6.50.2.2. setTransform()

```
void sf::RenderStates::setTransform (
    const float newTransform[4][4] )
```

## 6.50.3. Adattagok dokumentációja

### 6.50.3.1. blendMode

```
BlendMode sf::RenderStates::blendMode
```

### 6.50.3.2. transform

```
Transform sf::RenderStates::transform
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/fake\_sfml/fake\_sfml.h
- src/fake\_sfml/fake\_sfml.cpp

## 6.51. sf::RenderWindow osztályreferencia

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- [RenderWindow](#) ()
- [RenderWindow](#) (const std::string &title\_, std::size\_t w, std::size\_t h)
- [RenderWindow](#) ([VideoMode](#) vmode, const std::string &title\_)
- void [create](#) (const std::string &title\_, std::size\_t w, std::size\_t h)
- bool [isOpen](#) () const
- bool [pollEvent](#) ([Event](#) &event)
- void [close](#) ()
- void [setFramerateLimit](#) (std::size\_t limit)
- void [clear](#) ()
- void [draw](#) (const [Sprite](#) &sprite)
- void [draw](#) (const [Sprite](#) &sprite, [RenderStates](#) states)
- void [draw](#) (const [RectangleShape](#) &shape)
- void [display](#) ()
- void [clear](#) ([Color](#) clr)

## 6.51.1. Konstruktorkok és destruktorkok dokumentációja

### 6.51.1.1. [RenderWindow\(\)](#) [1/3]

```
sf::RenderWindow::RenderWindow ( )
```

### 6.51.1.2. [RenderWindow\(\)](#) [2/3]

```
sf::RenderWindow::RenderWindow (
    const std::string & title_,
    std::size_t w,
    std::size_t h )
```

### 6.51.1.3. [RenderWindow\(\)](#) [3/3]

```
sf::RenderWindow::RenderWindow (
    VideoMode vmode,
    const std::string & title_ )
```

## 6.51.2. Tagfüggvények dokumentációja

**6.51.2.1. clear()** [1/2]

```
void sf::RenderWindow::clear ( )
```

**6.51.2.2. clear()** [2/2]

```
void sf::RenderWindow::clear (
    Color clr )
```

**6.51.2.3. close()**

```
void sf::RenderWindow::close ( )
```

**6.51.2.4. create()**

```
void sf::RenderWindow::create (
    const std::string & title_,
    std::size_t w,
    std::size_t h )
```

**6.51.2.5. display()**

```
void sf::RenderWindow::display ( )
```

**6.51.2.6. draw()** [1/3]

```
void sf::RenderWindow::draw (
    const RectangleShape & shape )
```

**6.51.2.7. draw()** [2/3]

```
void sf::RenderWindow::draw (
    const Sprite & sprite )
```

**6.51.2.8. draw()** [3/3]

```
void sf::RenderWindow::draw (
    const Sprite & sprite,
    RenderStates states )
```

**6.51.2.9. isOpen()**

```
bool sf::RenderWindow::isOpen ( ) const
```

**6.51.2.10. pollEvent()**

```
bool sf::RenderWindow::pollEvent (
    Event & event )
```

**6.51.2.11. setFrameRateLimit()**

```
void sf::RenderWindow::setFramerateLimit (
    std::size_t limit )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

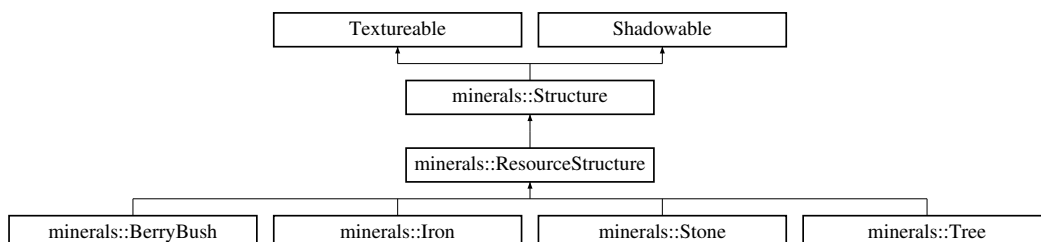
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

**6.52. minerals::ResourceStructure osztályreferencia**

Az erőforrás struktúra osztály leírása.

```
#include <ResourceStructure.h>
```

A minerals::ResourceStructure osztály származási diagramja:



## Publikus tagfüggvények

- bool `get_harvested` () const  
*Megadja, hogy kitermelt-e az erőforrás.*
- `ResourceStructure` (int x, int y)  
*Konstruktor ami lerakja az erőforrást egy (x,y) pontra.*
- virtual bool `harvest` ()  
*virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.*
- virtual `~ResourceStructure` ()=default  
*Virtuális destruktork.*
- virtual void `play_destroy_sound` (`SoundPlayer` &sound\_player) const =0
- bool `get_needs_remove` () const  
*Rombolás hangjának lejátszása.*

## Védett attribútumok

- float `inner_timer`  
*Az idéződés óta eltelt idő.*
- bool `harvested`  
*Ki van-e termelve az erőforrás.*
- bool `needs_remove`

## További örökölt tagok

### 6.52.1. Részletes leírás

Az erőforrás struktúra osztály leírása.

Ez az interface rendelkezik azokról az adatokról, hogy kibányászható-e még ez az objektum és ahhoz szükséges metódusokkal.

### 6.52.2. Konstruktorok és destruktork dokumentációja

#### 6.52.2.1. ResourceStructure()

```
minerals::ResourceStructure::ResourceStructure (
    int x,
    int y )
```

Konstruktor ami lerakja az erőforrást egy (x,y) pontra.

#### 6.52.2.2. ~ResourceStructure()

```
virtual minerals::ResourceStructure::~~ResourceStructure ( ) [virtual], [default]
```

Virtuális destruktor.

### 6.52.3. Tagfüggvények dokumentációja

#### 6.52.3.1. get\_harvested()

```
bool minerals::ResourceStructure::get_harvested ( ) const
```

Megadja, hogy kitermelt-e az erőforrás.

#### 6.52.3.2. get\_needs\_remove()

```
bool minerals::ResourceStructure::get_needs_remove ( ) const
```

Rombolás hangjának lejátszása.

#### 6.52.3.3. harvest()

```
bool minerals::ResourceStructure::harvest ( ) [virtual]
```

virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.

Újraimplementáló leszármazottak: [minerals::BerryBush](#).

#### 6.52.3.4. play\_destroy\_sound()

```
virtual void minerals::ResourceStructure::play_destroy_sound (
    SoundPlayer & sound_player ) const [pure virtual]
```

Megvalósítják a következők: [minerals::Tree](#), [minerals::Stone](#), [minerals::Iron](#) és [minerals::BerryBush](#).

### 6.52.4. Adattagok dokumentációja



#### 6.52.4.1. harvested

```
bool minerals::ResourceStructure::harvested [protected]
```

Ki van-e termelve az erőforrás.

#### 6.52.4.2. inner\_timer

```
float minerals::ResourceStructure::inner_timer [protected]
```

Az idéződéskésztési idő.

#### 6.52.4.3. needs\_remove

```
bool minerals::ResourceStructure::needs_remove [protected]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/world\_object/[ResourceStructure.h](#)
- src/world\_object/[ResourceStructure.cpp](#)

## 6.53. RoleOption struktúráreferencia

Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni.

```
#include <SaveHelpers.h>
```

### Publikus attribútumok

- std::function< [creature::Human](#) \*(int, int, [creature::ENTITY\\_GENDER](#))> [create](#)
- std::vector< std::pair< std::string, int > > [requirements](#)

#### 6.53.1. Részletes leírás

Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni.

#### 6.53.2. Adattagok dokumentációja

#### 6.53.2.1. create

```
std::function<creature::Human*(int, int, creature::ENTITY_GENDER)> RoleOption::create
```

#### 6.53.2.2. requirements

```
std::vector<std::pair<std::string, int> > RoleOption::requirements
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- src/[SaveHelpers.h](#)

### 6.54. SaveHelper osztályreferencia

Factory-k.

```
#include <SaveHelpers.h>
```

#### Statikus publikus tagfüggvények

- static const std::unordered\_map< std::string, CreatureFactory > & [getCreatureFactory](#) ()
- static const std::unordered\_map< std::string, HumanFactory > & [getHumanFactory](#) ()
- static const std::unordered\_map< std::string, ResourceFactory > & [getResourceFactory](#) ()
- static const std::vector< [RoleOption](#) > & [get\\_roles](#) ()
- static std::string [trim\\_brackets](#) (const std::string &s)

#### 6.54.1. Részletes leírás

Factory-k.

#### 6.54.2. Tagfüggvények dokumentációja

##### 6.54.2.1. get\_roles()

```
const std::vector< RoleOption > & SaveHelper::get_roles ( ) [static]
```

#### 6.54.2.2. getCreatureFactory()

```
const std::unordered_map< std::string, SaveHelper::CreatureFactory > & SaveHelper::getCreatureFactory ( ) [static]
```

#### 6.54.2.3. getHumanFactory()

```
const std::unordered_map< std::string, SaveHelper::HumanFactory > & SaveHelper::getHumanFactory ( ) [static]
```

#### 6.54.2.4. getResourceFactory()

```
const std::unordered_map< std::string, SaveHelper::ResourceFactory > & SaveHelper::getResourceFactory ( ) [static]
```

#### 6.54.2.5. trim\_brackets()

```
std::string SaveHelper::trim_brackets (
    const std::string & s ) [static]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/SaveHelpers.h](#)
- [src/SaveHelpers.cpp](#)

## 6.55. SaveManager osztályreferencia

A fájl menedzseléshez szolgáló osztály leírása.

```
#include <SaveManager.h>
```

### Publikus tagfüggvények

- [SaveManager](#) (const std::string &file)  
*A konstruktor, ahol beállítható, hogy mi a neve és elérési útvonala a mentés fájlnak.*
- void [saveFile](#) (World &world)  
*Elment egy világot a fájlba.*
- void [loadFile](#) (World &world)  
*Elment egy fájlt a világba.*
- void [deleteFile](#) ()  
*Kitörli a jelenlegi mentés fájl tartalmát.*

### 6.55.1. Részletes leírás

A fájl menedzseléshez szolgáló osztály leírása.

Képes betölteni mentést, eltárolni és törölni is.

### 6.55.2. Konstruktorok és destruktorok dokumentációja

#### 6.55.2.1. SaveManager()

```
SaveManager::SaveManager (
    const std::string & file )
```

A konstruktor, ahol beállítható, hogy mi a neve és elérési útvonala a mentés fájlnak.

##### Paraméterek

<i>file</i>	Az elérési útvonal.
-------------	---------------------

### 6.55.3. Tagfüggvények dokumentációja

#### 6.55.3.1. deleteFile()

```
void SaveManager::deleteFile ( )
```

Kitörli a jelenlegi mentés fájl tartalmát.

#### 6.55.3.2. loadFile()

```
void SaveManager::loadFile (
    World & world )
```

Elment egy fájlt a világba.

##### Paraméterek

<i>world</i>	Referencia a világra.
--------------	-----------------------

## 6.55.3.3. saveFile()

```
void SaveManager::saveFile (
    World & world )
```

Elment egy világot a fájlba.

## Paraméterek

<i>world</i>	Referencia a világra.
--------------	-----------------------

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

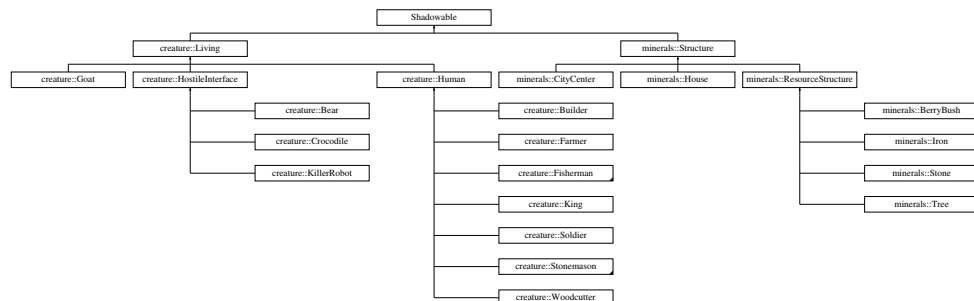
- [src/SaveManager.h](#)
- [src/SaveManager.cpp](#)

## 6.56. Shadowable osztályreferencia

Az árnyékoláshoz szükséges interface.

```
#include <Shadowable.h>
```

A Shadowable osztály származási diagramja:



## Publikus tagfüggvények

- double [get\\_height\\_offset](#) () const  
*Egy getter a magasságpont eltolásának megszerzésére.*
- int [get\\_shadow\\_strength](#) () const  
*Egy getter az árnyék erősségére.*
- float [get\\_skew\\_offset](#) () const  
*Egy getter az elnyújtás mértékére.*
- void [set\\_height\\_offset](#) (double new\_val)  
*Egy setter a magasságpont eltolásához.*
- void [set\\_shadow\\_strength](#) (int new\_val)  
*Egy setter az árnyék erősséghez.*
- void [set\\_skew\\_offset](#) (float new\_val)  
*Egy setter a elnyújtás mértékéhez.*

- virtual `~Shadowable()`=default  
*Virtuális destruktork.*
- bool `setShadowTexture` (const std::string &filename)  
*Az árnyék kinézetét állítja be.*
- void `setShadow` (float ySize, float xSkew)  
*Beállítja az árnyék nyújtását és eltolását.*
- void `setShadowDayNightCycle` (float delta\_time)  
*Beállítja az árnyék nyújtását a napszaktól függően.*
- void `setShadowPosition` (double x, double y)  
*Beállítja az árnyék helyét.*
- void `drawShadow` (sf::RenderWindow &window)  
*Kirajzolja az árnyékot.*

## Védett attribútumok

- double `height_offset` =0.0  
*Milyen messze kezdődjön az árnyék az objektum alsó pontjától.*

### 6.56.1. Részletes leírás

Az árnyékoláshoz szükséges interface.

Tárolja az árnyék textúráját, valamiért felelős annak mozgatásáért és rendes kirajzolásáért.

### 6.56.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.56.2.1. `~Shadowable()`

```
virtual Shadowable::~~Shadowable ( ) [virtual], [default]
```

Virtuális destruktork.

### 6.56.3. Tagfüggvények dokumentációja

#### 6.56.3.1. `drawShadow()`

```
void Shadowable::drawShadow (
    sf::RenderWindow & window )
```

Kirajzolja az árnyékot.

## Paraméterek

<i>window</i>	Az ablak, ahova ki kell rajzolni.
---------------	-----------------------------------

**6.56.3.2. get\_height\_offset()**

```
double Shadowable::get_height_offset ( ) const
```

Egy getter a magasságpont eltolásának megszerzésére.

**Visszatérési érték**

A magasságpont eltolásának értéke.

**6.56.3.3. get\_shadow\_strength()**

```
int Shadowable::get_shadow_strength ( ) const
```

Egy getter az árnyék erősségre.

**Visszatérési érték**

Az árnyék erőssége.

**6.56.3.4. get\_skew\_offset()**

```
float Shadowable::get_skew_offset ( ) const
```

Egy getter az elnyújtás mértékére.

**Visszatérési érték**

Az elnyújtás mértéke.

**6.56.3.5. set\_height\_offset()**

```
void Shadowable::set_height_offset (
    double new_val )
```

Egy setter a magasságpont eltolásához.

## Paraméterek

<i>new_val</i>	Az új érték, amire be kell állítani.
----------------	--------------------------------------

**6.56.3.6. set\_shadow\_strength()**

```
void Shadowable::set_shadow_strength (
    int new_val )
```

Egy setter az árnyék erősségéhez.

## Paraméterek

<i>new_val</i>	Az új érték, amire be kell állítani.
----------------	--------------------------------------

**6.56.3.7. set\_skew\_offset()**

```
void Shadowable::set_skew_offset (
    float new_val )
```

Egy setter a elnyújtás mértékéhez.

## Paraméterek

<i>new_val</i>	Az új érték, amire be kell állítani.
----------------	--------------------------------------

**6.56.3.8. setShadow()**

```
void Shadowable::setShadow (
    float ySize,
    float xSkew )
```

Beállítja az árnyék nyújtását és eltolását.

## Paraméterek

<i>ySize</i>	Az Y tengelyen való nyújtás.
<i>xSkew</i>	Az X elnyújtás.



#### 6.56.3.9. setShadowDayNightCycle()

```
void Shadowable::setShadowDayNightCycle (
    float delta_time )
```

Beállítja az árnyék nyújtását a napszaktól függően.

##### Paraméterek

<i>delta_time</i>	Az előző frissítés óta eltelt idő.
-------------------	------------------------------------

#### 6.56.3.10. setShadowPosition()

```
void Shadowable::setShadowPosition (
    double x,
    double y )
```

Beállítja az árnyék helyét.

##### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

#### 6.56.3.11. setShadowTexture()

```
bool Shadowable::setShadowTexture (
    const std::string & filename )
```

Az árnyék kinézetét állítja be.

##### Paraméterek

<i>filename</i>	A textúra elérési útvonala
-----------------	----------------------------

##### Visszatérési érték

Igaz, ha sikeres a textúra beállítás, különben hamis.

### 6.56.4. Adattagok dokumentációja

#### 6.56.4.1. height\_offset

```
double Shadowable::height_offset =0.0 [protected]
```

Milyen messze kezdődjön az árnyék az objektum alsó pontjától.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

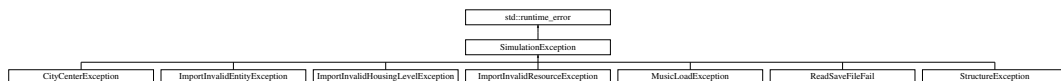
- [src/Shadowable.h](#)
- [src/Shadowable.cpp](#)

## 6.57. SimulationException osztályreferencia

Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik.

```
#include <SimulationException.h>
```

A SimulationException osztály származási diagramja:



### Publikus tagfüggvények

- [SimulationException](#) (const std::string &msg)

#### 6.57.1. Részletes leírás

Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik.

#### 6.57.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.57.2.1. SimulationException()

```
SimulationException::SimulationException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

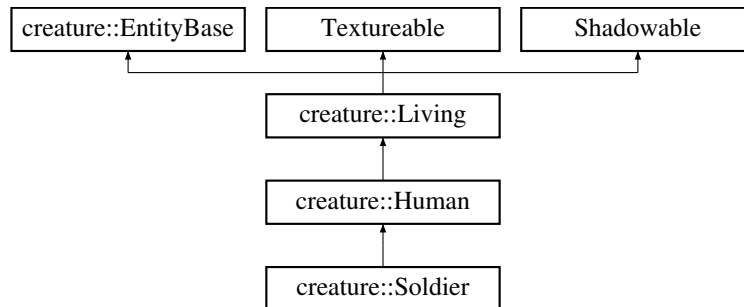
- [src/exceptions/SimulationException.h](#)

## 6.58. creature::Soldier osztályreferencia

A katona szakmájú ember osztály leírása.

```
#include <Soldier.h>
```

A creature::Soldier osztály származási diagramja:



### Publikus tagfüggvények

- **Soldier** (int x, int y, ENTITY\_GENDER gender\_modifier)  
*Inicializál egy katonát egy pontos x és y koordinátára és beállítja az attribútumait.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- **~Soldier** ()  
*A katona destruktora.*

### További örökölt tagok

#### 6.58.1. Részletes leírás

A katona szakmájú ember osztály leírása.

Ez a szakmájú ember vadászik állatokat és megvédi a népet az ellenséges entitásoktól.

#### 6.58.2. Konstruktorok és destruktorkok dokumentációja

##### 6.58.2.1. Soldier()

```
creature::Soldier::Soldier (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy katonát egy pontos x és y koordinátára és beállítja az attribútumait.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	A katona neve.

**6.58.2.2. ~Soldier()**

```
creature::Soldier::~~Soldier ( )
```

A katona destruktora.

**6.58.3. Tagfüggvények dokumentációja****6.58.3.1. update\_logic()**

```
void creature::Soldier::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

< Még mindig nincs?

Újrimplementált ősök: [creature::Human](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/creatures/humans/Soldier.h](#)
- [src/creatures/humans/Soldier.cpp](#)

**6.59. sf::Sound osztályreferencia**

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- void [setBuffer](#) ([SoundBuffer](#) &buf)
- void [play](#) ()
- void [stop](#) ()
- [~Sound](#) ()

### 6.59.1. Konstruktorkok és destruktorkok dokumentációja

#### 6.59.1.1. ~Sound()

```
sf::Sound::~~Sound ( )
```

### 6.59.2. Tagfüggvények dokumentációja

#### 6.59.2.1. play()

```
void sf::Sound::play ( )
```

#### 6.59.2.2. setBuffer()

```
void sf::Sound::setBuffer (
    SoundBuffer & buf )
```

#### 6.59.2.3. stop()

```
void sf::Sound::stop ( )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.60. sf::SoundBuffer osztályreferencia

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- bool [loadFromFile](#) (const std::string &filepath)

### 6.60.1. Tagfüggvények dokumentációja

#### 6.60.1.1. loadFromFile()

```
bool sf::SoundBuffer::loadFromFile (
    const std::string & filepath )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/fake\_sfml/fake\_sfml.h
- src/fake\_sfml/fake\_sfml.cpp

## 6.61. SoundPlayer osztályreferencia

A hanglejátszó osztály leírása.

```
#include <SoundPlayer.h>
```

## Publikus tagfüggvények

- void [load\\_sound](#) (const std::string &filename)  
*Betölt egy hangot az elérési útvonalról.*
- void [play\\_sound](#) (const std::string &filename)  
*Lejátszik egy hangot az elérési útvonalról. Ha még nem volt ez betöltve akkor először betölti.*
- void [stop\\_sound](#) ()  
*Megállítja az éppen lejátszott hangot.*

### 6.61.1. Részletes leírás

A hanglejátszó osztály leírása.

Képes hangokat betölteni, elindítani, lejátszani és megállítani.

### 6.61.2. Tagfüggvények dokumentációja

#### 6.61.2.1. load\_sound()

```
void SoundPlayer::load_sound (
    const std::string & filename )
```

Betölt egy hangot az elérési útvonalról.

## Paraméterek

<i>filename</i>	Az elérési útvonal.
-----------------	---------------------

**6.61.2.2. play\_sound()**

```
void SoundPlayer::play_sound (
    const std::string & filename )
```

Lejátszik egy hangot az elérési útvonalról. Ha még nem volt ez betöltve akkor először betölti.

## Paraméterek

<i>filename</i>	Az elérési útvonal.
-----------------	---------------------

**6.61.2.3. stop\_sound()**

```
void SoundPlayer::stop_sound ( )
```

Megállítja az éppen lejátszott hangot.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/[SoundPlayer.h](#)
- src/[SoundPlayer.cpp](#)

**6.62. sf::SoundSource osztályreferencia**

```
#include <fake_sfml.h>
```

**Publikus típusok**

- enum [SoundSourceType](#) : char { [Playing](#) , [Stopped](#) , [Paused](#) }

**Publikus tagfüggvények**

- [SoundSource](#) ()
- virtual [~SoundSource](#) ()=default

## Publikus attribútumok

- [SoundSourceType](#) type

## 6.62.1. Enumeráció-tagok dokumentációja

### 6.62.1.1. SoundSourceType

```
enum sf::SoundSource::SoundSourceType : char
```

Enumeráció-értékek

Playing	
Stopped	
Paused	

## 6.62.2. Konstruktorok és destruktorok dokumentációja

### 6.62.2.1. SoundSource()

```
sf::SoundSource::SoundSource ( )
```

### 6.62.2.2. ~SoundSource()

```
virtual sf::SoundSource::~~SoundSource ( ) [virtual], [default]
```

## 6.62.3. Adattagok dokumentációja

### 6.62.3.1. type

[SoundSourceType](#) sf::SoundSource::type

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)



## 6.63. sf::Sprite osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [Sprite](#) ()
- void [setTexture](#) ([Texture](#) &tex)
- void [setTextureRect](#) (const [IntRect](#) &tex)
- [Texture](#) \* [getTexture](#) ()
- void [setPosition](#) (float x, float y)
- void [setOrigin](#) (float \_x, float \_y)
- void [setRotation](#) (float deg)
- [Vector2f](#) [getPosition](#) ()
- void [setScale](#) (float sx, float sy)
- [FloatRect](#) [getLocalBounds](#) ()
- [FloatRect](#) [getGlobalBounds](#) ()
- [FloatRect](#) [getGlobalBounds](#) () const
- void [draw](#) () const
- void [setColor](#) ([Color](#) \_clr)
- [~Sprite](#) ()

### 6.63.1. Konstruktork és destruktork dokumentációja

#### 6.63.1.1. Sprite()

```
sf::Sprite::Sprite ( )
```

#### 6.63.1.2. ~Sprite()

```
sf::Sprite::~~Sprite ( )
```

### 6.63.2. Tagfüggvények dokumentációja

#### 6.63.2.1. draw()

```
void sf::Sprite::draw ( ) const
```

**6.63.2.2. getGlobalBounds()** [1/2]

```
FloatRect sf::Sprite::getGlobalBounds ( )
```

**6.63.2.3. getGlobalBounds()** [2/2]

```
FloatRect sf::Sprite::getGlobalBounds ( ) const
```

**6.63.2.4. getLocalBounds()**

```
FloatRect sf::Sprite::getLocalBounds ( )
```

**6.63.2.5. getPosition()**

```
Vector2f sf::Sprite::getPosition ( )
```

**6.63.2.6. getTexture()**

```
Texture * sf::Sprite::getTexture ( )
```

**6.63.2.7. setColor()**

```
void sf::Sprite::setColor (
    Color _clr )
```

**6.63.2.8. setOrigin()**

```
void sf::Sprite::setOrigin (
    float _x,
    float _y )
```

**6.63.2.9. setPosition()**

```
void sf::Sprite::setPosition (
    float x,
    float y )
```

**6.63.2.10. setRotation()**

```
void sf::Sprite::setRotation (
    float deg )
```

**6.63.2.11. setScale()**

```
void sf::Sprite::setScale (
    float sx,
    float sy )
```

**6.63.2.12. setTexture()**

```
void sf::Sprite::setTexture (
    Texture & tex )
```

**6.63.2.13. setTextureRect()**

```
void sf::Sprite::setTextureRect (
    const IntRect & tex )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

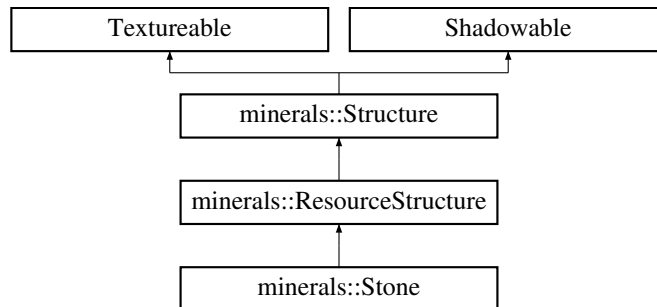
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.64. minerals::Stone osztályreferencia

A kő osztály leírása. követ ad, amikor kitermelik.

```
#include <Stone.h>
```

A minerals::Stone osztály származási diagramja:



### Publikus tagfüggvények

- [Stone](#) (int x, int y)  
*Konstruktor ami lerakja a házat egy (x,y) pontra.*
- [MINERAL\\_TYPE get\\_type](#) () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void [update\\_logic](#) (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- void [play\\_destroy\\_sound](#) ([SoundPlayer](#) &sound\_player) const override

### További örökölt tagok

#### 6.64.1. Részletes leírás

A kő osztály leírása. követ ad, amikor kitermelik.

#### 6.64.2. Konstruktorok és destruktorok dokumentációja

##### 6.64.2.1. Stone()

```
minerals::Stone::Stone (
    int x,
    int y )
```

Konstruktor ami lerakja a házat egy (x,y) pontra.

### 6.64.3. Tagfüggvények dokumentációja

#### 6.64.3.1. get\_type()

```
MINERAL_TYPE minerals::Stone::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.64.3.2. play\_destroy\_sound()

```
void minerals::Stone::play_destroy_sound (
    SoundPlayer & sound_player ) const [override], [virtual]
```

Megvalósítja a következőket: [minerals::ResourceStructure](#).

#### 6.64.3.3. update\_logic()

```
void minerals::Stone::update_logic (
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.

##### Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

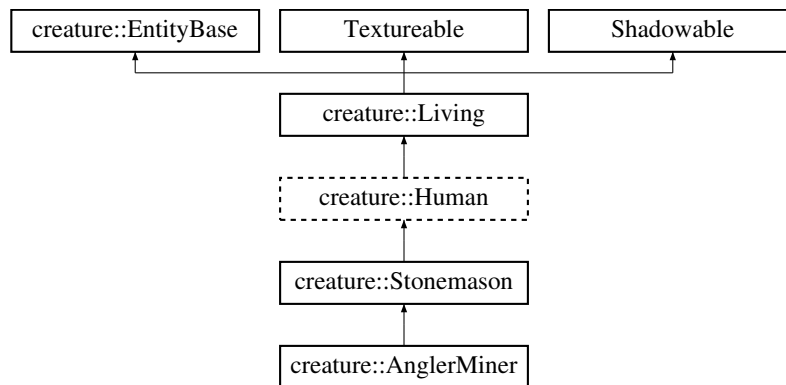
- [src/world\\_object/Stone.h](#)
- [src/world\\_object/Stone.cpp](#)

## 6.65. creature::Stonemason osztályreferencia

A bányász szakmájú ember osztály leírása.

```
#include <Stonemason.h>
```

A creature::Stonemason osztály származási diagramja:



## Publikus tagfüggvények

- **Stonemason** (int x, int y, ENTITY\_GENDER gender\_modifier)  
*Inicializál egy bányászt egy pontos x és y koordinátára és beállítja az attribútumait.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- ~**Stonemason** ()  
*A bányász destruktora.*

## Védett tagfüggvények

- void **try\_mine** (World &world)  
*Megpróbál magának egy kőoszlopot vagy vasércet keresni, amit aztán ki fog bányászni.*

## Védett attribútumok

- bool **mining\_iron**  
*Vasat bányászik-e? Ha hamis, akkor követ bányászik.*

## További örökölt tagok

### 6.65.1. Részletes leírás

A bányász szakmájú ember osztály leírása.

Ez a szakmájú ember követ vagy vasat keres és kitermeli őket, így követ és vasat szerez.

### 6.65.2. Konstruktorok és destruktorkok dokumentációja

#### 6.65.2.1. Stonemason()

```

creature::Stonemason::Stonemason (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
  
```

Inicializál egy bányászt egy pontos x és y koordinátára és beállítja az attribútumait.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	A bányász neme.

## 6.65.2.2. ~Stonemason()

```
creature::Stonemason::~~Stonemason ( )
```

A bányász destruktora.

## 6.65.3. Tagfüggvények dokumentációja

## 6.65.3.1. try\_mine()

```
void creature::Stonemason::try_mine (
    World & world ) [protected]
```

Megpróbál magának egy kőoszlopot vagy vasércet keresni, amit aztán ki fog bányászni.

## Paraméterek

<i>world</i>	A világ, amibe keresni kell az érceket.
--------------	---

## 6.65.3.2. update\_logic()

```
void creature::Stonemason::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

< Még mindig nincs?

Újrimplementált ősök: [creature::Human](#).

## 6.65.4. Adattagok dokumentációja

### 6.65.4.1. mining\_iron

```
bool creature::Stonemason::mining_iron [protected]
```

Vasat bányászik-e? Ha hamis, akkor követ bányászik.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

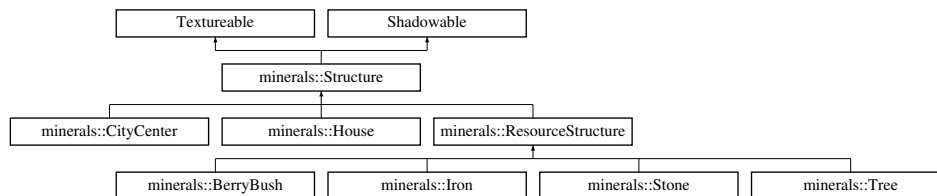
- [src/creatures/humans/Stonemason.h](#)
- [src/creatures/humans/Stonemason.cpp](#)

## 6.66. minerals::Structure osztályreferencia

A struktúra osztály leírása.

```
#include <Structure.h>
```

A minerals::Structure osztály származási diagramja:



### Publikus tagfüggvények

- [Structure](#) (int x, int y)  
*Létrehozza magát az x és y megadott pontban.*
- bool [setTexture](#) (const std::string &filename) override  
*Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- void [setPosition](#) (double x, double y) override  
*Beállítja, hogy hova kell rajzolni a textúrázható leszármozottat.*
- void [draw](#) (sf::RenderWindow &window) override  
*Kirajzolja az objektumot.*
- bool [needs\\_drawn](#) ()  
*Igazat ad vissza, ha látható és ezért ki kell rajzolni.*
- virtual [MINERAL\\_TYPE get\\_type](#) () const =0  
*Szimbólum, ami a fájlba mentéshez kell.*
- virtual void [update\\_logic](#) (float deltaTime)=0  
*Frissíti magát az idő függvényében.*
- void [draw\\_logic](#) (sf::RenderWindow &window, float elapsed\_time, int offx, int offy)  
*Kirajzolja a struktúrát attól függően, hogy ki kell-e.*
- virtual [~Structure](#) ()=default  
*Alap virtuális destruktork.*



## Publikus attribútumok

- int `posx`  
*Az X koordináta, amin elhelyezkedik.*
- int `posy`  
*Az Y koordináta, amin elhelyezkedik.*

## Védett attribútumok

- const int `MAX_OBJECT_SIZE` =64  
*Egy határ, minnél nagyobb, annál nagyobb környezetbe lesznek kirajzolva a nézőpontból.*

### 6.66.1. Részletes leírás

A struktúra osztály leírása.

A Textúrázható és Árnyékolható interface-ből öröklődik. Alaposztály amiből később jönnek a házak, erőforrások.

### 6.66.2. Konstruktorok és destruktorok dokumentációja

#### 6.66.2.1. Structure()

```
minerals::Structure::Structure (
    int x,
    int y )
```

Létrehozza magát az x és y megadott pontban.

#### Paraméterek

<code>x</code>	Az x koordináta.
<code>y</code>	Az y koordináta.

#### 6.66.2.2. ~Structure()

```
virtual minerals::Structure::~~Structure ( ) [virtual], [default]
```

Alap virtuális destruktor.

### 6.66.3. Tagfüggvények dokumentációja

#### 6.66.3.1. draw()

```
void minerals::Structure::draw (
    sf::RenderWindow & window ) [override], [virtual]
```

Kirajzolja az objektumot.

##### Paraméterek

<i>window</i>	Ahova ki kell rajzolni a textúrázható leszármazottat.
---------------	---

Megvalósítja a következőket: [Textureable](#).

#### 6.66.3.2. draw\_logic()

```
void minerals::Structure::draw_logic (
    sf::RenderWindow & window,
    float elapsed_time,
    int offx,
    int offy )
```

Kirajzolja a struktúrát attól függően, hogy ki kell-e.

##### Paraméterek

<i>window</i>	Az ablak, ahova rajzolni kell.
<i>elapsed_time</i>	A világ megléte óta eltelt idő.
<i>offx</i>	A kamera X eltolása.
<i>offy</i>	A kamera Y eltolása.

#### 6.66.3.3. get\_type()

```
virtual MINERAL_TYPE minerals::Structure::get_type ( ) const [pure virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítják a következők: [minerals::Tree](#), [minerals::Stone](#), [minerals::Iron](#), [minerals::House](#), [minerals::CityCenter](#) és [minerals::BerryBush](#).

#### 6.66.3.4. needs\_drawn()

```
bool minerals::Structure::needs_drawn ( )
```

Igazat ad vissza, ha látható és ezért ki kell rajzolni.

### 6.66.3.5. setPosition()

```
void minerals::Structure::setPosition (
    double x,
    double y ) [override], [virtual]
```

Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.

#### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

Megvalósítja a következőket: [Textureable](#).

### 6.66.3.6. setTexture()

```
bool minerals::Structure::setTexture (
    const std::string & filename ) [override], [virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

#### Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

#### Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítja a következőket: [Textureable](#).

### 6.66.3.7. update\_logic()

```
virtual void minerals::Structure::update_logic (
    float deltaTime ) [pure virtual]
```

Frissíti magát az idő függvényében.

#### Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítják a következők: [minerals::Tree](#), [minerals::Stone](#), [minerals::Iron](#), [minerals::House](#), [minerals::CityCenter](#) és [minerals::BerryBush](#).

## 6.66.4. Adattagok dokumentációja

### 6.66.4.1. MAX\_OBJECT\_SIZE

```
const int minerals::Structure::MAX_OBJECT_SIZE = 64 [protected]
```

Egy határ, minnél nagyobb, annál nagyobb környezetbe lesznek kirajzolva a nézőpontból.

### 6.66.4.2. posx

```
int minerals::Structure::posx
```

Az X koordináta, amin elhelyezkedik.

### 6.66.4.3. posy

```
int minerals::Structure::posy
```

Az Y koordináta, amin elhelyezkedik.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

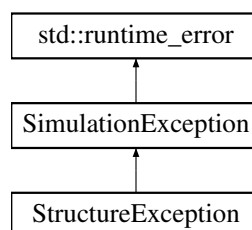
- [src/world\\_object/Structure.h](#)
- [src/world\\_object/Structure.cpp](#)

## 6.67. StructureException osztályreferencia

Akkor kell dobni, ha egy struktúra hibásan működött.

```
#include <WorldExceptions.h>
```

A StructureException osztály származási diagramja:



## Publikus tagfüggvények

- [StructureException](#) (const std::string &msg)

### 6.67.1. Részletes leírás

Akkor kell dobni, ha egy struktúra hibásan működött.

### 6.67.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.67.2.1. StructureException()

```
StructureException::StructureException (
    const std::string & msg ) [inline], [explicit]
```

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- src/exceptions/[WorldExceptions.h](#)

## 6.68. TerrainContainer< T > osztálysablon-referencia

A világ terepét tároló osztály.

```
#include <TerrainContainer.hpp>
```

## Publikus tagfüggvények

- int [get\\_width](#) () const  
*Egy getter a tömb szélességére.*
- int [get\\_height](#) () const  
*Egy getter a tömb magasságára.*
- [TerrainContainer](#) ()  
*Alapkonstruktor. Nem történik memória foglalás.*
- [TerrainContainer](#) (int awidth, int aheight, T def\_value)  
*Konstruktor, ami már foglal memóriát és generál terepet.*
- void [swap\\_at](#) (int x1, int y1, int x2, int y2)  
*Kicseréli a (x1,y1) kockát az (x2,y2) helyen lévő kockával.*
- bool [is\\_valid\\_coordinate](#) (int x, int y)  
*Visszaadja, hogy az (x,y) koordinátán lévő terepkocka helyesen van definiálva-e.*
- bool [is\\_on\\_screen](#) (int x, int y)  
*Visszaadja, hogy az (x,y) koordinátán lévő terepkocka rajta van-e a látható síkon.*
- T \*& [operator\[\]](#) (std::size\_t row)  
*operator[] hogy elérhetőek legyenek a belső elemek.*
- T \*const & [operator\[\]](#) (std::size\_t row) const

- konstans operator[] hogy elérhetőek legyenek a belső elemek.*
- void `draw` (`sf::RenderWindow` &window, int offx, int offy)  
*Kirajzolja az (x,y) koordinátán lévő terepkockát.*
- void `generate_world` ()  
*Generál egy új terepet.*
- void `clear_at` (int x, int y)  
*Felszabadítja az adott sorban és oszlopban elhelyezkedő terepkockát.*
- void `clear` ()  
*Felszabadítja a tárolt terepkockákat.*
- void `set_seed` (int new\_value)  
*Setter, beállítja a seedet az új megadott értékre.*
- int `get_seed` () const  
*Getter, visszaadja a seedet.*
- template<typename Defvalue >  
void `resize` (int awidth, int aheight)  
*Újraméretezi a terepet az új méreteir és feltölti a megadott típusú adattal.*
- ~`TerrainContainer` ()  
*A destruktork, ami kitörli az összes tárolt terepkockát a `clear()` meghívásával.*

## Publikus attribútumok

- const int `TILE_SIZE` =32  
*Egy terep maximális textúra mérete. Ennél nagyobb terepkockák talán nem rajzolódnak ki, mert a kamera úgy érzékeli, hogy már nincs benne a látótérben.*

### 6.68.1. Részletes leírás

```
template<typename T>
class TerrainContainer< T >
```

A világ terepét tároló osztály.

Sablon paraméterek

<code>T</code>	A generikus elem, amiket eltárol ez a konténer.
----------------	---

Egy dinamikus 2 dimenziós n\*m-es tömb. Rendelkezik a szükséges getterekkel. Ez az osztály felelős a világ terepének a véletlen generálásáért is.

### 6.68.2. Konstruktork és destruktork dokumentációja

#### 6.68.2.1. TerrainContainer() [1/2]

```
template<typename T >
TerrainContainer< T >::TerrainContainer ( ) [default]
```

Alapkonstruktor. Nem történik memória foglalás.

### 6.68.2.2. TerrainContainer() [2/2]

```
template<typename T >
TerrainContainer< T >::TerrainContainer (
    int awidth,
    int aheight,
    T def_value )
```

Konstruktor, ami már foglal memóriát és generál terepet.

#### Paraméterek

<i>awidth</i>	Az új szélesség.
<i>ahheight</i>	Az új magasság.
<i>def_value</i>	Alap érték, amivel feltöltődik a tömb.

### 6.68.2.3. ~TerrainContainer()

```
template<typename T >
TerrainContainer< T >::~~TerrainContainer
```

A destruktork, ami kitörli az összes tárolt terepkockát a [clear\(\)](#) meghívásával.

## 6.68.3. Tagfüggvények dokumentációja

### 6.68.3.1. clear()

```
template<typename T >
void TerrainContainer< T >::clear
```

Felszabadítja a tárolt terepkockákat.

### 6.68.3.2. clear\_at()

```
template<typename T >
void TerrainContainer< T >::clear_at (
    int x,
    int y )
```

Felszabadítja az adott sorban és oszlopban elhelyezkedő terepkockát.

## Paraméterek

<i>x</i>	Oszlop index.
<i>y</i>	Sor index.

**6.68.3.3. draw()**

```
template<typename T >
void TerrainContainer< T >::draw (
    sf::RenderWindow & window,
    int offx,
    int offy )
```

Kirajzolja az (x,y) koordinátán lévő terepkockát.

## Paraméterek

<i>window</i>	Ahova ki kell rajzolni a terepkockát.
<i>x</i>	Oszlop index.
<i>y</i>	Sor index.

**6.68.3.4. generate\_world()**

```
template<typename T >
void TerrainContainer< T >::generate_world
```

Generál egy új terepet.

**6.68.3.5. get\_height()**

```
template<typename T >
int TerrainContainer< T >::get_height
```

Egy getter a tömb magasságára.

## Visszatérési érték

A tömb magassága.



#### 6.68.3.6. get\_seed()

```
template<typename T >
int TerrainContainer< T >::get_seed
```

Getter, visszaadja a seedet.

##### Visszatérési érték

A seed.

#### 6.68.3.7. get\_width()

```
template<typename T >
int TerrainContainer< T >::get_width
```

Egy getter a tömb szélességére.

##### Visszatérési érték

A tömb szélessége.

#### 6.68.3.8. is\_on\_screen()

```
template<typename T >
bool TerrainContainer< T >::is_on_screen (
    int x,
    int y )
```

Visszaadja, hogy az (x,y) koordinátán lévő terepkocka rajta van-e a látható síkon.

##### Paraméterek

x	Oszlop index.
y	Sor index.

##### Visszatérési érték

Igaz, rajta van, különben hamis.

#### 6.68.3.9. is\_valid\_coordinate()

```
template<typename T >
bool TerrainContainer< T >::is_valid_coordinate (
```

```
int x,
int y )
```

Visszaadja, hogy az (x,y) koordinátán lévő terepkocka helyesen van definiálva-e.

#### Paraméterek

<i>x</i>	Oszlop index.
<i>y</i>	Sor index.

#### Visszatérési érték

Igaz, ha rendesen van definiálva és már használható, különben hamis.

#### 6.68.3.10. operator[]() [1/2]

```
template<typename T >
T *& TerrainContainer< T >::operator[] (
    std::size_t row )
```

operator[] hogy elérhetőek legyenek a belső elemek.

#### 6.68.3.11. operator[]() [2/2]

```
template<typename T >
T *const & TerrainContainer< T >::operator[] (
    std::size_t row ) const
```

konstans operator[] hogy elérhetőek legyenek a belső elemek.

#### 6.68.3.12. resize()

```
template<typename T >
template<typename Defvalue >
void TerrainContainer< T >::resize (
    int awidth,
    int aheight )
```

Újraméretezi a terepet az új méreteir és feltölti a megadott típusú adattal.

#### Sablon paraméterek

<i>Defvalue</i>	Az érték amivel az új terep fel lesz töltve.
-----------------	--

## Paraméterek

<i>awidth</i>	Az új érték.
<i>aheight</i>	Az új érték.

**6.68.3.13. set\_seed()**

```
template<typename T >
void TerrainContainer< T >::set_seed (
    int new_value )
```

Setter, beállítja a seedet az új megadott értékre.

## Paraméterek

<i>new_value</i>	Az új érték.
------------------	--------------

**6.68.3.14. swap\_at()**

```
template<typename T >
void TerrainContainer< T >::swap_at (
    int x1,
    int y1,
    int x2,
    int y2 )
```

Kicseréli a (x1,y1) kockát az (x2,y2) helyen lévő kockával.

## Paraméterek

<i>x1</i>	Az 1. kocka x koordinátája.
<i>y1</i>	Az 1. kocka y koordinátája.
<i>x2</i>	A 2. kocka x koordinátája.
<i>y2</i>	A 2. kocka y koordinátája.

**6.68.4. Adattagok dokumentációja****6.68.4.1. TILE\_SIZE**

```
template<typename T >
const int TerrainContainer< T >::TILE_SIZE =32
```

Egy terep maximális textúra mérete. Ennél nagyobb terepkockák talán nem rajzolódnak ki, mert a kamera úgy érzékeli, hogy már nincs benne a látótérben.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/[TerrainContainer.hpp](#)
- src/[TerrainContainer\\_def.hpp](#)

## 6.69. gtest\_lite::Test struktúrareferencia

```
#include <modified_gtest_lite.h>
```

### Publikus tagfüggvények

- void [begin](#) (const char \*n)  
*Teszt kezdete.*
- std::ostream & [end](#) (bool memchk=false)  
*Teszt vége.*
- bool [fail](#) ()
- bool [astatus](#) ()
- std::ostream & [expect](#) (bool st, const char \*file, int line, const char \*expr, bool pr=false)  
*Eredményt adminisztráló tagfüggvény True a jó eset.*
- [~Test](#) ()  
*Destruktor.*

### Statikus publikus tagfüggvények

- static [Test](#) & [getTest](#) ()

### Publikus attribútumok

- int [sum](#)  
*tesztek számlálója*
- int [failed](#)  
*hibás tesztek*
- int [ablocks](#)  
*allokált blokkok száma*
- bool [status](#)  
*éppen futó teszt státusza.*
- bool [tmp](#)  
*temp a kivételkezeléshez;*
- std::string [name](#)  
*éppen futó teszt neve.*
- std::fstream [null](#)  
*nyelő, ha nem kell kiírni semmit*
- std::ostream & [os](#)  
*ide írunk*

### 6.69.1. Részletes leírás

Tesztek állapotát tároló osztály. Egyetlen egy statikus példány keletkezik, aminek a destruktora a futás végén hívódik meg.

### 6.69.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.69.2.1. ~Test()

```
gtest_lite::Test::~~Test ( ) [inline]
```

Destruktor.

### 6.69.3. Tagfüggvények dokumentációja

#### 6.69.3.1. astatus()

```
bool gtest_lite::Test::astatus ( ) [inline]
```

#### 6.69.3.2. begin()

```
void gtest_lite::Test::begin (
    const char * n ) [inline]
```

Teszt kezdete.

#### 6.69.3.3. end()

```
std::ostream& gtest_lite::Test::end (
    bool memchk = false ) [inline]
```

Teszt vége.

#### 6.69.3.4. expect()

```
std::ostream& gtest_lite::Test::expect (
    bool st,
    const char * file,
    int line,
    const char * expr,
    bool pr = false ) [inline]
```

Eredményt adminisztráló tagfüggvény True a jó eset.

#### 6.69.3.5. fail()

```
bool gtest_lite::Test::fail ( ) [inline]
```

#### 6.69.3.6. getTest()

```
static Test& gtest_lite::Test::getTest ( ) [inline], [static]
```

< egyedüli (singleton) példány

### 6.69.4. Adattagok dokumentációja

#### 6.69.4.1. ablocks

```
int gtest_lite::Test::ablocks
```

allokált blokkok száma

#### 6.69.4.2. failed

```
int gtest_lite::Test::failed
```

hibás tesztek

#### 6.69.4.3. name

```
std::string gtest_lite::Test::name
```

éppen futó teszt neve.

#### 6.69.4.4. null

```
std::fstream gtest_lite::Test::null
```

nyelő, ha nem kell kiírni semmit

#### 6.69.4.5. os

```
std::ostream& gtest_lite::Test::os
```

ide írunk

#### 6.69.4.6. status

```
bool gtest_lite::Test::status
```

éppen futó teszt státusza.

#### 6.69.4.7. sum

```
int gtest_lite::Test::sum
```

tesztek számlálója

#### 6.69.4.8. tmp

```
bool gtest_lite::Test::tmp
```

temp a kivételkezeléshez;

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- src/external/[modified\\_gtest\\_lite.h](#)

## 6.70. sf::Texture osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [Texture](#) ()
- bool [loadFromFile](#) (const std::string &filepath)
- [Vector2i](#) [getSize](#) ()
- [~Texture](#) ()
- [Texture](#) (const [Texture](#) &other)
- [Texture](#) & [operator=](#) (const [Texture](#) &other)

### 6.70.1. Konstruktorkok és destruktorok dokumentációja

#### 6.70.1.1. Texture() [1/2]

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

#### 6.70.1.2. ~Texture()

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

#### 6.70.1.3. Texture() [2/2]

```
sf::Texture::Texture (
    const Texture & other )
```

### 6.70.2. Tagfüggvények dokumentációja

#### 6.70.2.1. getSize()

```
Vector2i sf::Texture::getSize ( )
```



### 6.70.2.2. loadFromFile()

```
bool sf::Texture::loadFromFile (
    const std::string & filepath )
```

### 6.70.2.3. operator=()

```
Texture & sf::Texture::operator= (
    const Texture & other )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

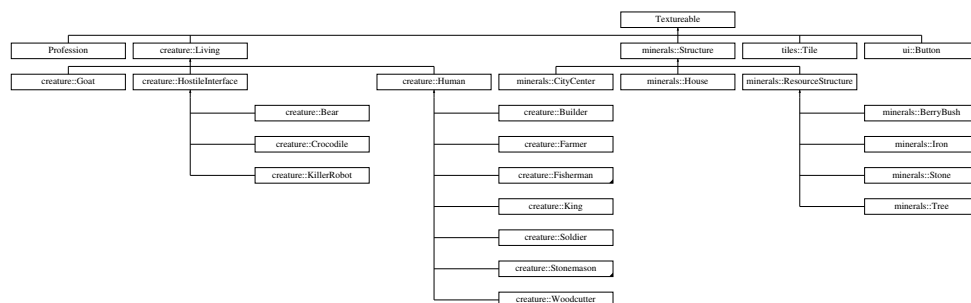
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.71. Textureable osztályreferencia

Egy interface, ami a textúrázáshoz kell.

```
#include <Textureable.h>
```

A Textureable osztály származási diagramja:



## Publikus tagfüggvények

- virtual `~Textureable()`=default  
*Alap virtuális destruktork.*
- virtual bool `setTexture(const std::string &filename)=0`  
*Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- virtual void `setPosition(double x, double y)=0`  
*Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.*
- virtual void `draw(sf::RenderWindow &window)=0`  
*Kirajzolja az objektumot.*

### 6.71.1. Részletes leírás

Egy interface, ami a textúrázáshoz kell.

Jelzi, hogy a kirajzoláshoz és a világban való megjelenítéséhez milyen metódusokat kell elkészíteni.

### 6.71.2. Konstruktorok és destruktorok dokumentációja

#### 6.71.2.1. ~Textureable()

```
virtual Textureable::~Textureable ( ) [virtual], [default]
```

Alap virtuális destruktor.

### 6.71.3. Tagfüggvények dokumentációja

#### 6.71.3.1. draw()

```
virtual void Textureable::draw (
    sf::RenderWindow & window ) [pure virtual]
```

Kirajzolja az objektumot.

##### Paraméterek

<i>window</i>	Ahova ki kell rajzolni a textúrázható leszármazottat.
---------------	---

Megvalósítják a következők: [minerals::Structure](#), [ui::Button](#), [tiles::Tile](#), [Profession](#) és [creature::Living](#).

#### 6.71.3.2. setPosition()

```
virtual void Textureable::setPosition (
    double x,
    double y ) [pure virtual]
```

Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.

##### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

Megvalósítják a következők: [minerals::Structure](#), [ui::Button](#), [tiles::Tile](#), [Profession](#) és [creature::Living](#).

### 6.71.3.3. setTexture()

```
virtual bool Textureable::setTexture (
    const std::string & filename ) [pure virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

#### Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

#### Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítják a következők: [minerals::Structure](#), [ui::Button](#), [tiles::Tile](#), [Profession](#) és [creature::Living](#).

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [src/Textureable.h](#)

## 6.72. TextureManager osztályreferencia

A Textúra kezelő osztály.

```
#include <TextureManager.h>
```

### Publikus tagfüggvények

- [sf::Texture](#) \* [loadTexture](#) (const std::string &filename)  
*Betölti a kért textúrát, ha kell. Ha nem, akkor csak visszaadja a már régen betöltött textúrát.*
- [sf::Texture](#) [getTexture](#) (const std::string &filename)  
*Odaadja a kért textúrát. Azért kell, mert a Memtrace hibát dob, ha a másik metódust használom.*
- void [clear](#) ()  
*Kitörli az összes betöltött textúrát.*

### Statikus publikus tagfüggvények

- static [TextureManager](#) & [getInstance](#) ()  
*Odaadja a referenciát a singleton-ra.*

### 6.72.1. Részletes leírás

A Textúra kezelő osztály.

A textúrák betöltéséért, tárolásáért és kiosztásáért felelős osztály. Rendelkezik egy tisztítás metódussal is.

### 6.72.2. Tagfüggvények dokumentációja

#### 6.72.2.1. clear()

```
void TextureManager::clear ( )
```

Kitörli az összes betöltött textúrát.

#### 6.72.2.2. getInstance()

```
TextureManager & TextureManager::getInstance ( ) [static]
```

Odaadja a referenciát a singleton-ra.

Visszatérési érték

A referencia a textúramezőre.

#### 6.72.2.3. getTexture()

```
sf::Texture TextureManager::getTexture (
    const std::string & filename )
```

Odaadja a kért textúrát. Azért kell, mert a Memtrace hibát dob, ha a másik metódust használom.

Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

Visszatérési érték

Referencia a kért textúrára.

## 6.72.2.4. loadTexture()

```
sf::Texture * TextureManager::loadTexture (
    const std::string & filename )
```

Betölti a kért textúrát, ha kell. Ha nem, akkor csak visszaadja a már régen betöltött textúrát.

## Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

## Visszatérési érték

Referencia a kért textúrára.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

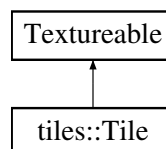
- [src/TextureManager.h](#)
- [src/TextureManager.cpp](#)

## 6.73. tiles::Tile osztályreferencia

A terepkocka osztály leírása.

```
#include <Tile.h>
```

A tiles::Tile osztály származási diagramja:



## Publikus tagfüggvények

- void [init](#) ([TILETYPE](#) newtype)  
*Inicializálja a terepkocka kinézetét a biotípusa alapján.*
- [TILETYPE get\\_type](#) () const  
*Egy getter ami visszaadja a terepkocka biotípusát.*
- bool [setTexture](#) (const std::string &filename) override  
*Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.*
- void [setPosition](#) (double x, double y) override  
*Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.*
- void [draw](#) ([sf::RenderWindow](#) &window) override  
*Kirajzolja az objektumot.*

### 6.73.1. Részletes leírás

A terepkocka osztály leírása.

Tárolja a terepkocka kinézetét és azt, hogy milyen biom típusú.

### 6.73.2. Tagfüggvények dokumentációja

#### 6.73.2.1. draw()

```
void tiles::Tile::draw (
    sf::RenderWindow & window ) [override], [virtual]
```

Kirajzolja az objektumot.

##### Paraméterek

<i>window</i>	Ahova ki kell rajzolni a textúrázható leszármozottat.
---------------	---

Megvalósítja a következőket: [Textureable](#).

#### 6.73.2.2. get\_type()

```
TILETYPE tiles::Tile::get_type ( ) const
```

Egy getter ami visszaadja a terepkocka biotípusát.

##### Visszatérési érték

A biotípusa.

#### 6.73.2.3. init()

```
void tiles::Tile::init (
    TILETYPE newtype )
```

Inicializálja a terepkocka kinézetét a biotípusa alapján.

##### Paraméterek

<i>newtype</i>	A biotípusa.
----------------	--------------

#### 6.73.2.4. setPosition()

```
void tiles::Tile::setPosition (
    double x,
    double y ) [override], [virtual]
```

Beállítja, hogy hova kell rajzolni a textúrázható leszármazottat.

##### Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

Megvalósítja a következőket: [Textureable](#).

#### 6.73.2.5. setTexture()

```
bool tiles::Tile::setTexture (
    const std::string & filename ) [override], [virtual]
```

Beállít egy fájl elérési útból egy textúrát. Megvalósítástól függően esetleg 2-t.

##### Paraméterek

<i>filename</i>	A textúra elérési útja.
-----------------	-------------------------

##### Visszatérési érték

Sikeres volt-e a textúra beállítása.

Megvalósítja a következőket: [Textureable](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/terrain\_tiles/[Tile.h](#)
- src/terrain\_tiles/[Tile.cpp](#)

## 6.74. sf::Transform osztályreferencia

```
#include <fake_sfml.h>
```

## Publikus tagfüggvények

- [Transform](#) ()
- [Transform](#) (float a00, float a01, float a02, float a10, float a11, float a12, float a20, float a21, float a22)
- [Transform combine](#) (const [Transform](#) &other)
- void [transformPoint](#) (float x, float y) const
- void [translate](#) (float tx, float ty)
- void [translate](#) ([Vector2f](#) Vy)

## Publikus attribútumok

- float [matrix](#) [9]

## 6.74.1. Konstruktorkok és destruktorkok dokumentációja

### 6.74.1.1. Transform() [1/2]

```
sf::Transform::Transform ( )
```

### 6.74.1.2. Transform() [2/2]

```
sf::Transform::Transform (
    float a00,
    float a01,
    float a02,
    float a10,
    float a11,
    float a12,
    float a20,
    float a21,
    float a22 )
```

## 6.74.2. Tagfüggvények dokumentációja

### 6.74.2.1. combine()

```
Transform sf::Transform::combine (
    const Transform & other )
```



### 6.74.2.2. transformPoint()

```
void sf::Transform::transformPoint (
    float x,
    float y ) const
```

### 6.74.2.3. translate() [1/2]

```
void sf::Transform::translate (
    float tx,
    float ty )
```

### 6.74.2.4. translate() [2/2]

```
void sf::Transform::translate (
    Vector2f Vy )
```

## 6.74.3. Adattagok dokumentációja

### 6.74.3.1. matrix

```
float sf::Transform::matrix[9]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

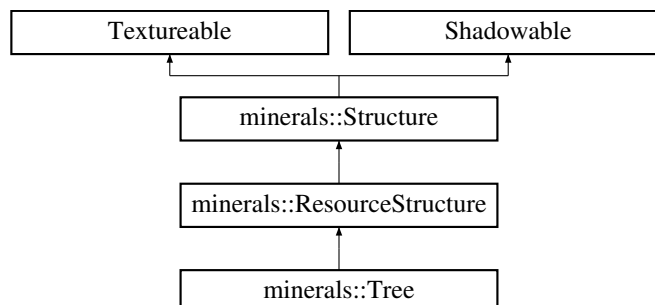
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.75. minerals::Tree osztályreferencia

A fa osztály leírása. Fát ad, ha kitermelik.

```
#include <Tree.h>
```

A minerals::Tree osztály származási diagramja:



## Publikus tagfüggvények

- [Tree](#) (int x, int y)  
*Konstruktor ami lerakja az erőforrást egy (x,y) pontra.*
- [MINERAL\\_TYPE get\\_type](#) () const override  
*Szimbólum, ami a fájlba mentéshez kell.*
- void [update\\_logic](#) (float deltaTime) override  
*Frissíti magát az idő függvényében.*
- void [play\\_destroy\\_sound](#) ([SoundPlayer](#) &sound\_player) const override

## További örökölt tagok

### 6.75.1. Részletes leírás

A fa osztály leírása. Fát ad, ha kitermelik.

### 6.75.2. Konstruktorok és destruktorok dokumentációja

#### 6.75.2.1. Tree()

```
minerals::Tree::Tree (  
    int x,  
    int y )
```

Konstruktor ami lerakja az erőforrást egy (x,y) pontra.

### 6.75.3. Tagfüggvények dokumentációja

#### 6.75.3.1. get\_type()

```
MINERAL\_TYPE minerals::Tree::get_type ( ) const [override], [virtual]
```

Szimbólum, ami a fájlba mentéshez kell.

Megvalósítja a következőket: [minerals::Structure](#).

#### 6.75.3.2. play\_destroy\_sound()

```
void minerals::Tree::play_destroy_sound (  
    SoundPlayer & sound_player ) const [override], [virtual]
```

Megvalósítja a következőket: [minerals::ResourceStructure](#).

#### 6.75.3.3. update\_logic()

```
void minerals::Tree::update_logic (  
    float deltaTime ) [override], [virtual]
```

Frissíti magát az idő függvényében.

## Paraméterek

<i>deltaTime</i>	Az előző frissítés óta eltelt idő.
------------------	------------------------------------

Megvalósítja a következőket: [minerals::Structure](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/world\_object/[Tree.h](#)
- src/world\_object/[Tree.cpp](#)

## 6.76. sf::Vector2f osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [Vector2f](#) ()
- [Vector2f](#) (float x1, float y1)

### Publikus attribútumok

- float [x](#)
- float [y](#)

### 6.76.1. Konstruktorkok és destruktorkok dokumentációja

#### 6.76.1.1. Vector2f() [1/2]

```
sf::Vector2f::Vector2f ( )
```

#### 6.76.1.2. Vector2f() [2/2]

```
sf::Vector2f::Vector2f (
    float x1,
    float y1 )
```

### 6.76.2. Adattagok dokumentációja

#### 6.76.2.1. x

```
float sf::Vector2f::x
```

#### 6.76.2.2. y

```
float sf::Vector2f::y
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

### 6.77. sf::Vector2i osztályreferencia

```
#include <fake_sfml.h>
```

#### Publikus tagfüggvények

- [Vector2i](#) ()
- [Vector2i](#) (int x1, int y1)

#### Publikus attribútumok

- int [x](#)
- int [y](#)

#### 6.77.1. Konstruktorkok és destruktorkok dokumentációja

##### 6.77.1.1. Vector2i() [1/2]

```
sf::Vector2i::Vector2i ( )
```

##### 6.77.1.2. Vector2i() [2/2]

```
sf::Vector2i::Vector2i (
    int x1,
    int y1 )
```

## 6.77.2. Adattagok dokumentációja

### 6.77.2.1. x

```
int sf::Vector2i::x
```

### 6.77.2.2. y

```
int sf::Vector2i::y
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.78. sf::VideoMode osztályreferencia

```
#include <fake_sfml.h>
```

### Publikus tagfüggvények

- [VideoMode](#) (std::size\_t w=800, std::size\_t h=600)
- bool [isValid](#) () const

### Statikus publikus tagfüggvények

- static [VideoMode](#) [getDesktopMode](#) ()

### Publikus attribútumok

- std::size\_t [width](#)
- std::size\_t [height](#)
- std::size\_t [bitsPerPixel](#)

## 6.78.1. Konstruktorok és destruktorok dokumentációja

### 6.78.1.1. VideoMode()

```
sf::VideoMode::VideoMode (
    std::size_t w = 800,
    std::size_t h = 600 )
```

## 6.78.2. Tagfüggvények dokumentációja

### 6.78.2.1. getDesktopMode()

```
VideoMode sf::VideoMode::getDesktopMode ( ) [static]
```

< (HD)

### 6.78.2.2. isValid()

```
bool sf::VideoMode::isValid ( ) const
```

## 6.78.3. Adattagok dokumentációja

### 6.78.3.1. bitsPerPixel

```
std::size_t sf::VideoMode::bitsPerPixel
```

### 6.78.3.2. height

```
std::size_t sf::VideoMode::height
```

### 6.78.3.3. width

```
std::size_t sf::VideoMode::width
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

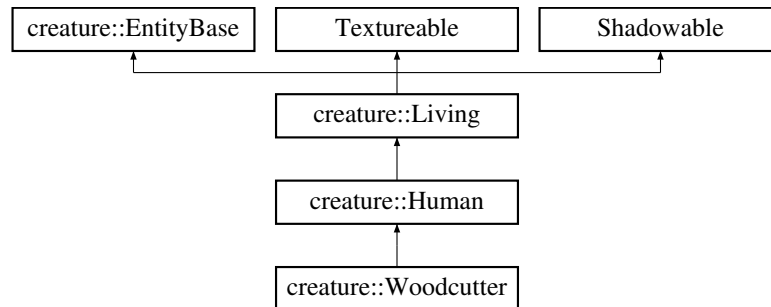
- [src/fake\\_sfml/fake\\_sfml.h](#)
- [src/fake\\_sfml/fake\\_sfml.cpp](#)

## 6.79. creature::Woodcutter osztályreferencia

A favágó szakmájú ember osztály leírása.

```
#include <Woodcutter.h>
```

A creature::Woodcutter osztály származási diagramja:



### Publikus tagfüggvények

- **Woodcutter** (int x, int y, ENTITY\_GENDER gender\_modifier)  
*Inicializál egy favágót egy pontos x és y koordinátára és beállítja az attribútumait.*
- void **update\_logic** (World &world, float deltaTime) override  
*Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.*
- **~Woodcutter** ()  
*A favágó destruktora.*

### További örökölt tagok

#### 6.79.1. Részletes leírás

A favágó szakmájú ember osztály leírása.

Ez a szakmájú ember fákat keres és kivágja őket, így fát szerez.

#### 6.79.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.79.2.1. Woodcutter()

```
creature::Woodcutter::Woodcutter (
    int x,
    int y,
    ENTITY_GENDER gender_modifier )
```

Inicializál egy favágót egy pontos x és y koordinátára és beállítja az attribútumait.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	A favágó neme.

**6.79.2.2. ~Woodcutter()**

```
creature::Woodcutter::~~Woodcutter ( )
```

A favágó destruktora.

**6.79.3. Tagfüggvények dokumentációja****6.79.3.1. update\_logic()**

```
void creature::Woodcutter::update_logic (
    World & world,
    float deltaTime ) [override], [virtual]
```

Az entitás frissítési logikája itt van definiálva. Például a halász vizet keres, ha odaért akkor halászik.

## Paraméterek

<i>world</i>	A világ, amibe frissíti magát az entitás.
<i>deltaTime</i>	Az előző frissítés óta eltelt idő.

< Még mindig nincs?

Újraimplementált ősök: [creature::Human](#).

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/creatures/humans/Woodcutter.h](#)
- [src/creatures/humans/Woodcutter.cpp](#)

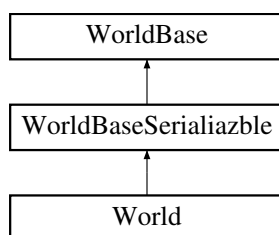
**6.80. World osztályreferencia**

A világ osztály leírása.

```
#include <World.hpp>
```



A World osztály származási diagramja:



## Publikus tagfüggvények

- int `get_border_width` () const  
*Egy getter a horizontális kamera határ nagyságához.*
- int `get_border_height` () const  
*Egy getter a vertikális kamera határ nagyságához.*
- void `set_border_width` (int newwidth)  
*Egy setter a horizontális kamera határ nagyságához.*
- void `set_border_height` (int newheight)  
*Egy setter a vertikális kamera határ nagyságához.*
- void `clear` () override  
*Segédfüggvény a világ törléséhez. Felszabadítja az entitásokat, erőforrásokat.*
- `World` ()  
*A világ konstruktora. A konstruktor generál egy alap világot.*
- `~World` ()  
*A világ destruktora. Itt felszabadul minden, ami a világban "van". Emberek, állatok, város, erőforrások.*
- void `draw` (sf::RenderWindow &>window, float delta\_time, int offx, int offy)  
*Kirajzol mindent, ami a világba van.*
- void `update_world` (float delta\_time)  
*Frissíti a világban lévő entitásokat, napszakot, árnyékolást.*
- void `regenerate` ()  
*Újra épít egy világot az előző világ helyére. Az előző világ tartalmát üríti.*
- void `populate_world` ()  
*Idéz entításokat és fákat, bokrokat a világba. Ha pehhesek az emberek akkor egy gyilkos robot is megjelenik :(.*
- void `try_develop_random_role` (creature::Human \*&human\_ptr)  
*Kiválaszt egy új szakmát egy embernek. Ha ez az ember építette a várost, király lesz belőle.*
- bool `spawn_entity_at_pos` (creature::Living \*entity)  
*Berak egy entitás pointert már megadott pozícióval a világba.*
- bool `spawn_human` (creature::Human \*human)  
*Berak egy ember pointert már megadott pozícióval a világba.*

## További örökölt tagok

### 6.80.1. Részletes leírás

A világ osztály leírása.

Tárolja az erőforrásokat, embereket, entításokat, a terepet. Rendelkezik a szimulációhoz tartozó metódusokkal. És kirajzolással is.

## 6.80.2. Konstruktorkok és destruktorkok dokumentációja

### 6.80.2.1. World()

```
World::World ( )
```

A világ konstruktora. A konstruktor generál egy alap világot.

### 6.80.2.2. ~World()

```
World::~~World ( )
```

A világ destruktora. Itt felszabadul minden, ami a világban "van". Emberek, állatok, város, erőforrások.

## 6.80.3. Tagfüggvények dokumentációja

### 6.80.3.1. clear()

```
void World::clear ( ) [override], [virtual]
```

Segédfüggvény a világ törléséhez. Felszabadítja az entitásokat, erőforrásokat.

Megvalósítja a következőket: [WorldBaseSerializable](#).

### 6.80.3.2. draw()

```
void World::draw (
    sf::RenderWindow & window,
    float delta_time,
    int offx,
    int offy )
```

Kirajzol mindent, ami a világba van.

#### Paraméterek

<i>window</i>	A játéklablak.
<i>delta_time</i>	Az előző frissítés óta eltelt idő.
<i>offx</i>	A kamera elmozdulása horizontálisan a felső bal csücsökhöz képest.
<i>offy</i>	A kamera elmozdulása vertikálisan a felső bal csücsökhöz képest.

**6.80.3.3. get\_border\_height()**

```
int World::get_border_height ( ) const
```

Egy getter a vertikális kamera határ nagyságához.

**Visszatérési érték**

A határ mérete.

**6.80.3.4. get\_border\_width()**

```
int World::get_border_width ( ) const
```

Egy getter a horizontális kamera határ nagyságához.

**Visszatérési érték**

A határ mérete.

**6.80.3.5. populate\_world()**

```
void World::populate_world ( )
```

Idéz entitásokat és fákat, bokrokat a világba. Ha pehhesek az emberek akkor egy gyilkos robot is megjelenik :(.

**6.80.3.6. regenerate()**

```
void World::regenerate ( )
```

Újra épít egy világot az előző világ helyére. Az előző világ tartalmát üríti.

**6.80.3.7. set\_border\_height()**

```
void World::set_border_height (
    int newheight )
```

Egy setter a vertikális kamera határ nagyságához.

## Paraméterek

<i>newheight</i>	Az új határ méret.
------------------	--------------------

**6.80.3.8. set\_border\_width()**

```
void World::set_border_width (
    int newwidth )
```

Egy setter a horizontális kamera határ nagyságához.

## Paraméterek

<i>newwidth</i>	Az új határ méret.
-----------------	--------------------

**6.80.3.9. spawn\_entity\_at\_pos()**

```
bool World::spawn_entity_at_pos (
    creature::Living * entity )
```

Berak egy entitás pointert már megadott pozícióval a világba.

**6.80.3.10. spawn\_human()**

```
bool World::spawn_human (
    creature::Human * human )
```

Berak egy ember pointert már megadott pozícióval a világba.

**6.80.3.11. try\_develop\_random\_role()**

```
void World::try_develop_random_role (
    creature::Human *& human_ptr )
```

Kiválaszt egy új szakmát egy embernek. Ha ez az ember építette a várost, király lesz belőle.

## Paraméterek

<i>human_ptr</i>	Az ember pointer referenciája, akinek új szakmát kell adni.
------------------	---

### 6.80.3.12. update\_world()

```
void World::update_world (
    float delta_time )
```

Frissíti a világban lévő entitásokat, napszakot, árnyékolást.

#### Paraméterek

<code>delta_time</code>	Az előző frissítés óta eltelt idő.
-------------------------	------------------------------------

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

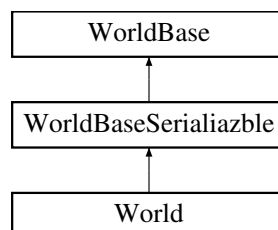
- [src/World.hpp](#)
- [src/World.cpp](#)

## 6.81. WorldBase osztályreferencia

A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza.

```
#include <World.hpp>
```

A WorldBase osztály származási diagramja:



### Publikus tagfüggvények

- [HumanResources](#) & [get\\_resources](#) ()  
*Külső referenciát ad a tárolt erőforrások elérésére.*
- [minerals::CityCenter](#) \* [get\\_current\\_city\\_center](#) ()  
*Egy függvény, amivel kívülről el lehet érni a jelenlegi városközpontot.*
- [tiles::Tile](#) & [getTileAt](#) (int x, int y) const  
*Visszaadja a világban az [y][x]-edik terepkockát.*
- `template<typename T >`  
`void spawn\_structure (bool mountain_exclusive)`  
*Idéz egy struktúrát. Opcionálisan bekapcsolható, hogy csak a hegyekben idéződjön.*
- [minerals::Structure](#) \* [get\\_structure\\_type](#) ([minerals::MINERAL\\_TYPE](#) atype)  
*Keres egy olyan erőforrás struktúrát, ami bizonyos erőforrást tartalmaz.*
- `void remove\_structure\_at (int x, int y)`

- Lerombol egy struktúrát egy bizonyos x és y koordinátán. Ezt az emberek hívják meg bányászatkor, favágáskor. Ha házra vagy városközpontra hívódik meg akkor hiba keletkezik.*
- `sf::Vector2f get_position_nearby_town ()`  
*Keres egy nem foglalt lakóterületnek való helyet a városközpontához közel.*
- `sf::Vector2f get_random_house_pos ()`  
*Keres egy olyan (x,y) koordinátát, amin van ház.*
- `void upgrade_house_at (int x, int y)`  
*Megpróbál megfrissíteni egy házat az (x,y) koordinátán. Ezt az építész ember hívja meg.*
- `creature::Living * get_excluded_entities (creature::ENTITY_TYPE excluded_type)`  
*Visszaad egy olyan entitás típust, ami nem a specifikált típus. Ezt a ragadozó állatok hívják meg, hogy ne egymást vadásszák. Az ember vadász sem öl embereket.*
- `template<typename T >`  
`void spawn_structure_at (int x, int y)`  
*Idéz egy struktúrát egy pontos x és y koordinátára.*
- `template<typename T >`  
`void spawn_entity (tiles::TILETYPE goal_habitat, const std::string &savefile_identifier)`  
*Idéz egy entitást, egy bizonyos típusú biomba.*
- `sf::Vector2f get_random_suitable_position (tiles::TILETYPE suitable_tile)`  
*Keres egy optimális helyet terepkocka típus szerint.*
- `void build_city_center_at (int x, int y)`  
*Épít egy városközpontot egy x és y koordinátára. Ha már létezik városközpont, hiba keletkezik.*
- `virtual ~WorldBase ()=default`  
*Virtuális destruktork.*

## Védett attribútumok

- `TerrainContainer< tiles::Tile * > terrain`  
*A terep tároló, 2 dimenziós dinamikus tömb.*
- `std::vector< creature::Living * > entities`  
*Az entitások pointerének tárolása. Nem tárol embert.*
- `std::vector< creature::Human * > humans`  
*Az ember pointerek tárolása.*
- `minerals::CityCenter * current_city_center`  
*A városközpont pointerének tárolása. Csak 1 városközpont lehet, ha 2-t próbálnak építeni, az hibás működést jelent.*
- `std::vector< minerals::ResourceStructure * > structures`  
*Az erőforrást tartalmazó struktúrák pointerének (fa, bókör, kő, vasérc) tárolásáért felelős heterogén kollekció.*
- `std::vector< minerals::House * > houses`  
*Az emberek által épített lakóházak pointerének tárolásáért felelős tároló.*
- `bool camp_needs_spawn`  
*Kell-e idéznünk új városlakókat?*
- `SoundPlayer sound_player`  
*Hanglejátszó modul. Mindennek hangja van a világban, ezért kell ez az osztály.*

## Statikus védett attribútumok

- `static constexpr int MAX_OBJECT_SIZE =64`  
*Amikor kikerül valami ennyire messziről a látótérből akkor az már nem rajzolódik ki.*

### 6.81.1. Részletes leírás

A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza.

### 6.81.2. Konstruktorok és destruktorok dokumentációja

#### 6.81.2.1. ~WorldBase()

```
virtual WorldBase::~~WorldBase ( ) [virtual], [default]
```

Virtuális destruktor.

### 6.81.3. Tagfüggvények dokumentációja

#### 6.81.3.1. build\_city\_center\_at()

```
void WorldBase::build_city_center_at (
    int x,
    int y )
```

Épít egy városközpontot egy x és y koordinátára. Ha már létezik városközpont, hiba keletkezik.

##### Paraméterek

x	Az x koordináta.
y	Az y koordináta.

#### 6.81.3.2. get\_current\_city\_center()

```
minerals::CityCenter * WorldBase::get_current_city_center ( )
```

Egy függvény, amivel kívülről el lehet érni a jelenlegi városközpontot.

##### Visszatérési érték

A városközpont.

### 6.81.3.3. `get_excluded_entities()`

```
creature::Living * WorldBase::get_excluded_entities (
    creature::ENTITY_TYPE excluded_type )
```

Visszaad egy olyan entitás típust, ami nem a specifikált típus. Ezt a ragadozó állatok hívják meg, hogy ne egymást vadásszák. Az ember vadász sem öl embereket.

#### Paraméterek

<code>excluded_type</code>	A típus amit nem akar megkapni.
----------------------------	---------------------------------

#### Visszatérési érték

Egy entitás pointer.

### 6.81.3.4. `get_position_nearby_town()`

```
sf::Vector2f WorldBase::get_position_nearby_town ( )
```

Keres egy nem foglalt lakóterületnek való helyet a városközpontához közel.

#### Visszatérési érték

Egy (x,y) koordináta, ahova lehet házat építeni. Ha nincs ilyen akkor (-1,-1).

### 6.81.3.5. `get_random_house_pos()`

```
sf::Vector2f WorldBase::get_random_house_pos ( )
```

Keres egy olyan (x,y) koordinátát, amin van ház.

#### Visszatérési érték

Egy (x,y) koordinátpár ahol van ház. (-1,-1) ha nincs ilyen.

### 6.81.3.6. `get_random_suitable_position()`

```
sf::Vector2f WorldBase::get_random_suitable_position (
    tiles::TILETYPE suitable_tile )
```

Keres egy optimális helyet terepkocka típus szerint.



## Paraméterek

<code>suitable_tile</code>	Az optimális terepkocka típusa.
----------------------------	---------------------------------

## Visszatérési érték

A szabad koordináta vektora. Ha nincs jó hely akkor a (-1,-1) vektor.

6.81.3.7. `get_resources()`

```
HumanResources & WorldBase::get_resources ( )
```

Külső referenciát ad a tárolt erőforrások elérésére.

6.81.3.8. `get_structure_type()`

```
minerals::Structure * WorldBase::get_structure_type (
    minerals::MINERAL_TYPE atype )
```

Keres egy olyan erőforrás struktúrát, ami bizonyos erőforrást tartalmaz.

## Paraméterek

<code>atype</code>	Az erőforrás típusa, ami keresett.
--------------------	------------------------------------

## Visszatérési érték

Egy struktúra pointer.

6.81.3.9. `getTileAt()`

```
tiles::Tile & WorldBase::getTileAt (
    int x,
    int y ) const
```

Visszaadja a világban az [y][x]-edik terepkockát.

## Paraméterek

<code>x</code>	Oszlop index.
<code>y</code>	Sor index.

**Visszatérési érték**

Az x.Oszlop y.Sor-i terepkocka.

**6.81.3.10. remove\_structure\_at()**

```
void WorldBase::remove_structure_at (
    int x,
    int y )
```

Lerombol egy struktúrát egy bizonyos x és y koordinátán. Ezt az emberek hívják meg bányászatkor, favágáskor. Ha házra vagy városközpontre hívódik meg akkor hiba keletkezik.

**Paraméterek**

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

**6.81.3.11. spawn\_entity()**

```
template<typename T >
void WorldBase::spawn_entity (
    tiles::TILETYPE goal_habitat,
    const std::string & savefile_identifier )
```

Idéz egy entitást, egy bizonyos típusú biomba.

**Sablon paraméterek**

<i>T</i>	Az entitás típusa.
----------	--------------------

**Paraméterek**

<i>goal_habitat</i>	A terepkocka, ami a cél.
<i>savefile_identifier</i>	Ez egy azonosító, így fog a mentés fájlba megjelenni.

**6.81.3.12. spawn\_structure()**

```
template<typename T >
void WorldBase::spawn_structure (
    bool mountain_exclusive )
```

Idéz egy struktúrát. Opcionálisan bekapcsolható, hogy csak a hegyekben idéződjön.

## Sablon paraméterek

<i>T</i>	A struktúra.
----------	--------------

## Paraméterek

<i>mountain_exclusive</i>	Ha igaz, akkor csak a hegyekben idéződik a struktúra.
---------------------------	---

## 6.81.3.13. spawn\_structure\_at()

```
template<typename T >
void WorldBase::spawn_structure_at (
    int x,
    int y ) [inline]
```

Idéz egy struktúrát egy pontos x és y koordinátára.

## Sablon paraméterek

<i>T</i>	A struktúra.
----------	--------------

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

## 6.81.3.14. upgrade\_house\_at()

```
void WorldBase::upgrade_house_at (
    int x,
    int y )
```

Megpróbál megfrissíteni egy házat az (x,y) koordinátán. Ezt az építész ember hívja meg.

## Paraméterek

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.

## 6.81.4. Adattagok dokumentációja

#### 6.81.4.1. camp\_needs\_spawn

```
bool WorldBase::camp_needs_spawn [protected]
```

Kell-e idéznünk új városlakókat?

#### 6.81.4.2. current\_city\_center

```
minerals::CityCenter* WorldBase::current_city_center [protected]
```

A városközpont pointerének tárolása. Csak 1 városközpont lehet, ha 2-t próbálnak építeni, az hibás működést jelent.

#### 6.81.4.3. entities

```
std::vector<creature::Living*> WorldBase::entities [protected]
```

Az entitások pointerének tárolása. Nem tárol embert.

#### 6.81.4.4. houses

```
std::vector<minerals::House*> WorldBase::houses [protected]
```

Az emberek által épített lakóházak pointerének tárolásáért felelős tároló.

#### 6.81.4.5. humans

```
std::vector<creature::Human*> WorldBase::humans [protected]
```

Az ember pointerek tárolása.

#### 6.81.4.6. MAX\_OBJECT\_SIZE

```
constexpr int WorldBase::MAX_OBJECT_SIZE =64 [static], [constexpr], [protected]
```

Amikor kikerül valami ennyire messziről a látótérből akkor az már nem rajzolódik ki.

#### 6.81.4.7. sound\_player

```
SoundPlayer WorldBase::sound_player [protected]
```

Hanglejátszó modul. Mindennek hangja van a világban, ezért kell ez az osztály.

#### 6.81.4.8. structures

```
std::vector<minerals::ResourceStructure*> WorldBase::structures [protected]
```

Az erőforrást tartalmazó struktúrák pointerének (fa, bokor, kő, vasérc) tárolásáért felelős heterogén kollekció.

#### 6.81.4.9. terrain

```
TerrainContainer<tiles::Tile*> WorldBase::terrain [protected]
```

A terep tároló, 2 dimenziós dinamikus tömb.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

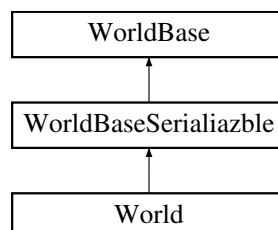
- src/World.hpp
- src/World.cpp
- src/WorldBase.cpp

## 6.82. WorldBaseSerialiazble osztályreferencia

A világ osztály bővítése, rendelkezik insertorral és extractorral.

```
#include <World.hpp>
```

A WorldBaseSerialiazble osztály származási diagramja:



### Publikus tagfüggvények

- virtual void `clear` ()=0

## Védett tagfüggvények

- virtual void [reinitialize\\_self](#) (bool from\_file)=0

## Védett attribútumok

- float [elapsed\\_time](#)  
*A világba eltelt idő másodpercben.*
- int [saved\\_size](#) =-1  
*A világ elmentett mérete.*

## Barátok

- std::ostream & [operator<<](#) (std::ostream &os, [WorldBaseSerialiazble](#) &w)  
*A világ adatait segíti kimenteni egy folyamba.*
- std::ifstream & [operator>>](#) (std::ifstream &in, [WorldBaseSerialiazble](#) &w)  
*Egy folyamból tölti fel a világot új adatokkal.*

## További örökölt tagok

### 6.82.1. Részletes leírás

A világ osztály bővítése, rendelkezik insertorral és extractorral.

Be lehet olvasni adatot és ki lehet belőle olvasni adatot.

### 6.82.2. Tagfüggvények dokumentációja

#### 6.82.2.1. clear()

```
virtual void WorldBaseSerialiazble::clear ( ) [pure virtual]
```

Megvalósítják a következők: [World](#).

#### 6.82.2.2. reinitialize\_self()

```
virtual void WorldBaseSerialiazble::reinitialize_self (
    bool from_file ) [protected], [pure virtual]
```

### 6.82.3. Barát és kapcsolódó függvények dokumentációja

### 6.82.3.1. operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    WorldBaseSerialiazble & w ) [friend]
```

A világ adatait segíti kimenteni egy folyamba.

### 6.82.3.2. operator>>

```
std::ifstream& operator>> (
    std::ifstream & in,
    WorldBaseSerialiazble & w ) [friend]
```

Egy folyamból tölti fel a világot új adatokkal.

## 6.82.4. Adattagok dokumentációja

### 6.82.4.1. elapsed\_time

```
float WorldBaseSerialiazble::elapsed_time [protected]
```

A világba eltelt idő másodpercben.

### 6.82.4.2. saved\_size

```
int WorldBaseSerialiazble::saved_size =-1 [protected]
```

A világ elmentett mérete.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/[World.hpp](#)
- src/[WorldBaseSerializable.cpp](#)

## 6.83. YAMLParser osztályreferencia

Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.

```
#include <YAMLParser.h>
```

## Publikus tagfüggvények

- [YAMLParse](#) ()  
*Az alap konstruktor, semmit nem csinál csak jelzi.*
- bool [parse\\_file](#) (const std::string &filepath)  
*Beolvassa a YML fájlt. Vigyáz a kommentekre.*
- std::string [get\\_value\\_of\\_key](#) (const std::string &key) const  
*Arra szolgál, hogy a beolvasott fájl tokenjeit kívülről is el lehet érni.*
- bool [try\\_generate\\_config\\_file](#) (const std::string &filename)

### 6.83.1. Részletes leírás

Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.

A YML specifikációnak megfelelően be tud olvasni YML fájlokat (.yml).

### 6.83.2. Konstruktorok és destruktorok dokumentációja

#### 6.83.2.1. [YAMLParse](#)()

```
YAMLParse::YAMLParse ( )
```

Az alap konstruktor, semmit nem csinál csak jelzi.

### 6.83.3. Tagfüggvények dokumentációja

#### 6.83.3.1. [get\\_value\\_of\\_key](#)()

```
std::string YAMLParse::get\_value\_of\_key (  
    const std::string & key ) const
```

Arra szolgál, hogy a beolvasott fájl tokenjeit kívülről is el lehet érni.

#### 6.83.3.2. [parse\\_file](#)()

```
bool YAMLParse::parse\_file (  
    const std::string & filepath )
```

Beolvassa a YML fájlt. Vigyáz a kommentekre.

#### 6.83.3.3. [try\\_generate\\_config\\_file](#)()

```
bool YAMLParse::try\_generate\_config\_file (  
    const std::string & filename )
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/YAMLParse.h](#)
- [src/YAMLParse.cpp](#)



## 7. fejezet

# Fájlok dokumentációja

### 7.1. src/creatures/EntityBase.cpp fájlreferencia

```
#include "EntityBase.h"  
#include "../exceptions/FileExceptions.h"
```

#### Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

### 7.2. src/creatures/EntityBase.d fájlreferencia

### 7.3. src/creatures/EntityBase.h fájlreferencia

```
#include "EntityUtils.h"  
#include "../Utils.h"
```

#### Osztályok

- class [creature::EntityBase](#)

*Egy alap, nem rajzolható entitás osztálya.*

#### Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.4. src/creatures/EntityUtils.h fájlreferencia

```
#include <string>
#include <iostream>
#include "../GameConfig.h"
```

### Osztályok

- class [creature::LivingTexture](#)

*Az élő entitások kinézetének adatai.*

### Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

### Enumerációk

- enum class [creature::ENTITY\\_TYPE](#) : char { [creature::HUMAN](#) , [creature::ANIMAL](#) , [creature::ROBOTIC](#) }
- enum class [creature::ENTITY\\_GENDER](#) : char { [creature::MALE](#) , [creature::FEMALE](#) }
- enum class [creature::FACING](#) : bool { [creature::RIGHT](#) , [creature::LEFT](#) }
- enum class [creature::LIVINGSTATE](#) : int {  
    [creature::IDLE](#) , [creature::RUN](#) , [creature::WALK](#) , [creature::DEATH](#) ,  
    [creature::ATTACKING](#) , [creature::DOING\\_ITS\\_WORK](#) }

## 7.5. src/creatures/Goat.cpp fájlreferencia

```
#include "Goat.h"
#include "../World.hpp"
```

### Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.6. src/creatures/Goat.d fájlreferencia

## 7.7. src/creatures/Goat.h fájlreferencia

```
#include "Living.h"
#include "../Random_Gen.h"
#include "../Utils.h"
```

## Osztályok

- class `creature::Goat`  
*A kecske osztály leírása.*

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.8. src/creatures/HostileInterface.cpp fájlreferencia

```
#include "HostileInterface.h"  
#include "../World.hpp"
```

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.9. src/creatures/HostileInterface.d fájlreferencia

## 7.10. src/creatures/HostileInterface.h fájlreferencia

```
#include "Living.h"
```

## Osztályok

- class `creature::HostileInterface`  
*A vadállat entitások interface leírása.*

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.11. src/creatures/hostiles/Bear.cpp fájlreferencia

```
#include "Bear.h"  
#include "../../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.12. src/creatures/hostiles/Bear.d fájlreferencia

## 7.13. src/creatures/hostiles/Bear.h fájlreferencia

```
#include "../HostileInterface.h"
#include "../../Random_Gen.h"
#include "../../Utils.h"
```

## Osztályok

- class [creature::Bear](#)

*A medve osztály leírása.*

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.14. src/creatures/hostiles/Crocodile.cpp fájlreferencia

```
#include "Crocodile.h"
#include "../../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.15. src/creatures/hostiles/Crocodile.d fájlreferencia

## 7.16. src/creatures/hostiles/Crocodile.h fájlreferencia

```
#include "../HostileInterface.h"
#include "../../Random_Gen.h"
#include "../../Utils.h"
```

## Osztályok

- class `creature::Crocodile`  
*A krokodil osztály leírása.*

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.17. src/creatures/hostiles/KillerRobot.cpp fájlreferencia

```
#include "KillerRobot.h"  
#include "../World.hpp"
```

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.18. src/creatures/hostiles/KillerRobot.d fájlreferencia

## 7.19. src/creatures/hostiles/KillerRobot.h fájlreferencia

```
#include "../HostileInterface.h"  
#include "../Random_Gen.h"  
#include "../Utils.h"
```

## Osztályok

- class `creature::KillerRobot`  
*A gyilkos robot osztály leírása.*

## Névterek

- `creature`  
*Az összes élőlény és entitás ebben a névtérben van.*

## 7.20. src/creatures/humans/AnglerMiner.cpp fájlreferencia

```
#include "AnglerMiner.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.21. src/creatures/humans/AnglerMiner.d fájlreferencia

## 7.22. src/creatures/humans/AnglerMiner.h fájlreferencia

```
#include "Fisherman.h"  
#include "Stonemason.h"
```

## Osztályok

- class [creature::AnglerMiner](#)

*Az "AnglerMiner" szakmájú ember osztály leírása.*

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.23. src/creatures/humans/Builder.cpp fájlreferencia

```
#include "Builder.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.24. src/creatures/humans/Builder.d fájlreferencia

## 7.25. src/creatures/humans/Builder.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class `creature::Builder`

*Az építész szakmájú ember osztály leírása.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.26. src/creatures/humans/Farmer.cpp fájlreferencia

```
#include "Farmer.h"  
#include "../World.hpp"
```

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.27. src/creatures/humans/Farmer.d fájlreferencia

## 7.28. src/creatures/humans/Farmer.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class `creature::Farmer`

*A farmer szakmájú ember osztály leírása.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.29. src/creatures/humans/Fisherman.cpp fájlreferencia

```
#include "Fisherman.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.30. src/creatures/humans/Fisherman.d fájlreferencia

## 7.31. src/creatures/humans/Fisherman.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class [creature::Fisherman](#)

*A halász szakmájú ember osztály leírása.*

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.32. src/creatures/humans/Human.cpp fájlreferencia

```
#include "Human.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.33. src/creatures/humans/Human.d fájlreferencia

## 7.34. src/creatures/humans/Human.h fájlreferencia

```
#include "../Living.h"  
#include "../Random_Gen.h"  
#include "../Profession.h"  
#include "../world_object/CityCenter.h"  
#include "../Utils.h"
```



## Osztályok

- class `creature::Human`

*Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.35. src/creatures/humans/King.cpp fájlreferencia

```
#include "King.h"  
#include "../World.hpp"
```

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.36. src/creatures/humans/King.d fájlreferencia

## 7.37. src/creatures/humans/King.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class `creature::King`

*A király szakmájú ember osztály leírása.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.38. src/creatures/humans/Soldier.cpp fájlreferencia

```
#include "Soldier.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.39. src/creatures/humans/Soldier.d fájlreferencia

## 7.40. src/creatures/humans/Soldier.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class [creature::Soldier](#)

*A katona szakmájú ember osztály leírása.*

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.41. src/creatures/humans/Stonemason.cpp fájlreferencia

```
#include "Stonemason.h"  
#include "../World.hpp"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.42. src/creatures/humans/Stonemason.d fájlreferencia

## 7.43. src/creatures/humans/Stonemason.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class `creature::Stonemason`

*A bányász szakmájú ember osztály leírása.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.44. src/creatures/humans/Woodcutter.cpp fájlreferencia

```
#include "Woodcutter.h"  
#include "../World.hpp"
```

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.45. src/creatures/humans/Woodcutter.d fájlreferencia

## 7.46. src/creatures/humans/Woodcutter.h fájlreferencia

```
#include "Human.h"
```

## Osztályok

- class `creature::Woodcutter`

*A favágó szakmájú ember osztály leírása.*

## Névterek

- `creature`

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.47. src/creatures/Living.cpp fájlreferencia

```
#include "Living.h"
```

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.48. src/creatures/Living.d fájlreferencia

## 7.49. src/creatures/Living.h fájlreferencia

```
#include "../Textureable.h"
#include "../TextureManager.h"
#include "../GameConfig.h"
#include "../Shadowable.h"
#include <string>
#include <iostream>
#include "EntityBase.h"
```

## Osztályok

- class [creature::Living](#)

*Az élő entitások interface leírása.*

## Névterek

- [creature](#)

*Az összes élőlény és entitás ebben a névtérben van.*

## 7.50. src/EntityPlacer.cpp fájlreferencia

```
#include "EntityPlacer.h"
```

## 7.51. src/EntityPlacer.d fájlreferencia

## 7.52. src/EntityPlacer.h fájlreferencia

```
#include <string>
#include <unordered_map>
#include <algorithm>
#include "World.hpp"
#include "GameConfig.h"
#include "creatures/Living.h"
#include "creatures/humans/Human.h"
```

```
#include "creatures/humans/King.h"
#include "creatures/Goat.h"
#include "creatures/hostiles/Crocodile.h"
#include "creatures/hostiles/Bear.h"
#include "creatures/hostiles/KillerRobot.h"
#include "world_object/BerryBush.h"
#include "world_object/Stone.h"
#include "world_object/Tree.h"
#include "world_object/Iron.h"
#include "Utils.h"
```

## Osztályok

- class [ObjectRegistry](#)  
*Az entitások és más világ objektumok lerakásának intézéséért felelős osztály.*
- class [EntityPlacer](#)  
*Az entitások a kattintással való lerakása.*

## 7.53. src/exceptions/FileExceptions.h fájlreferencia

```
#include "SimulationException.h"
```

## Osztályok

- class [ImportInvalidEntityException](#)  
*Akkor kell dobni, ha egy entitás hibásan lett beolvasva.*
- class [ImportInvalidHumanProfessionException](#)  
*Akkor kell dobni, ha egy szakma hibásan lett beolvasva.*
- class [ImportInvalidHousingLevelException](#)  
*Akkor kell dobni, ha egy ház hibásan lett beolvasva.*
- class [ImportInvalidResourceException](#)  
*Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva.*
- class [ReadSaveFileFail](#)  
*Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás.*

## 7.54. src/exceptions/MusicLoadException.h fájlreferencia

```
#include "SimulationException.h"
```

## Osztályok

- class [MusicLoadException](#)  
*Akkor kell dobni, ha egy zene hibásan lett lejátszva vagy hibás a beolvasott zene.*

## 7.55. src/exceptions/SimulationException.h fájlreferencia

```
#include <stdexcept>
#include <string>
```

### Osztályok

- class [SimulationException](#)  
*Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik.*

## 7.56. src/exceptions/WorldExceptions.h fájlreferencia

```
#include "SimulationException.h"
```

### Osztályok

- class [CityCenterException](#)  
*Akkor kell dobni, ha a városközpont hibásan működött.*
- class [StructureException](#)  
*Akkor kell dobni, ha egy struktúra hibásan működött.*
- class [InvalidBorderSizeException](#)  
*Akkor kell dobni, ha egy világhatárnak nem jó értéket akarnak beállítani.*

## 7.57. src/external/memtrace.cpp fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <ctype.h>
```

## 7.58. src/external/memtrace.h fájlreferencia

## 7.59. src/external/modified\_gtest\_lite.h fájlreferencia

```
#include <iostream>
#include <cassert>
#include <cmath>
#include <cstring>
#include <limits>
#include <cstdlib>
#include <string>
#include <fstream>
```

## Osztályok

- struct [\\_Is\\_Types< F, T >](#)  
*Segédsablon típuskonverzió futás közbeni ellenőrzésére.*
- struct [gtest\\_lite::Test](#)
- class [gtest\\_lite::ostreamRedir](#)

## Névterek

- [gtest\\_lite](#)  
*[gtest\\_lite](#): a keretrendszer függvényinek és objektumainak névtere*

## Makródefiníciók

- #define [TEST](#)(C, N) do { gtest\_lite::test.begin(#C".#N");
- #define [END](#) gtest\_lite::test.end(); } while (false);  
*Tesztet vége.*
- #define [ENDM](#) gtest\_lite::test.end(true); } while (false);
- #define [ENDMSG](#)(t) gtest\_lite::test.end(true) << t << std::endl; } while (false);
- #define [SUCCEED](#)() gtest\_lite::test.expect(true, \_\_FILE\_\_, \_\_LINE\_\_, "SUCCEED()", true)  
*Sikeres teszt makrója.*
- #define [FAIL](#)() gtest\_lite::test.expect(false, \_\_FILE\_\_, \_\_LINE\_\_, "FAIL()", true)  
*Sikertelen teszt fatális hiba makrója.*
- #define [ADD\\_FAILURE](#)() gtest\_lite::test.expect(false, \_\_FILE\_\_, \_\_LINE\_\_, "ADD\_FAILURE()", true)  
*Sikertelen teszt makrója.*
- #define [EXPECT\\_EQ](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::eq](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_EQ(" #expected ", " #actual ")")  
*Azonosságot elváró makró*
- #define [EXPECT\\_NE](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::ne](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_NE(" #expected ", " #actual ")", "etalon" )  
*Eltérést elváró makró*
- #define [EXPECT\\_LE](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::le](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_LE(" #expected ", " #actual ")", "etalon" )  
*Kisebb, vagy egyenlő relációt elváró makró*
- #define [EXPECT\\_LT](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::lt](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_LT(" #expected ", " #actual ")", "etalon" )  
*Kisebb, mint relációt elváró makró*
- #define [EXPECT\\_GE](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::ge](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_GE(" #expected ", " #actual ")", "etalon" )  
*Nagyobb, vagy egyenlő relációt elváró makró*
- #define [EXPECT\\_GT](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::gt](#), \_\_FILE\_\_, \_\_↵  
\_\_LINE\_\_, "EXPECT\_GT(" #expected ", " #actual ")", "etalon" )  
*Nagyobb, mint relációt elváró makró*
- #define [EXPECT\\_TRUE](#)(actual) [gtest\\_lite::EXPECT\\_](#)(true, actual, [gtest\\_lite::eq](#), \_\_FILE\_\_, \_\_LINE\_\_↵  
, "EXPECT\_TRUE(" #actual ")")  
*Igaz értéket elváró makró*
- #define [EXPECT\\_FALSE](#)(actual) [gtest\\_lite::EXPECT\\_](#)(false, actual, [gtest\\_lite::eq](#), \_\_FILE\_\_, \_\_LINE\_\_↵  
, "EXPECT\_FALSE(" #actual ")")  
*Hamis értéket elváró makró*
- #define [EXPECT\\_FLOAT\\_EQ](#)(expected, actual) [gtest\\_lite::EXPECT\\_](#)(expected, actual, [gtest\\_lite::almostEQ](#),  
\_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_FLOAT\_EQ(" #expected ", " #actual ")")

- Valós számok azonosságát elváró makró
  - #define `EXPECT_DOUBLE_EQ`(expected, actual) `gtest_lite::EXPECT_`(expected, actual, `gtest_lite::almostEQ`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_DOUBLE\_EQ(" #expected ", " #actual ")")
  - Valós számok azonosságát elváró makró
  - #define `EXPECT_STREQ`(expected, actual) `gtest_lite::EXPECTSTR`(expected, actual, `gtest_lite::eqstr`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_STREQ(" #expected ", " #actual ")")
  - C stringek (const char \*) azonosságát tesztelő makró
  - #define `EXPECT_STRNE`(expected, actual) `gtest_lite::EXPECTSTR`(expected, actual, `gtest_lite::nestr`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_STRNE(" #expected ", " #actual ")", "etalon")
  - C stringek (const char \*) eltérést tesztelő makró
  - #define `EXPECT_STRCASEEQ`(expected, actual) `gtest_lite::EXPECTSTR`(expected, actual, `gtest_lite::eqstrcase`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_STRCASEEQ(" #expected ", " #actual ")")
  - C stringek (const char \*) azonosságát tesztelő makró (kisbetű/nagybetű azonos)
  - #define `EXPECT_STRCASENE`(expected, actual) `gtest_lite::EXPECTSTR`(expected, actual, `gtest_lite::nestrcase`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_STRCASENE(" #expected ", " #actual ")", "etalon")
  - C stringek (const char \*) eltérést tesztelő makró (kisbetű/nagybetű azonos)
  - #define `EXPECT_THROW`(statement, exception\_type)
  - Kivételt várunk.
  - #define `EXPECT_ANY_THROW`(statement)
  - Kivételt várunk.
  - #define `EXPECT_NO_THROW`(statement)
  - Nem várunk kivételt.
  - #define `ASSERT_NO_THROW`(statement)
  - Nem várunk kivételt.
  - #define `EXPECT_THROW_THROW`(statement, exception\_type)
  - Kivételt várunk és továbbdobjuk – ilyen nincs a gtest-ben.
  - #define `EXPECT_ENVEQ`(expected, actual) `gtest_lite::EXPECTSTR`(std::getenv(expected), actual, `gtest_lite::eqstr`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_ENVEQ(" #expected ", " #actual ")")
  - Környezeti változóhoz hasonlít – ilyen nincs a gtest-ben.
  - #define `EXPECT_ENVCASEEQ`(expected, actual) `gtest_lite::EXPECTSTR`(std::getenv(expected), actual, `gtest_lite::eqstrcase`, \_\_FILE\_\_, \_\_LINE\_\_, "EXPECT\_ENVCASEEQ(" #expected ", " #actual ")")
  - Környezeti változóhoz hasonlít – ilyen nincs a gtest-ben (kisbetű/nagybetű azonos)
  - #define `ASSERT_EQ`(expected, actual) `gtest_lite::ASSERT_`(expected, actual, `gtest_lite::eq`, "ASSER\_EQ")
  - Azonosságot elváró makró
  - #define `ASSERT_NO_THROW`(statement)
  - Nem várunk kivételt.
  - #define `CREATE_Has`(X)
  - #define `CREATE_Has_fn`(X, S)
  - #define `EXPECTTHROW`(statement, exp, act)
  - `EXPECTTHROW`: kivételkezelés.
  - #define `ASSERTTHROW`(statement, exp, act)
  - #define `ASSERT_`(expected, actual, fn, op)
  - #define `GTINIT`(IS)
  - #define `GTEND`(os)

## Függvények

- void `hasMember` (...)
- template<typename T1, typename T2 >  
std::ostream & `gtest_lite::EXPECT_` (T1 exp, T2 act, bool(\*pred)(T1, T1), const char \*file, int line, const char \*expr, const char \*lhs="elvar", const char \*rhs="aktual")  
általános sablon a várt értékhez.



- `template<typename T1, typename T2 >`  
`std::ostream & gtest_lite::EXPECT_ (T1 *exp, T2 *act, bool(*pred)(T1 *, T1 *), const char *file, int line, const char *expr, const char *lhs="elvart", const char *rhs="aktual")`  
*pointerre specializált sablon a várt értékhez.*
- `std::ostream & gtest_lite::EXPECTSTR (const char *exp, const char *act, bool(*pred)(const char *, const char *), const char *file, int line, const char *expr, const char *lhs="elvart", const char *rhs="aktual")`
- `template<typename T >`  
`bool gtest_lite::eq (T a, T b)`
- `bool gtest_lite::eqstr (const char *a, const char *b)`
- `bool gtest_lite::eqstrcase (const char *a, const char *b)`
- `template<typename T >`  
`bool gtest_lite::ne (T a, T b)`
- `bool gtest_lite::nestr (const char *a, const char *b)`
- `template<typename T >`  
`bool gtest_lite::le (T a, T b)`
- `template<typename T >`  
`bool gtest_lite::lt (T a, T b)`
- `template<typename T >`  
`bool gtest_lite::ge (T a, T b)`
- `template<typename T >`  
`bool gtest_lite::gt (T a, T b)`
- `template<typename T >`  
`bool gtest_lite::almostEQ (T a, T b)`

## 7.59.1. Makródefiníciók dokumentációja

### 7.59.1.1. ADD\_FAILURE

```
#define ADD_FAILURE( ) gtest_lite::test.expect(false, __FILE__, __LINE__, "ADD_FAILURE()", true)
```

Sikertelen teszt makrója.

### 7.59.1.2. ASSERT\_

```
#define ASSERT_(  
    expected,  
    actual,  
    fn,  
    op )
```

Érték:

```
EXPECT_(expected, actual, fn, __FILE__, __LINE__, #op "(" #expected ", " #actual ")" ); \
if (!gtest_lite::test.status) { gtest_lite::test.end(); break; }
```

### 7.59.1.3. ASSERT\_EQ

```
#define ASSERT_EQ(  
    expected,  
    actual ) gtest_lite::ASSERT_(expected, actual, gtest_lite::eq, "ASSERT_EQ")
```

Azonosságot elváró makró

ASSERT típusú ellenőrzések. Csak 1-2 van megvalósítva. Nem ostream& -val térnek vissza !!! Kivételt várunk

### 7.59.1.4. ASSERT\_NO\_THROW [1/2]

```
#define ASSERT_NO_THROW(  
    statement )
```

Érték:

```
try { gtest_lite::test.tmp = true; statement; } \  
catch (...) { gtest_lite::test.tmp = false; }\  
ASSERT_THROW(statement, "nem dob kivetelt.", "kivetelt dobott.")
```

Nem várunk kivételt.

### 7.59.1.5. ASSERT\_NO\_THROW [2/2]

```
#define ASSERT_NO_THROW(  
    statement )
```

Érték:

```
try { gtest_lite::test.tmp = true; statement; } \  
catch (...) { gtest_lite::test.tmp = false; }\  
ASSERT_THROW(statement, "nem dob kivetelt.", "kivetelt dobott.")
```

Nem várunk kivételt.

### 7.59.1.6. ASSERTTHROW

```
#define ASSERTTHROW(  
    statement,  
    exp,  
    act )
```

Érték:

```
gtest_lite::test.expect(gtest_lite::test.tmp, __FILE__, __LINE__, #statement) \  
« "*** Az utasítás " « (act) \  
« "\n** Azt vartuk, hogy " « (exp) « std::endl; if (!gtest_lite::test.status) { gtest_lite::test.end();  
break; }
```

### 7.59.1.7. CREATE\_Has\_

```
#define CREATE_Has_  
    X )
```

Érték:

```
template<typename T> struct _Has_##X { \  
    struct Fallback { int X; }; \  
    struct Derived : T, Fallback {}; \  
    template<typename C, C> struct ChT; \  
    template<typename D> static char (&f(ChT<int Fallback::*, &D::X>*)) [1]; \  
    template<typename D> static char (&f(...)) [2]; \  
    static bool const member = sizeof(f<Derived>(0)) == 2; \  
};
```

Segédmakró egy adattag, vagy tagfüggvény létezésének tesztelésére futási időben Ötlet: <https://cpptalk.wordpress.com/2009/09/12/substitution-failure-is-not-an-error-2>

Használat: `CREATE_Has_(size) ... if (_Has_size<std::string>::member)...`

**7.59.1.8. CREATE\_Has\_fn\_**

```
#define CREATE_Has_fn_(
    X,
    S )
```

**Érték:**

```
template<typename R, typename T> struct _Has_fn_##X##_##S { \
    template<typename C, R (C::*f)() S> struct ChT; \
    template<typename D> static char (&f(ChT<D, &D::X>*)) [1]; \
    template<typename D> static char (&f(...)) [2]; \
    static bool const fn = sizeof(f<T>(0)) == 1; \
};
```

**7.59.1.9. END**

```
#define END gtest_lite::test.end(); } while (false);
```

Tesztet vége.

**7.59.1.10. ENDM**

```
#define ENDM gtest_lite::test.end(true); } while (false);
```

Tesztet vége allokált blokkok számának összehasonlításával Ez az ellenőrzés nem bomba biztos.

**7.59.1.11. ENDMsg**

```
#define ENDMsg(
    t ) gtest_lite::test.end(true) << t << std::endl; } while (false);
```

Tesztet vége allokált blokkok számának összehasonlításával Ez az ellenőrzés nem bomba biztos. Ha hiba van kiírja az üzenetet.

**7.59.1.12. EXPECT\_ANY\_THROW**

```
#define EXPECT_ANY_THROW(
    statement )
```

**Érték:**

```
try { gtest_lite::test.tmp = false; statement; } \
catch (...) { gtest_lite::test.tmp = true; } \
EXPECT_THROW(statement, "kivetelt dob.", "nem dobott kivetelt.")
```

Kivételt várunk.

**7.59.1.13. EXPECT\_DOUBLE\_EQ**

```
#define EXPECT_DOUBLE_EQ(
    expected,
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::almostEQ, __FILE__, ↵
    __LINE__, "EXPECT_DOUBLE_EQ(" #expected ", " #actual ")")
```

Valós számok azonosságát elváró makró

**7.59.1.14. EXPECT\_ENVCASEEQ**

```
#define EXPECT_ENVCASEEQ(
    expected,
    actual ) gtest_lite::EXPECTSTR(std::getenv(expected), actual, gtest_lite::eqstrcase,
    __FILE__, __LINE__, "EXPECT_ENVCASEEQ(" #expected ", " #actual ")")
```

Környezeti változóhoz hasonlít – ilyen nincs a gtest-ben (kisbetű/nagybetű azonos)

#### 7.59.1.15. EXPECT\_ENVEQ

```
#define EXPECT_ENVEQ(  
    expected,  
    actual ) gtest_lite::EXPECTSTR(std::getenv(expected), actual, gtest_lite::eqstr,  
    __FILE__, __LINE__, "EXPECT_ENVEQ(" #expected ", " #actual ")" )
```

Környezeti változóhoz hasonlít – ilyen nincs a gtest-ben.

#### 7.59.1.16. EXPECT\_EQ

```
#define EXPECT_EQ(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::eq, __FILE__, __LINE__,  
    ____, "EXPECT_EQ(" #expected ", " #actual ")" )
```

Azonosságot elváró makró

#### 7.59.1.17. EXPECT\_FALSE

```
#define EXPECT_FALSE(  
    actual ) gtest_lite::EXPECT_(false, actual, gtest_lite::eq, __FILE__, __LINE__,  
    "EXPECT_FALSE(" #actual ")" )
```

Hamis értéket elváró makró

#### 7.59.1.18. EXPECT\_FLOAT\_EQ

```
#define EXPECT_FLOAT_EQ(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::almostEq, __FILE__, __  
    __LINE__, "EXPECT_FLOAT_EQ(" #expected ", " #actual ")" )
```

Valós számok azonosságát elváró makró

#### 7.59.1.19. EXPECT\_GE

```
#define EXPECT_GE(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::ge, __FILE__, __LINE__,  
    ____, "EXPECT_GE(" #expected ", " #actual ")", "etalon" )
```

Nagyobb, vagy egyenlő relációt elváró makró

#### 7.59.1.20. EXPECT\_GT

```
#define EXPECT_GT(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::gt, __FILE__, __LINE__,  
    ____, "EXPECT_GT(" #expected ", " #actual ")", "etalon" )
```

Nagyobb, mint relációt elváró makró

#### 7.59.1.21. EXPECT\_LE

```
#define EXPECT_LE(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::le, __FILE__, __LINE__,  
    ____, "EXPECT_LE(" #expected ", " #actual ")", "etalon" )
```

Kisebb, vagy egyenlő relációt elváró makró

**7.59.1.22. EXPECT\_LT**

```
#define EXPECT_LT(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::lt, __FILE__, __LINE__  
    __, "EXPECT_LT(" #expected ", " #actual ")", "etalon" )
```

Kiseb, mint relációt elváró makró

**7.59.1.23. EXPECT\_NE**

```
#define EXPECT_NE(  
    expected,  
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::ne, __FILE__, __LINE__  
    __, "EXPECT_NE(" #expected ", " #actual ")", "etalon" )
```

Eltérést elváró makró

**7.59.1.24. EXPECT\_NO\_THROW**

```
#define EXPECT_NO_THROW(  
    statement )
```

Érték:

```
try { gtest_lite::test.tmp = true; statement; } \  
catch (...) { gtest_lite::test.tmp = false; }\  
EXPECT_THROW(statement, "nem dob kivetelt.", "kivetelt dobott.")
```

Nem várunk kivételt.

**7.59.1.25. EXPECT\_STRCASEEQ**

```
#define EXPECT_STRCASEEQ(  
    expected,  
    actual ) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::eqstrcase, __FILE__  
    __, __LINE__, "EXPECT_STRCASEEQ(" #expected ", " #actual ")" )
```

C stringek (const char \*) azonosságát tesztelő makró (kisbetű/nagybetű azonos)

**7.59.1.26. EXPECT\_STRCASENE**

```
#define EXPECT_STRCASENE(  
    expected,  
    actual ) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::nestrcase, __FILE__  
    __, __LINE__, "EXPECT_STRCASENE(" #expected ", " #actual ")", "etalon" )
```

C stringek (const char \*) eltérést tesztelő makró (kisbetű/nagybetű azonos)

**7.59.1.27. EXPECT\_STREQ**

```
#define EXPECT_STREQ(  
    expected,  
    actual ) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::eqstr, __FILE__, __  
    __LINE__, "EXPECT_STREQ(" #expected ", " #actual ")" )
```

C stringek (const char \*) azonosságát tesztelő makró

### 7.59.1.28. EXPECT\_STRNE

```
#define EXPECT_STRNE(
    expected,
    actual ) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::nestr, __FILE__, __LINE__, "EXPECT_STRNE(" #expected ", " #actual ")", "etalon" )
```

C stringek (const char \*) eltérést tesztelő makró

### 7.59.1.29. EXPECT\_THROW

```
#define EXPECT_THROW(
    statement,
    exception_type )
```

Érték:

```
try { gtest_lite::test.tmp = false; statement; } \
catch (exception_type &e) { gtest_lite::test.tmp = true; } \
catch (...) { } \
EXPECT_THROW(statement, "kivetelt dob.", "nem dobott '" #exception_type "' kivetelt.")
```

Kivételt várunk.

### 7.59.1.30. EXPECT\_THROW\_THROW

```
#define EXPECT_THROW_THROW(
    statement,
    exception_type )
```

Érték:

```
try { gtest_lite::test.tmp = false; statement; } \
catch (exception_type &e) { gtest_lite::test.tmp = true; throw; } \
EXPECT_THROW(statement, "kivetelt dob.", "nem dobott '" #exception_type "' kivetelt.")
```

Kivételt várunk és továbbdobjuk – ilyen nincs a gtest-ben.

### 7.59.1.31. EXPECT\_TRUE

```
#define EXPECT_TRUE(
    actual ) gtest_lite::EXPECT_(true, actual, gtest_lite::eq, __FILE__, __LINE__, "EXPECT_TRUE(" #actual ")")
```

Igaz értéket elváró makró

### 7.59.1.32. EXPECT\_THROW

```
#define EXPECT_THROW(
    statement,
    exp,
    act )
```

Érték:

```
gtest_lite::test.expect(gtest_lite::test.tmp, __FILE__, __LINE__, #statement) \
« "*** Az utasítás " « (act) \
« "\n** Azt vartuk, hogy " « (exp) « std::endl
```

**EXPECT\_THROW: kivételkezelés.**

Belső megvalósításhoz tartozó makrók, és osztályok.

### 7.59.1.33. Nem célszerű közvetlenül használni, vagy módosítani

### 7.59.1.34. FAIL

```
#define FAIL( ) gtest_lite::test.expect(false, __FILE__, __LINE__, "FAIL()", true)
```

Sikertelen teszt fatális hiba makrója.

**7.59.1.35. GTEND**

```
#define GTEND(
    os )
```

**7.59.1.36. GTINIT**

```
#define GTINIT(
    IS )
```

**7.59.1.37. SUCCEED**

```
#define SUCCEED( ) gtest_lite::test.expect(true, __FILE__, __LINE__, "SUCCEED()", true)
```

Sikeres teszt makrója.

**7.59.1.38. TEST**

```
#define TEST(
    C,
    N ) do { gtest_lite::test.begin(#C".#N);
```

Teszt kezdete. A makró paraméterezése hasonlít a gtest paraméterezéséhez. Így az itt elkészített tesztek könnyen átemelhetők a gtest keretrendszerbe.

**Paraméterek**

<i>C</i>	- teszteset neve (csak a gtest kompatibilitás miatt van külön neve az eseteknek)
<i>N</i>	- teszt neve

**7.59.2. Függvények dokumentációja****7.59.2.1. hasMember()**

```
void hasMember (
    ... ) [inline]
```

Segédfüggvény egy publikus adattag, vagy tagfüggvény létezésének tesztelésére fordítási időben

**7.60. src/fake\_sfml/fake\_sfml.cpp fájlreferencia**

```
#include "fake_sfml.h"
```

**Névterek**

- [sf](#)

**Függvények**

- `std::string sf::to_string` (const Color &c)
- `bool sf::file_exists_at_path` (const std::string &name)

## 7.61. src/fake\_sfml/fake\_sfml.d fájlreferencia

## 7.62. src/fake\_sfml/fake\_sfml.h fájlreferencia

```
#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include "../Utils.h"
```

### Osztályok

- class [sf::Vector2f](#)
- class [sf::Transform](#)
- class [sf::FloatRect](#)
- class [sf::Vector2i](#)
- class [sf::Texture](#)
- class [sf::Bound](#)
- class [sf::Color](#)
- class [sf::IntRect](#)
- class [sf::Sprite](#)
- class [sf::Event](#)
- class [sf::ClockTime](#)
- class [sf::Clock](#)
- class [sf::SoundBuffer](#)
- class [sf::Sound](#)
- class [sf::SoundSource](#)
- class [sf::Music](#)
- class [sf::RectangleShape](#)
- class [sf::Keyboard](#)
- class [sf::RenderStates](#)
- class [sf::VideoMode](#)
- class [sf::RenderWindow](#)
- class [sf::Mouse](#)

### Névterek

- [sf](#)

### Enumerációk

- enum class [sf::BlendMode](#) {  
    [sf::None](#) , [sf::Alpha](#) , [sf::Additive](#) , [sf::Multiply](#) ,  
    [sf::BlendAdd](#) }

### Függvények

- bool [sf::file\\_exists\\_at\\_path](#) (const std::string &name)

### Változók

- constexpr BlendMode [sf::BlendAdd](#) = BlendMode::BlendAdd



## 7.63. src/GameConfig.cpp fájlreferencia

```
#include "GameConfig.h"
```

### Függvények

- `std::string trim` (`const std::string &str`)

### 7.63.1. Függvények dokumentációja

#### 7.63.1.1. trim()

```
std::string trim (
    const std::string & str )
```

## 7.64. src/GameConfig.d fájlreferencia

## 7.65. src/GameConfig.h fájlreferencia

```
#include "YAMLParse.h"
#include <mutex>
#include <iostream>
#include <string>
```

### Osztályok

- class `GameConfig`  
*A világ szimulációjának leírása.*

### Enumerációk

- enum class `Language` { `MAGYAR` , `ENGLISH` , `NONE` }

### 7.65.1. Enumerációk dokumentációja

#### 7.65.1.1. Language

```
enum Language [strong]
```

#### Enumeráció-értékek

MAGYAR	
ENGLISH	
NONE	

## 7.66. src/GameManager.cpp fájlreferencia

```
#include "GameManager.h"
```

## 7.67. src/GameManager.d fájlreferencia

## 7.68. src/GameManager.h fájlreferencia

```
#include "Utils.h"
#include <SFML/Graphics.hpp>
#include <iostream>
#include <vector>
#include "GameConfig.h"
#include "ui/button.h"
#include "World.hpp"
#include "EntityPlacer.h"
#include "PostProcessor.h"
#include "MusicPlayer.h"
#include "SaveManager.h"
#include "SoundPlayer.h"
```

### Osztályok

- class [GameManager](#)

*A világ szimulálásáért és a kirazolás irányításáért felelős osztály.*

## 7.69. src/HumanResources.cpp fájlreferencia

```
#include "HumanResources.h"
```

## 7.70. src/HumanResources.d fájlreferencia

## 7.71. src/HumanResources.h fájlreferencia

```
#include <string>
#include <unordered_map>
```

### Osztályok

- class [HumanResources](#)

*Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva.*

## 7.72. src/main.cpp fájlreferencia

```
#include "GameManager.h"
#include "GameConfig.h"
#include "World.hpp"
#include "TextureManager.h"
#include "Random_Gen.h"
#include <filesystem>
#include "external/modified_gtest_lite.h"
```

### Függvények

- int [main](#) ()

### 7.72.1. Függvények dokumentációja

#### 7.72.1.1. main()

```
int main ( )
```

### 7.73. src/main.d fájlreferencia

### 7.74. src/MusicPlayer.cpp fájlreferencia

```
#include "MusicPlayer.h"
```

### 7.75. src/MusicPlayer.d fájlreferencia

### 7.76. src/MusicPlayer.h fájlreferencia

```
#include "Utils.h"  
#include <SFML/Audio.hpp>  
#include <unordered_map>  
#include <string>  
#include <memory>  
#include "../exceptions/MusicLoadException.h"
```

#### Osztályok

- class [MusicPlayer](#)

*A zene játsszó osztály leírása.*

### 7.77. src/PostProcessor.cpp fájlreferencia

```
#include "PostProcessor.h"
```

### 7.78. src/PostProcessor.d fájlreferencia

### 7.79. src/PostProcessor.h fájlreferencia

```
#include "Utils.h"  
#include <SFML/Graphics.hpp>  
#include <string>  
#include "TextureManager.h"
```

#### Osztályok

- class [PostProcessor](#)

*A grafikus szépítő osztály leírása.*

## 7.80. src/Profession.cpp fájlreferencia

```
#include "Profession.h"
```

## 7.81. src/Profession.d fájlreferencia

## 7.82. src/Profession.h fájlreferencia

```
#include "Textureable.h"  
#include "TextureManager.h"
```

### Osztályok

- class [Profession](#)

*A szakma osztály leírása.*

## 7.83. src/Random\_Gen.cpp fájlreferencia

```
#include "Random_Gen.h"
```

## 7.84. src/Random\_Gen.d fájlreferencia

## 7.85. src/Random\_Gen.h fájlreferencia

```
#include <iostream>  
#include <chrono>  
#include <random>  
#include <mutex>
```

### Osztályok

- class [RandomGenerator](#)

*Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály.*

## 7.86. src/SaveHelpers.cpp fájlreferencia

```
#include "SaveHelpers.h"
```

### Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

- [creatures](#)

### Függvények

- std::string [minerals::mineral\\_to\\_string](#) (MINERAL\_TYPE type)

*Mentést elősegítő függvények.*

## 7.87. src/SaveHelpers.d fájlreferencia

## 7.88. src/SaveHelpers.h fájlreferencia

```
#include <string>
#include <unordered_map>
#include <functional>
#include "creatures/Living.h"
#include "creatures/humans/Human.h"
#include "creatures/humans/Woodcutter.h"
#include "creatures/humans/Farmer.h"
#include "creatures/humans/Stonemason.h"
#include "creatures/humans/Fisherman.h"
#include "creatures/humans/Builder.h"
#include "creatures/humans/King.h"
#include "creatures/humans/AnglerMiner.h"
#include "creatures/humans/Soldier.h"
#include "creatures/Goat.h"
#include "creatures/hostiles/Crocodile.h"
#include "creatures/hostiles/Bear.h"
#include "creatures/hostiles/KillerRobot.h"
#include "world_object/Structure.h"
#include "world_object/ResourceStructure.h"
#include "world_object/BerryBush.h"
#include "world_object/Stone.h"
#include "world_object/Tree.h"
#include "world_object/Iron.h"
#include "world_object/CityCenter.h"
#include "world_object/House.h"
```

### Osztályok

- struct [RoleOption](#)  
*Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni.*
- class [SaveHelper](#)  
*Factory-k.*

### Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

### Függvények

- std::string [minerals::mineral\\_to\\_string](#) (MINERAL\_TYPE type)  
*Mentést elősegítő függvények.*

## 7.89. src/SaveManager.cpp fájlreferencia

```
#include "SaveManager.h"
```

## 7.90. src/SaveManager.d fájlreferencia

## 7.91. src/SaveManager.h fájlreferencia

```
#include <iostream>
#include <string>
#include <fstream>
#include <cstdio>
#include "Utils.h"
#include "World.hpp"
#include "SaveHelpers.h"
#include "GameConfig.h"
#include "../exceptions/FileExceptions.h"
```

### Osztályok

- class [SaveManager](#)

*A fájl menedzseléshez szolgáló osztály leírása.*

## 7.92. src/Shadowable.cpp fájlreferencia

```
#include "Shadowable.h"
```

## 7.93. src/Shadowable.d fájlreferencia

## 7.94. src/Shadowable.h fájlreferencia

```
#include "TextureManager.h"
#include "GameConfig.h"
#include "Utils.h"
#include <SFML/Graphics.hpp>
#include <string>
#include <cmath>
```

### Osztályok

- class [Shadowable](#)

*Az árnyékoláshoz szükséges interface.*

## 7.95. src/SoundPlayer.cpp fájlreferencia

```
#include "SoundPlayer.h"
```

## 7.96. src/SoundPlayer.d fájlreferencia

## 7.97. src/SoundPlayer.h fájlreferencia

```
#include "Utils.h"
#include <SFML/Audio.hpp>
#include <unordered_map>
```

```
#include <string>
#include <memory>
#include <iostream>
```

## Osztályok

- class [SoundPlayer](#)

*A hanglejátszó osztály leírása.*

## 7.98. src/terrain\_tiles/Tile.cpp fájlreferencia

```
#include "Tile.h"
```

## Névterek

- [tiles](#)

*Az összes terepkocka elem ebben a névtérben van.*

## 7.99. src/terrain\_tiles/Tile.d fájlreferencia

## 7.100. src/terrain\_tiles/Tile.h fájlreferencia

```
#include "../Textureable.h"
#include "../TextureManager.h"
```

## Osztályok

- class [tiles::Tile](#)

*A terepkocka osztály leírása.*

## Névterek

- [tiles](#)

*Az összes terepkocka elem ebben a névtérben van.*

## Enumerációk

- enum class [tiles::TILETYPE](#) : char { [tiles::GRASS](#) , [tiles::WATER](#) , [tiles::MOUNTAIN](#) }

## 7.101. src/TerrainContainer.hpp fájlreferencia

A Világ terepének a deklarálása ebben a fájlba van.

```
#include "Utils.h"
#include <SFML/Graphics.hpp>
#include "GameManager.h"
#include "GameConfig.h"
#include "Random_Gen.h"
#include "terrain_tiles/Tile.h"
#include "TerrainContainer_def.hpp"
```

## Osztályok

- class [TerrainContainer< T >](#)  
*A világ terepét tároló osztály.*

### 7.101.1. Részletes leírás

A Világ terepének a deklarálása ebben a fájlba van.

#### Szerző

Funk Gábor

#### Dátum

2025-04-21

## 7.102. src/TerrainContainer\_def.hpp fájlreferencia

```
#include "TerrainContainer.hpp"
```

## 7.103. src/Textureable.h fájlreferencia

```
#include "Utils.h"  
#include <SFML/Graphics.hpp>  
#include <string>
```

## Osztályok

- class [Textureable](#)  
*Egy interface, ami a textúrázáshoz kell.*

## 7.104. src/TextureManager.cpp fájlreferencia

```
#include "TextureManager.h"
```

## 7.105. src/TextureManager.d fájlreferencia

## 7.106. src/TextureManager.h fájlreferencia

```
#include "Utils.h"  
#include <SFML/Graphics.hpp>  
#include <unordered_map>  
#include <string>  
#include <memory>  
#include <iostream>
```

## Osztályok

- class [TextureManager](#)  
*A Textúra kezelő osztály.*



## 7.107. src/ui/button.cpp fájlreferencia

```
#include "button.h"
```

### Névterek

- [ui](#)

*Az összes UI elem ebben a névtérben van.*

## 7.108. src/ui/button.d fájlreferencia

## 7.109. src/ui/button.h fájlreferencia

```
#include "../Textureable.h"
#include "../TextureManager.h"
#include <iostream>
#include <functional>
#include <string>
```

### Osztályok

- class [ui::Button](#)

*A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak.*

### Névterek

- [ui](#)

*Az összes UI elem ebben a névtérben van.*

## 7.110. src/Utils.cpp fájlreferencia

```
#include "Utils.h"
#include "GameConfig.h"
```

### Függvények

- double [distance\\_to](#) (double x1, double y1, double x2, double y2)  
*Visszaadja a távolságot az (x1,y1) (x2,y2) pont között.*
- void [log\\_text](#) (const std::string &english, const std::string &magyar)  
*Könnyebb tesztelést biztosít, ha az SFML magyarul írja ki, mit csinált. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.*
- void [warn\\_text](#) (const std::string &english, const std::string &magyar, int config\_minimum)  
*Könnyebb tesztelést biztosít, a program magyarul írja ki, mit csinál. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.*

### 7.110.1. Függvények dokumentációja

**7.110.1.1. distance\_to()**

```
double distance_to (
    double x1,
    double y1,
    double x2,
    double y2 )
```

Visszaadja a távolságot az (x1,y1) (x2,y2) pont között.

**Paraméterek**

<i>x1</i>	Az 1. pont x koordinátája.
<i>y1</i>	Az 1. pont y koordinátája.
<i>x2</i>	Az 2. pont x koordinátája.
<i>y2</i>	Az 2. pont y koordinátája.

**Visszatérési érték**

A távolság a 2 pont között.

**Figyelmeztetés**

Ez a függvény jelenleg nincs használatban.

**7.110.1.2. log\_text()**

```
void log_text (
    const std::string & english,
    const std::string & magyar )
```

Könnyebb tesztelést biztosít, ha az SFML magyarul írja ki, mit csinált. Van angol nyelvű is lehetőség és arra is, hogy ne írjon semmit.

**7.110.1.3. warn\_text()**

```
void warn_text (
    const std::string & english,
    const std::string & magyar,
    int config_minimum )
```

Könnyebb tesztelést biztosít, a program magyarul írja ki, mit csinál. Van angol nyelvű is lehetőség és arra is, hogy ne írjon semmit.

**7.111. src/Utils.d fájlreferencia****7.112. src/Utils.h fájlreferencia**

```
#include <cmath>
#include <string>
```

**Makródefiníciók**

- `#define WITH_SFML_RENDER`

*Ez arra kell, ha nem headless mode kell a programból.*

## Függvények

- double `distance_to` (double x1, double y1, double x2, double y2)  
*Visszaadja a távolságot az (x1,y1) (x2,y2) pont között.*
- void `log_text` (const std::string &english, const std::string &magyar)  
*Könnyebb tesztelést biztosít, ha az SFML magyarul írja ki, mit csinált. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.*
- void `warn_text` (const std::string &english, const std::string &magyar, int config\_minimum)  
*Könnyebb tesztelést biztosít, a program magyarul írja ki, mit csinál. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.*

### 7.112.1. Makródefiníciók dokumentációja

#### 7.112.1.1. WITH\_SFML\_RENDER

```
#define WITH_SFML_RENDER
```

Ez arra kell, ha nem headless mode kell a programból.

### 7.112.2. Függvények dokumentációja

#### 7.112.2.1. distance\_to()

```
double distance_to (
    double x1,
    double y1,
    double x2,
    double y2 )
```

Visszaadja a távolságot az (x1,y1) (x2,y2) pont között.

##### Paraméterek

<i>x1</i>	Az 1. pont x koordinátája.
<i>y1</i>	Az 1. pont y koordinátája.
<i>x2</i>	Az 2. pont x koordinátája.
<i>y2</i>	Az 2. pont y koordinátája.

##### Visszatérési érték

A távolság a 2 pont között.

##### Figyelmeztetés

Ez a függvény jelenleg nincs használatban.

#### 7.112.2.2. log\_text()

```
void log_text (
    const std::string & english,
    const std::string & magyar )
```

Könnyebb tesztelést biztosít, ha az SFML magyarul írja ki, mit csinált. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.

### 7.112.2.3. warn\_text()

```
void warn_text (
    const std::string & english,
    const std::string & magyar,
    int config_minimum )
```

Könnyebb tesztelést biztosít, a program magyarul írja ki, mit csinál. Van angol nyelvre is lehetőség és arra is, hogy ne írjon semmit.

## 7.113. src/World.cpp fájlreferencia

```
#include "World.hpp"
```

## 7.114. src/World.d fájlreferencia

## 7.115. src/World.hpp fájlreferencia

A Világ osztály, ami a fő szimulációs elemek tárolásáért felelős.

```
#include "terrain_tiles/Tile.h"
#include <fstream>
#include <string>
#include <sstream>
#include "Utils.h"
#include "TerrainContainer.hpp"
#include "creatures/Living.h"
#include "creatures/humans/Human.h"
#include "creatures/humans/Woodcutter.h"
#include "creatures/humans/Farmer.h"
#include "creatures/humans/Stonemason.h"
#include "creatures/humans/Fisherman.h"
#include "creatures/humans/Builder.h"
#include "creatures/humans/King.h"
#include "creatures/humans/AnglerMiner.h"
#include "creatures/humans/Soldier.h"
#include "creatures/Goat.h"
#include "creatures/hostiles/Crocodile.h"
#include "creatures/hostiles/Bear.h"
#include "creatures/hostiles/KillerRobot.h"
#include <unordered_map>
#include <queue>
#include "world_object/Structure.h"
#include "world_object/ResourceStructure.h"
#include "world_object/BerryBush.h"
#include "world_object/Stone.h"
#include "world_object/Tree.h"
#include "world_object/Iron.h"
#include "world_object/CityCenter.h"
#include "world_object/House.h"
#include <vector>
#include "SoundPlayer.h"
#include "SaveHelpers.h"
#include "../exceptions/FileExceptions.h"
#include "../exceptions/WorldExceptions.h"
#include "HumanResources.h"
```

## Osztályok

- class [WorldBase](#)  
*A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza.*
- class [WorldBaseSerializable](#)  
*A világ osztály bővítése, rendelkezik insertorral és extractorral.*
- class [World](#)  
*A világ osztály leírása.*

### 7.115.1. Részletes leírás

A Világ osztály, ami a fő szimulációs elemek tárolásáért felelős.  
Ez az osztály felelős a szimulációs elemekért, felszabadítja őket, ha kell.

#### Szerző

Funk Gábor

#### Dátum

2025-04-21

## 7.116. src/world\_object/BerryBush.cpp fájlreferencia

```
#include "BerryBush.h"
```

### Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

## 7.117. src/world\_object/BerryBush.d fájlreferencia

## 7.118. src/world\_object/BerryBush.h fájlreferencia

```
#include "ResourceStructure.h"
```

## Osztályok

- class [minerals::BerryBush](#)  
*A bokr osztály leírása. Ételt ad, ha kitermelik.*

### Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

## 7.119. src/world\_object/CityCenter.cpp fájlreferencia

```
#include "CityCenter.h"  
#include "../exceptions/WorldExceptions.h"
```

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.120. src/world\_object/CityCenter.d fájlreferencia

## 7.121. src/world\_object/CityCenter.h fájlreferencia

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

## Osztályok

- class [minerals::CityCenter](#)

*A városközpont osztály leírása. E köré épülnek a házak.*

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.122. src/world\_object/House.cpp fájlreferencia

```
#include "House.h"
```

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.123. src/world\_object/House.d fájlreferencia

## 7.124. src/world\_object/House.h fájlreferencia

```
#include "Structure.h"
#include <string>
#include "../Random_Gen.h"
#include "../HumanResources.h"
```

## Osztályok

- class [minerals::House](#)

*A ház osztály leírása. Szinttől függően idéz embereket.*

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.125. src/world\_object/Iron.cpp fájlreferencia

```
#include "Iron.h"
```

### Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.126. src/world\_object/Iron.d fájlreferencia

## 7.127. src/world\_object/Iron.h fájlreferencia

```
#include "ResourceStructure.h"  
#include "../Random_Gen.h"
```

### Osztályok

- class [minerals::Iron](#)

*A vasérc osztály leírása. Vasat ad, amikor kitermelik.*

### Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.128. src/world\_object/ResourceStructure.cpp fájlreferencia

```
#include "ResourceStructure.h"
```

### Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.129. src/world\_object/ResourceStructure.d fájlreferencia

## 7.130. src/world\_object/ResourceStructure.h fájlreferencia

```
#include "Structure.h"  
#include "../SoundPlayer.h"  
#include <string>
```

### Osztályok

- class [minerals::ResourceStructure](#)

*Az erőforrás struktúra osztály leírása.*

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.131. src/world\_object/Stone.cpp fájlreferencia

```
#include "Stone.h"
```

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.132. src/world\_object/Stone.d fájlreferencia

## 7.133. src/world\_object/Stone.h fájlreferencia

```
#include "ResourceStructure.h"  
#include "../Random_Gen.h"
```

## Osztályok

- class [minerals::Stone](#)

*A kő osztály leírása. követ ad, amikor kitermelik.*

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.134. src/world\_object/Structure.cpp fájlreferencia

```
#include "Structure.h"
```

## Névterek

- [minerals](#)

*Az összes struktúra ebben a névtérben van.*

## 7.135. src/world\_object/Structure.d fájlreferencia

## 7.136. src/world\_object/Structure.h fájlreferencia

```
#include "../Textureable.h"  
#include "../TextureManager.h"  
#include "../GameConfig.h"  
#include "../Shadowable.h"  
#include <string>  
#include <iostream>
```



## Osztályok

- class [minerals::Structure](#)  
*A struktúra osztály leírása.*

## Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

## Enumerációk

- enum class [minerals::MINERAL\\_TYPE](#) : char {  
    [minerals::STONE](#) , [minerals::WOOD](#) , [minerals::IRON](#) , [minerals::FOOD](#) ,  
    [minerals::HOUSING](#) , [minerals::CITY\\_CENTER](#) }

## 7.137. src/world\_object/Tree.cpp fájlreferencia

```
#include "Tree.h"
```

## Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

## 7.138. src/world\_object/Tree.d fájlreferencia

## 7.139. src/world\_object/Tree.h fájlreferencia

```
#include "ResourceStructure.h"  
#include "../Random_Gen.h"
```

## Osztályok

- class [minerals::Tree](#)  
*A fa osztály leírása. Fát ad, ha kitermelik.*

## Névterek

- [minerals](#)  
*Az összes struktúra ebben a névtérben van.*

## 7.140. src/WorldBase.cpp fájlreferencia

```
#include "World.hpp"
```

## 7.141. src/WorldBase.d fájlreferencia

## 7.142. src/WorldBaseSerializable.cpp fájlreferencia

```
#include "World.hpp"
```

### Függvények

- `std::ostream & operator<<` (`std::ostream &os`, [WorldBaseSerialiazble](#) &`w`)
- `std::ifstream & operator>>` (`std::ifstream &in`, [WorldBaseSerialiazble](#) &`w`)

### 7.142.1. Függvények dokumentációja

#### 7.142.1.1. `operator<<()`

```
std::ostream& operator<< (  
    std::ostream & os,  
    WorldBaseSerialiazble & w )
```

#### 7.142.1.2. `operator>>()`

```
std::ifstream& operator>> (  
    std::ifstream & in,  
    WorldBaseSerialiazble & w )
```

## 7.143. src/WorldBaseSerializable.d fájlreferencia

## 7.144. src/YAMLParser.cpp fájlreferencia

```
#include "YAMLParser.h"  
#include "GameConfig.h"  
#include "Utils.h"  
#include <filesystem>
```

## 7.145. src/YAMLParser.d fájlreferencia

## 7.146. src/YAMLParser.h fájlreferencia

```
#include <iostream>  
#include <string>  
#include <unordered_map>  
#include <algorithm>  
#include <fstream>
```

### Osztályok

- class [YAMLParser](#)  
*Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.*

# Tárgymutató

- `_Is_Types< F, T >`, [25](#)
  - `convertable`, [26](#)
  - `f`, [25](#), [26](#)
- `~AnglerMiner`
  - `creature::AnglerMiner`, [27](#)
- `~Bear`
  - `creature::Bear`, [29](#)
- `~Builder`
  - `creature::Builder`, [35](#)
- `~Crocodile`
  - `creature::Crocodile`, [47](#)
- `~EntityBase`
  - `creature::EntityBase`, [51](#)
- `~Farmer`
  - `creature::Farmer`, [62](#)
- `~Fisherman`
  - `creature::Fisherman`, [64](#)
- `~GameManager`
  - `GameManager`, [73](#)
- `~Goat`
  - `creature::Goat`, [76](#)
- `~HostileInterface`
  - `creature::HostileInterface`, [78](#)
- `~Human`
  - `creature::Human`, [86](#)
- `~KillerRobot`
  - `creature::KillerRobot`, [101](#)
- `~King`
  - `creature::King`, [105](#)
- `~Living`
  - `creature::Living`, [107](#)
- `~MusicPlayer`
  - `MusicPlayer`, [119](#)
- `~ResourceStructure`
  - `minerals::ResourceStructure`, [137](#)
- `~Shadowable`
  - `Shadowable`, [144](#)
- `~Soldier`
  - `creature::Soldier`, [150](#)
- `~Sound`
  - `sf::Sound`, [151](#)
- `~SoundSource`
  - `sf::SoundSource`, [154](#)
- `~Sprite`
  - `sf::Sprite`, [155](#)
- `~Stonemason`
  - `creature::Stonemason`, [161](#)
- `~Structure`
  - `minerals::Structure`, [163](#)
- `~TerrainContainer`
  - `TerrainContainer< T >`, [169](#)
- `~Test`
  - `gtest_lite::Test`, [175](#)
- `~Texture`
  - `sf::Texture`, [178](#)
- `~Textureable`
  - `Textureable`, [180](#)
- `~Woodcutter`
  - `creature::Woodcutter`, [194](#)
- `~World`
  - `World`, [196](#)
- `~WorldBase`
  - `WorldBase`, [201](#)
- `~ostreamRedir`
  - `gtest_lite::ostreamRedir`, [122](#)
- `a`
  - `sf::Color`, [44](#)
- `ablocks`
  - `gtest_lite::Test`, [176](#)
- `ADD_FAILURE`
  - `modified_gtest_lite.h`, [227](#)
- `add_resources`
  - `HumanResources`, [90](#)
- `Additive`
  - `sf`, [21](#)
- `almostEQ`
  - `gtest_lite`, [16](#)
- `Alpha`
  - `sf`, [21](#)
- `AnglerMiner`
  - `creature::AnglerMiner`, [27](#)
- `ANIMAL`
  - `creature`, [14](#)
- `animation_speed`
  - `creature::LivingTexture`, [114](#)
- `apply_age`
  - `creature::EntityBase`, [51](#)
- `asSeconds`
  - `sf::ClockTime`, [43](#)
- `ASSERT_`
  - `modified_gtest_lite.h`, [227](#)
- `ASSERT_EQ`
  - `modified_gtest_lite.h`, [227](#)
- `ASSERT_NO_THROW`
  - `modified_gtest_lite.h`, [228](#)
- `ASSERTTHROW`
  - `modified_gtest_lite.h`, [228](#)

astatus  
     gtest\_lite::Test, 175  
 attack\_speed  
     creature::HostileInterface, 80  
 attack\_texture\_path  
     creature::LivingTexture, 114  
 ATTACKING  
     creature, 15  
  
 b  
     sf::Color, 45  
 Bear  
     creature::Bear, 29  
 begin  
     gtest\_lite::Test, 175  
 BerryBush  
     minerals::BerryBush, 32  
 bitsPerPixel  
     sf::VideoMode, 192  
 Black  
     sf::Color, 45  
 BlendAdd  
     sf, 21, 22  
 BlendMode  
     sf, 21  
 blendMode  
     sf::RenderStates, 133  
 Blue  
     sf::Color, 45  
 build\_city\_center\_at  
     WorldBase, 201  
 Builder  
     creature::Builder, 34  
 Button  
     ui::Button, 36  
  
 camp\_needs\_spawn  
     WorldBase, 205  
 check\_aggroed  
     creature::HostileInterface, 79  
     creature::Living, 107  
 CITY\_CENTER  
     minerals, 20  
 CityCenter  
     minerals::CityCenter, 39  
 CityCenterException, 41  
     CityCenterException, 41  
 clear  
     sf::RenderWindow, 134, 135  
     TerrainContainer< T >, 169  
     TextureManager, 182  
     World, 196  
     WorldBaseSerialiazble, 208  
 clear\_at  
     TerrainContainer< T >, 169  
 ClockTime  
     sf::ClockTime, 42, 43  
 close  
     sf::RenderWindow, 135  
  
 Closed  
     sf::Event, 60  
 Color  
     sf::Color, 44  
 combine  
     sf::Transform, 186  
 contains  
     sf::FloatRect, 66  
 convertible  
     \_Is\_Types< F, T >, 26  
 create  
     RoleOption, 139  
     sf::RenderWindow, 135  
 CREATE\_Has\_  
     modified\_gtest\_lite.h, 228  
 CREATE\_Has\_fn\_  
     modified\_gtest\_lite.h, 228  
 creature, 13  
     ANIMAL, 14  
     ATTACKING, 15  
     DEATH, 15  
     DOING\_ITS\_WORK, 15  
     ENTITY\_GENDER, 14  
     ENTITY\_TYPE, 14  
     FACING, 15  
     FEMALE, 14  
     HUMAN, 14  
     IDLE, 15  
     LEFT, 15  
     LIVINGSTATE, 15  
     MALE, 14  
     RIGHT, 15  
     ROBOTIC, 14  
     RUN, 15  
     WALK, 15  
 creature::AnglerMiner, 26  
     ~AnglerMiner, 27  
     AnglerMiner, 27  
     update\_logic, 27  
 creature::Bear, 28  
     ~Bear, 29  
     Bear, 29  
     die, 29  
     draw\_logic, 29  
     get\_type, 30  
     select\_target, 30  
     update\_logic, 30  
 creature::Builder, 34  
     ~Builder, 35  
     Builder, 34  
     update\_logic, 35  
 creature::Crocodile, 46  
     ~Crocodile, 47  
     Crocodile, 47  
     die, 47  
     draw\_logic, 47  
     get\_type, 48  
     select\_target, 48

- update\_logic, 48
- creature::EntityBase, 49
  - ~EntityBase, 51
  - apply\_age, 51
  - death\_timer, 55
  - die, 52
  - facing, 55
  - gender, 55
  - get\_death\_timer, 52
  - get\_gender, 52
  - get\_save\_name, 52
  - get\_state, 52
  - get\_type, 53
  - health, 56
  - hit\_timer, 56
  - inner\_timer, 56
  - max\_age, 56
  - posx, 56
  - posy, 56
  - run\_speed\_modifier, 57
  - save\_name, 57
  - set\_attack\_texture, 53
  - set\_death\_texture, 53
  - set\_health, 54
  - set\_idle\_texture, 54
  - set\_run\_texture, 54
  - set\_save\_name, 54
  - set\_state, 55
  - set\_walk\_texture, 55
  - speed, 57
  - state, 57
  - texture\_data, 57
- creature::Farmer, 61
  - ~Farmer, 62
  - Farmer, 62
  - update\_logic, 62
- creature::Fisherman, 63
  - ~Fisherman, 64
  - Fisherman, 64
  - fishing, 65
  - try\_fishing, 64
  - update\_logic, 65
- creature::Goat, 75
  - ~Goat, 76
  - die, 76
  - draw\_logic, 76
  - get\_type, 76
  - Goat, 75
  - update\_logic, 77
- creature::HostileInterface, 77
  - ~HostileInterface, 78
  - attack\_speed, 80
  - check\_aggroed, 79
  - damage, 81
  - goal, 81
  - hostile\_run, 79
  - hostile\_walk, 79
  - retarget, 79
  - select\_target, 80
  - set\_hostile\_config, 80
  - target, 81
  - try\_attack, 80
- creature::Human, 84
  - ~Human, 86
  - die, 86
  - draw\_logic, 86
  - get\_profession\_string, 87
  - get\_type, 87
  - goal, 88
  - Human, 85
  - humanoid\_run, 87
  - humanoid\_walk, 87
  - initialize, 87
  - needs\_promotion, 89
  - needs\_to\_be\_royal, 89
  - profession, 89
  - select\_texture, 88
  - update\_logic, 88
- creature::KillerRobot, 100
  - ~KillerRobot, 101
  - die, 102
  - draw\_logic, 102
  - get\_type, 102
  - KillerRobot, 101
  - select\_target, 103
  - update\_logic, 103
- creature::King, 103
  - ~King, 105
  - King, 104
  - update\_logic, 105
- creature::Living, 105
  - ~Living, 107
  - check\_aggroed, 107
  - damage, 108
  - damaged\_by, 113
  - draw, 108
  - draw\_logic, 108
  - get\_width, 109
  - init\_spritesheet\_data, 109
  - look\_left, 109
  - look\_right, 110
  - MAX\_CREATURE\_SIZE, 113
  - needs\_drawn, 110
  - retarget, 110
  - set\_state, 110
  - setPosition, 111
  - setTexture, 111
  - setTheShadow, 111
  - shadow\_logic, 112
  - update\_logic, 112
  - update\_spritesheet, 112
- creature::LivingTexture, 113
  - animation\_speed, 114
  - attack\_texture\_path, 114
  - current\_animation\_time, 114
  - death\_texture, 114

- frame\_count, 114
- idle\_texture\_path, 114
- run\_texture\_path, 115
- walk\_texture\_path, 115
- creature::Soldier, 149
  - ~Soldier, 150
  - Soldier, 149
  - update\_logic, 150
- creature::Stonemason, 159
  - ~Stonemason, 161
  - mining\_iron, 162
  - Stonemason, 160
  - try\_mine, 161
  - update\_logic, 161
- creature::Woodcutter, 193
  - ~Woodcutter, 194
  - update\_logic, 194
  - Woodcutter, 193
- creatures, 15
- Crocodile
  - creature::Crocodile, 47
- current\_animation\_time
  - creature::LivingTexture, 114
- current\_city\_center
  - WorldBase, 206
- damage
  - creature::HostileInterface, 81
  - creature::Living, 108
- damaged\_by
  - creature::Living, 113
- day\_length
  - GameConfig, 71
- DEATH
  - creature, 15
- death\_texture
  - creature::LivingTexture, 114
- death\_timer
  - creature::EntityBase, 55
- deleteFile
  - SaveManager, 142
- die
  - creature::Bear, 29
  - creature::Crocodile, 47
  - creature::EntityBase, 52
  - creature::Goat, 76
  - creature::Human, 86
  - creature::KillerRobot, 102
- display
  - sf::RenderWindow, 135
- distance\_to
  - Utils.cpp, 243
  - Utils.h, 245
- DOING\_ITS\_WORK
  - creature, 15
- Down
  - sf::Keyboard, 99
- draw
  - creature::Living, 108
  - minerals::Structure, 163
  - PostProcessor, 123
  - Profession, 127
  - sf::RenderWindow, 135
  - sf::Sprite, 155
  - TerrainContainer< T >, 170
  - Textureable, 180
  - tiles::Tile, 184
  - ui::Button, 37
  - World, 196
- draw\_buttons
  - GameManager, 73
- draw\_logic
  - creature::Bear, 29
  - creature::Crocodile, 47
  - creature::Goat, 76
  - creature::Human, 86
  - creature::KillerRobot, 102
  - creature::Living, 108
  - minerals::Structure, 164
- drawShadow
  - Shadowable, 144
- elapsed\_time
  - WorldBaseSerialiazble, 209
- END
  - modified\_gtest\_lite.h, 229
- end
  - gtest\_lite::Test, 175
- ENDM
  - modified\_gtest\_lite.h, 229
- ENDMsg
  - modified\_gtest\_lite.h, 229
- ENGLISH
  - GameConfig.h, 235
- entities
  - WorldBase, 206
- ENTITY\_GENDER
  - creature, 14
- ENTITY\_TYPE
  - creature, 14
- EntityPlacer, 58
  - EntityPlacer, 58
  - reset\_mouse, 58
  - select\_entity, 59
  - setup\_factory, 59
  - spacePreviouslyPressed, 59
  - toggle\_placing, 59
  - try\_place\_entity, 59
- eq
  - gtest\_lite, 16
- eqstr
  - gtest\_lite, 16
- eqstrcase
  - gtest\_lite, 17
- EType
  - sf::Event, 60
- Event
  - sf::Event, 60

- expect
  - gtest\_lite::Test, [175](#)
- EXPECT\_
  - gtest\_lite, [17](#)
- EXPECT\_ANY\_THROW
  - modified\_gtest\_lite.h, [229](#)
- EXPECT\_DOUBLE\_EQ
  - modified\_gtest\_lite.h, [229](#)
- EXPECT\_ENVCASEEQ
  - modified\_gtest\_lite.h, [229](#)
- EXPECT\_ENVEQ
  - modified\_gtest\_lite.h, [229](#)
- EXPECT\_EQ
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_FALSE
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_FLOAT\_EQ
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_GE
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_GT
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_LE
  - modified\_gtest\_lite.h, [230](#)
- EXPECT\_LT
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_NE
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_NO\_THROW
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_STRCASEEQ
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_STRCASENE
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_STREQ
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_STRNE
  - modified\_gtest\_lite.h, [231](#)
- EXPECT\_THROW
  - modified\_gtest\_lite.h, [232](#)
- EXPECT\_THROW\_THROW
  - modified\_gtest\_lite.h, [232](#)
- EXPECT\_TRUE
  - modified\_gtest\_lite.h, [232](#)
- EXPECTSTR
  - gtest\_lite, [17](#)
- EXPECTTHROW
  - modified\_gtest\_lite.h, [232](#)
- f
  - \_Is\_Types< F, T >, [25](#), [26](#)
- FACING
  - creature, [15](#)
- facing
  - creature::EntityBase, [55](#)
- FAIL
  - modified\_gtest\_lite.h, [232](#)
- fail
  - gtest\_lite::Test, [176](#)
- failed
  - gtest\_lite::Test, [176](#)
- Farmer
  - creature::Farmer, [62](#)
- FEMALE
  - creature, [14](#)
- file\_exists\_at\_path
  - sf, [22](#)
- Fisherman
  - creature::Fisherman, [64](#)
- fishing
  - creature::Fisherman, [65](#)
- FloatRect
  - sf::FloatRect, [66](#)
- FOOD
  - minerals, [20](#)
- frame\_count
  - creature::LivingTexture, [114](#)
- g
  - sf::Color, [45](#)
- game\_loop
  - GameManager, [73](#)
- GameConfig, [67](#)
  - day\_length, [71](#)
  - GameConfig, [68](#)
  - get\_config\_level, [69](#)
  - get\_hostiles\_count, [69](#)
  - get\_instance, [69](#)
  - get\_lang, [69](#)
  - get\_max\_spawn\_tries, [69](#)
  - get\_resource\_scarcity, [69](#)
  - get\_screen\_height, [70](#)
  - get\_screen\_width, [70](#)
  - get\_sfml\_lang, [70](#)
  - get\_target\_fps, [70](#)
  - get\_world\_size, [70](#)
  - is\_chromatic\_aberration, [70](#)
  - is\_noise, [70](#)
  - operator=, [71](#)
  - read\_from\_config\_file, [71](#)
  - set\_config\_level, [71](#)
  - set\_world\_size, [71](#)
- GameConfig.cpp
  - trim, [235](#)
- GameConfig.h
  - ENGLISH, [235](#)
  - Language, [235](#)
  - MAGYAR, [235](#)
  - NONE, [235](#)
- GameManager, [72](#)
  - ~GameManager, [73](#)
  - draw\_buttons, [73](#)
  - game\_loop, [73](#)
  - GameManager, [73](#)
  - get\_elapsed\_time, [73](#)
  - handle\_unit\_placement, [73](#)
  - is\_valid, [74](#)
  - run, [74](#)

- setup\_buttons, [74](#)
  - simulate\_tick, [74](#)
  - update\_buttons, [74](#)
- ge
  - gtest\_lite, [18](#)
- gender
  - creature::EntityBase, [55](#)
- generate\_world
  - TerrainContainer< T >, [170](#)
- get\_border\_height
  - World, [197](#)
- get\_border\_width
  - World, [197](#)
- get\_config\_level
  - GameConfig, [69](#)
- get\_count\_from
  - HumanResources, [90](#)
- get\_current\_city\_center
  - WorldBase, [201](#)
- get\_death\_timer
  - creature::EntityBase, [52](#)
- get\_elapsed\_time
  - GameManager, [73](#)
- get\_excluded\_entities
  - WorldBase, [201](#)
- get\_gender
  - creature::EntityBase, [52](#)
- get\_harvested
  - minerals::ResourceStructure, [138](#)
- get\_height
  - TerrainContainer< T >, [170](#)
- get\_height\_offset
  - Shadowable, [145](#)
- get\_hostiles\_count
  - GameConfig, [69](#)
- get\_instance
  - GameConfig, [69](#)
  - RandomGenerator, [129](#)
- get\_lang
  - GameConfig, [69](#)
- get\_level
  - minerals::House, [82](#)
- get\_max\_spawn\_tries
  - GameConfig, [69](#)
- get\_needs\_remove
  - minerals::ResourceStructure, [138](#)
- get\_position\_nearby\_town
  - WorldBase, [202](#)
- get\_profession\_string
  - creature::Human, [87](#)
- get\_random\_house\_pos
  - WorldBase, [202](#)
- get\_random\_int
  - RandomGenerator, [129](#)
- get\_random\_suitable\_position
  - WorldBase, [202](#)
- get\_resource\_scarcity
  - GameConfig, [69](#)
- get\_resources
  - WorldBase, [203](#)
- get\_roles
  - SaveHelper, [140](#)
- get\_save\_name
  - creature::EntityBase, [52](#)
- get\_screen\_height
  - GameConfig, [70](#)
- get\_screen\_width
  - GameConfig, [70](#)
- get\_seed
  - TerrainContainer< T >, [170](#)
- get\_settlement\_age
  - minerals::CityCenter, [40](#)
- get\_sfml\_lang
  - GameConfig, [70](#)
- get\_shadow\_strength
  - Shadowable, [145](#)
- get\_skew\_offset
  - Shadowable, [145](#)
- get\_state
  - creature::EntityBase, [52](#)
- get\_structure\_type
  - WorldBase, [203](#)
- get\_target\_fps
  - GameConfig, [70](#)
- get\_type
  - creature::Bear, [30](#)
  - creature::Crocodile, [48](#)
  - creature::EntityBase, [53](#)
  - creature::Goat, [76](#)
  - creature::Human, [87](#)
  - creature::KillerRobot, [102](#)
  - minerals::BerryBush, [32](#)
  - minerals::CityCenter, [40](#)
  - minerals::House, [82](#)
  - minerals::Iron, [98](#)
  - minerals::Stone, [159](#)
  - minerals::Structure, [164](#)
  - minerals::Tree, [188](#)
  - tiles::Tile, [184](#)
- get\_value\_of\_key
  - YAMLParse, [210](#)
- get\_width
  - creature::Living, [109](#)
  - TerrainContainer< T >, [171](#)
- get\_world\_size
  - GameConfig, [70](#)
- getCreatureFactory
  - SaveHelper, [140](#)
- getDesktopMode
  - sf::VideoMode, [192](#)
- getElapsedTime
  - sf::Clock, [42](#)
- getGlobalBounds
  - sf::Sprite, [155](#), [156](#)
- getHumanFactory
  - SaveHelper, [141](#)



- getInstance
  - TextureManager, 182
- getLocalBounds
  - sf::Sprite, 156
- getPosition
  - sf::Mouse, 116
  - sf::Sprite, 156
- getResourceFactory
  - SaveHelper, 141
- getSize
  - sf::Texture, 178
- getStatus
  - sf::Music, 117
- getTest
  - gtest\_lite::Test, 176
- getTexture
  - sf::Sprite, 156
  - TextureManager, 182
- getTileAt
  - WorldBase, 203
- goal
  - creature::HostileInterface, 81
  - creature::Human, 88
- Goat
  - creature::Goat, 75
- GRASS
  - tiles, 23
- Green
  - sf::Color, 45
- gt
  - gtest\_lite, 18
- GTEND
  - modified\_gtest\_lite.h, 232
- gtest\_lite, 15
  - almostEQ, 16
  - eq, 16
  - eqstr, 16
  - eqstrcase, 17
  - EXPECT\_, 17
  - EXPECTSTR, 17
  - ge, 18
  - gt, 18
  - le, 18
  - lt, 18
  - ne, 18
  - nestr, 19
- gtest\_lite::ostreamRedir, 122
  - ~ostreamRedir, 122
  - ostreamRedir, 122
- gtest\_lite::Test, 174
  - ~Test, 175
  - ablocks, 176
  - astatus, 175
  - begin, 175
  - end, 175
  - expect, 175
  - fail, 176
  - failed, 176
  - getTest, 176
  - name, 176
  - null, 177
  - os, 177
  - status, 177
  - sum, 177
  - tmp, 177
- GTINIT
  - modified\_gtest\_lite.h, 233
- handle\_unit\_placement
  - GameManager, 73
- harvest
  - minerals::BerryBush, 32
  - minerals::ResourceStructure, 138
- harvested
  - minerals::ResourceStructure, 138
- hasMember
  - modified\_gtest\_lite.h, 233
- health
  - creature::EntityBase, 56
- height
  - sf::Bound, 33
  - sf::FloatRect, 66
  - sf::IntRect, 96
  - sf::VideoMode, 192
- height\_offset
  - Shadowable, 147
- hit\_timer
  - creature::EntityBase, 56
- hostile\_run
  - creature::HostileInterface, 79
- hostile\_walk
  - creature::HostileInterface, 79
- House
  - minerals::House, 82
- houses
  - WorldBase, 206
- HOUSING
  - minerals, 20
- HUMAN
  - creature, 14
- Human
  - creature::Human, 85
- humanoid\_run
  - creature::Human, 87
- humanoid\_walk
  - creature::Human, 87
- HumanResources, 89
  - add\_resources, 90
  - get\_count\_from, 90
  - is\_there\_enough\_resource, 90
  - remove\_resources, 91
  - set\_resources, 91
- humans
  - WorldBase, 206
- IDLE
  - creature, 15

- idle\_texture\_path
  - creature::LivingTexture, 114
- ImportInvalidEntityException, 92
  - ImportInvalidEntityException, 92
- ImportInvalidHousingLevelException, 93
  - ImportInvalidHousingLevelException, 93
- ImportInvalidHumanProfessionException, 93
  - ImportInvalidHumanProfessionException, 94
- ImportInvalidResourceException, 94
  - ImportInvalidResourceException, 95
- increment
  - sf::ClockTime, 43
- init
  - tiles::Tile, 184
- init\_spritesheet\_data
  - creature::Living, 109
- initialize
  - creature::Human, 87
- inner\_timer
  - creature::EntityBase, 56
  - minerals::ResourceStructure, 139
- IntRect
  - sf::IntRect, 95
- Invalid
  - sf::Event, 60
- InvalidBorderSizeException, 96
  - InvalidBorderSizeException, 97
- IRON
  - minerals, 20
- Iron
  - minerals::Iron, 98
- is\_chromatic\_aberration
  - GameConfig, 70
- is\_noise
  - GameConfig, 70
- is\_on\_screen
  - TerrainContainer< T >, 171
- is\_there\_enough\_resource
  - HumanResources, 90
- is\_there\_room\_for\_housing
  - minerals::CityCenter, 40
- is\_valid
  - GameManager, 74
- is\_valid\_coordinate
  - TerrainContainer< T >, 171
- isButtonPressed
  - sf::Mouse, 116
- isKeyPressed
  - sf::Keyboard, 100
- isOpen
  - sf::RenderWindow, 136
- isValid
  - sf::VideoMode, 192
- Key
  - sf::Keyboard, 99
- KillerRobot
  - creature::KillerRobot, 101
- King
  - creature::King, 104
- Language
  - GameConfig.h, 235
- le
  - gtest\_lite, 18
- LEFT
  - creature, 15
- Left
  - sf::Keyboard, 99
  - sf::Mouse, 116
- left
  - sf::FloatRect, 67
  - sf::IntRect, 96
- LIVINGSTATE
  - creature, 15
- load\_music
  - MusicPlayer, 120
- load\_profession
  - Profession, 127
- load\_sound
  - SoundPlayer, 152
- loadFile
  - SaveManager, 142
- loadFromFile
  - sf::SoundBuffer, 152
  - sf::Texture, 178
- loadTexture
  - TextureManager, 182
- log\_text
  - Utils.cpp, 244
  - Utils.h, 245
- look\_left
  - creature::Living, 109
- look\_right
  - creature::Living, 110
- It
  - gtest\_lite, 18
- MAGYAR
  - GameConfig.h, 235
- main
  - main.cpp, 237
- main.cpp
  - main, 237
- MALE
  - creature, 14
- matrix
  - sf::Transform, 187
- max\_age
  - creature::EntityBase, 56
- MAX\_CREATURE\_SIZE
  - creature::Living, 113
- MAX\_OBJECT\_SIZE
  - minerals::Structure, 166
  - WorldBase, 206
- mineral\_to\_string
  - minerals, 20
- MINERAL\_TYPE

- minerals, [20](#)
- minerals, [19](#)
  - CITY\_CENTER, [20](#)
  - FOOD, [20](#)
  - HOUSING, [20](#)
  - IRON, [20](#)
  - mineral\_to\_string, [20](#)
  - MINERAL\_TYPE, [20](#)
  - STONE, [20](#)
  - WOOD, [20](#)
- minerals::BerryBush, [31](#)
  - BerryBush, [32](#)
  - get\_type, [32](#)
  - harvest, [32](#)
  - play\_destroy\_sound, [32](#)
  - update\_logic, [32](#)
- minerals::CityCenter, [39](#)
  - CityCenter, [39](#)
  - get\_settlement\_age, [40](#)
  - get\_type, [40](#)
  - is\_there\_room\_for\_housing, [40](#)
  - register\_new\_house, [40](#)
  - update\_logic, [40](#)
- minerals::House, [81](#)
  - get\_level, [82](#)
  - get\_type, [82](#)
  - House, [82](#)
  - set\_level, [83](#)
  - try\_upgrade, [83](#)
  - update\_logic, [83](#)
  - upgrade\_house, [83](#)
- minerals::Iron, [97](#)
  - get\_type, [98](#)
  - Iron, [98](#)
  - play\_destroy\_sound, [98](#)
  - update\_logic, [98](#)
- minerals::ResourceStructure, [136](#)
  - ~ResourceStructure, [137](#)
  - get\_harvested, [138](#)
  - get\_needs\_remove, [138](#)
  - harvest, [138](#)
  - harvested, [138](#)
  - inner\_timer, [139](#)
  - needs\_remove, [139](#)
  - play\_destroy\_sound, [138](#)
  - ResourceStructure, [137](#)
- minerals::Stone, [158](#)
  - get\_type, [159](#)
  - play\_destroy\_sound, [159](#)
  - Stone, [158](#)
  - update\_logic, [159](#)
- minerals::Structure, [162](#)
  - ~Structure, [163](#)
  - draw, [163](#)
  - draw\_logic, [164](#)
  - get\_type, [164](#)
  - MAX\_OBJECT\_SIZE, [166](#)
  - needs\_drawn, [164](#)
  - posx, [166](#)
  - posy, [166](#)
  - setPosition, [164](#)
  - setTexture, [165](#)
  - Structure, [163](#)
  - update\_logic, [165](#)
- minerals::Tree, [187](#)
  - get\_type, [188](#)
  - play\_destroy\_sound, [188](#)
  - Tree, [188](#)
  - update\_logic, [188](#)
- mining\_iron
  - creature::Stonemason, [162](#)
- modified\_gtest\_lite.h
  - ADD\_FAILURE, [227](#)
  - ASSERT\_, [227](#)
  - ASSERT\_EQ, [227](#)
  - ASSERT\_NO\_THROW, [228](#)
  - ASSERTTHROW, [228](#)
  - CREATE\_Has\_, [228](#)
  - CREATE\_Has\_fn\_, [228](#)
  - END, [229](#)
  - ENDM, [229](#)
  - ENDMsg, [229](#)
  - EXPECT\_ANY\_THROW, [229](#)
  - EXPECT\_DOUBLE\_EQ, [229](#)
  - EXPECT\_ENVCASEEQ, [229](#)
  - EXPECT\_ENVEQ, [229](#)
  - EXPECT\_EQ, [230](#)
  - EXPECT\_FALSE, [230](#)
  - EXPECT\_FLOAT\_EQ, [230](#)
  - EXPECT\_GE, [230](#)
  - EXPECT\_GT, [230](#)
  - EXPECT\_LE, [230](#)
  - EXPECT\_LT, [231](#)
  - EXPECT\_NE, [231](#)
  - EXPECT\_NO\_THROW, [231](#)
  - EXPECT\_STRCASEEQ, [231](#)
  - EXPECT\_STRCASENE, [231](#)
  - EXPECT\_STREQ, [231](#)
  - EXPECT\_STRNE, [231](#)
  - EXPECT\_THROW, [232](#)
  - EXPECT\_THROW\_THROW, [232](#)
  - EXPECT\_TRUE, [232](#)
  - EXPECTTHROW, [232](#)
  - FAIL, [232](#)
  - GTEND, [232](#)
  - GTINIT, [233](#)
  - hasMember, [233](#)
  - SUCCEED, [233](#)
  - TEST, [233](#)
- MOUNTAIN
  - tiles, [23](#)
- Mousedowntype
  - sf::Mouse, [115](#)
- Multiply
  - sf, [21](#)
- Music

- sf::Music, 117
- MusicLoadException, 118
  - MusicLoadException, 118
- MusicPlayer, 119
  - ~MusicPlayer, 119
  - load\_music, 120
  - MusicPlayer, 119
  - set\_volume, 120
  - toggle\_music, 120
- name
  - gtest\_lite::Test, 176
- ne
  - gtest\_lite, 18
- needs\_drawn
  - creature::Living, 110
  - minerals::Structure, 164
- needs\_promotion
  - creature::Human, 89
- needs\_remove
  - minerals::ResourceStructure, 139
- needs\_to\_be\_royal
  - creature::Human, 89
- nestr
  - gtest\_lite, 19
- NoEvent
  - sf::Event, 60
- NONE
  - GameConfig.h, 235
- None
  - sf, 21
- null
  - gtest\_lite::Test, 177
- Num0
  - sf::Keyboard, 99
- Num1
  - sf::Keyboard, 99
- Num2
  - sf::Keyboard, 99
- Num3
  - sf::Keyboard, 99
- Num4
  - sf::Keyboard, 99
- Num5
  - sf::Keyboard, 99
- Num6
  - sf::Keyboard, 99
- Num7
  - sf::Keyboard, 99
- Num8
  - sf::Keyboard, 99
- Num9
  - sf::Keyboard, 99
- ObjectRegistry, 121
  - register\_type, 121
  - spawn, 121
- onClick
  - ui::Button, 37
- openFromFile
  - sf::Music, 117
- operator<<
  - WorldBaseSerializable, 208
  - WorldBaseSerializable.cpp, 252
- operator>>
  - WorldBaseSerializable, 209
  - WorldBaseSerializable.cpp, 252
- operator=
  - GameConfig, 71
  - RandomGenerator, 130
  - sf::Texture, 179
- operator[]
  - TerrainContainer< T >, 172
- os
  - gtest\_lite::Test, 177
- ostreamRedir
  - gtest\_lite::ostreamRedir, 122
- parse\_file
  - YAMLParse, 210
- Paused
  - sf::SoundSource, 154
- play
  - sf::Music, 117
  - sf::Sound, 151
- play\_destroy\_sound
  - minerals::BerryBush, 32
  - minerals::Iron, 98
  - minerals::ResourceStructure, 138
  - minerals::Stone, 159
  - minerals::Tree, 188
- play\_sound
  - SoundPlayer, 153
- Playing
  - sf::SoundSource, 154
- pollEvent
  - sf::RenderWindow, 136
- populate\_world
  - World, 197
- position
  - sf::RectangleShape, 132
- PostProcessor, 122
  - draw, 123
  - PostProcessor, 123
  - setColorOverlay, 124
  - setRenderSize, 124
  - setTextureFor, 124
  - toggle\_chromatic\_aberration, 125
  - toggle\_noise, 125
  - toggle\_vignette, 125
- posx
  - creature::EntityBase, 56
  - minerals::Structure, 166
- posy
  - creature::EntityBase, 56
  - minerals::Structure, 166
- Profession, 126
  - draw, 127

- load\_profession, [127](#)
- Profession, [126](#)
- setPosition, [127](#)
- setTexture, [128](#)
- to\_string, [128](#)
- profession
  - creature::Human, [89](#)
- r
  - sf::Color, [45](#)
- RandomGenerator, [128](#)
  - get\_instance, [129](#)
  - get\_random\_int, [129](#)
  - operator=, [130](#)
  - RandomGenerator, [129](#)
- read\_from\_config\_file
  - GameConfig, [71](#)
- ReadSaveFileFail, [130](#)
  - ReadSaveFileFail, [131](#)
- Red
  - sf::Color, [45](#)
- regenerate
  - World, [197](#)
- register\_new\_house
  - minerals::CityCenter, [40](#)
- register\_type
  - ObjectRegistry, [121](#)
- reinitialize\_self
  - WorldBaseSerialiazble, [208](#)
- remove\_resources
  - HumanResources, [91](#)
- remove\_structure\_at
  - WorldBase, [204](#)
- RenderStates
  - sf::RenderStates, [132](#)
- RenderWindow
  - sf::RenderWindow, [134](#)
- requirements
  - RoleOption, [140](#)
- reset
  - sf::ClockTime, [43](#)
- reset\_mouse
  - EntityPlacer, [58](#)
- resize
  - TerrainContainer< T >, [172](#)
- ResourceStructure
  - minerals::ResourceStructure, [137](#)
- restart
  - sf::Clock, [42](#)
- retarget
  - creature::HostileInterface, [79](#)
  - creature::Living, [110](#)
- RIGHT
  - creature, [15](#)
- Right
  - sf::Keyboard, [99](#)
  - sf::Mouse, [116](#)
- ROBOTIC
  - creature, [14](#)
- RoleOption, [139](#)
  - create, [139](#)
  - requirements, [140](#)
- RUN
  - creature, [15](#)
- run
  - GameManager, [74](#)
- run\_speed\_modifier
  - creature::EntityBase, [57](#)
- run\_texture\_path
  - creature::LivingTexture, [115](#)
- save\_name
  - creature::EntityBase, [57](#)
- saved\_size
  - WorldBaseSerialiazble, [209](#)
- saveFile
  - SaveManager, [142](#)
- SaveHelper, [140](#)
  - get\_roles, [140](#)
  - getCreatureFactory, [140](#)
  - getHumanFactory, [141](#)
  - getResourceFactory, [141](#)
  - trim\_brackets, [141](#)
- SaveManager, [141](#)
  - deleteFile, [142](#)
  - loadFile, [142](#)
  - saveFile, [142](#)
  - SaveManager, [142](#)
- select\_entity
  - EntityPlacer, [59](#)
- select\_target
  - creature::Bear, [30](#)
  - creature::Crocodile, [48](#)
  - creature::HostileInterface, [80](#)
  - creature::KillerRobot, [103](#)
- select\_texture
  - creature::Human, [88](#)
- set\_attack\_texture
  - creature::EntityBase, [53](#)
- set\_border\_height
  - World, [197](#)
- set\_border\_width
  - World, [198](#)
- set\_config\_level
  - GameConfig, [71](#)
- set\_death\_texture
  - creature::EntityBase, [53](#)
- set\_health
  - creature::EntityBase, [54](#)
- set\_height\_offset
  - Shadowable, [145](#)
- set\_hostile\_config
  - creature::HostileInterface, [80](#)
- set\_idle\_texture
  - creature::EntityBase, [54](#)
- set\_level
  - minerals::House, [83](#)
- set\_resources

- HumanResources, 91
- set\_run\_texture
  - creature::EntityBase, 54
- set\_save\_name
  - creature::EntityBase, 54
- set\_seed
  - TerrainContainer< T >, 173
- set\_shadow\_strength
  - Shadowable, 146
- set\_skew\_offset
  - Shadowable, 146
- set\_state
  - creature::EntityBase, 55
  - creature::Living, 110
- set\_volume
  - MusicPlayer, 120
- set\_walk\_texture
  - creature::EntityBase, 55
- set\_world\_size
  - GameConfig, 71
- setBlendMode
  - sf::RenderStates, 133
- setBuffer
  - sf::Sound, 151
- setCallback
  - ui::Button, 37
- setColor
  - sf::Sprite, 156
- setColorOverlay
  - PostProcessor, 124
- setFillColor
  - sf::RectangleShape, 131
- setFramerateLimit
  - sf::RenderWindow, 136
- setLoop
  - sf::Music, 117
- setOrigin
  - sf::Sprite, 156
- setPosition
  - creature::Living, 111
  - minerals::Structure, 164
  - Profession, 127
  - sf::RectangleShape, 131
  - sf::Sprite, 156
  - Textureable, 180
  - tiles::Tile, 185
  - ui::Button, 37
- setRenderSize
  - PostProcessor, 124
- setRotation
  - sf::Sprite, 157
- setScale
  - sf::Sprite, 157
- setShadow
  - Shadowable, 146
- setShadowDayNightCycle
  - Shadowable, 146
- setShadowPosition
  - Shadowable, 147
- setShadowTexture
  - Shadowable, 147
- setSize
  - sf::RectangleShape, 131
- setTexture
  - creature::Living, 111
  - minerals::Structure, 165
  - Profession, 128
  - sf::Sprite, 157
  - Textureable, 181
  - tiles::Tile, 185
  - ui::Button, 38
- setTextureFor
  - PostProcessor, 124
- setTextureRect
  - sf::Sprite, 157
- setTheShadow
  - creature::Living, 111
- setTransform
  - sf::RenderStates, 133
- setup\_buttons
  - GameManager, 74
- setup\_factory
  - EntityPlacer, 59
- setVolume
  - sf::Music, 117
- sf, 20
  - Additive, 21
  - Alpha, 21
  - BlendAdd, 21, 22
  - BlendMode, 21
  - file\_exists\_at\_path, 22
  - Multiply, 21
  - None, 21
  - to\_string, 22
- sf::Bound, 33
  - height, 33
  - width, 33
- sf::Clock, 42
  - getElapsedTime, 42
  - restart, 42
- sf::ClockTime, 42
  - asSeconds, 43
  - ClockTime, 42, 43
  - increment, 43
  - reset, 43
- sf::Color, 43
  - a, 44
  - b, 45
  - Black, 45
  - Blue, 45
  - Color, 44
  - g, 45
  - Green, 45
  - r, 45
  - Red, 45
  - Transparent, 45

- White, [46](#)
- sf::Event, [60](#)
  - Closed, [60](#)
  - EType, [60](#)
  - Event, [60](#)
  - Invalid, [60](#)
  - NoEvent, [60](#)
  - type, [61](#)
- sf::FloatRect, [65](#)
  - contains, [66](#)
  - FloatRect, [66](#)
  - height, [66](#)
  - left, [67](#)
  - top, [67](#)
  - width, [67](#)
- sf::IntRect, [95](#)
  - height, [96](#)
  - IntRect, [95](#)
  - left, [96](#)
  - top, [96](#)
  - width, [96](#)
- sf::Keyboard, [99](#)
  - Down, [99](#)
  - isKeyPressed, [100](#)
  - Key, [99](#)
  - Left, [99](#)
  - Num0, [99](#)
  - Num1, [99](#)
  - Num2, [99](#)
  - Num3, [99](#)
  - Num4, [99](#)
  - Num5, [99](#)
  - Num6, [99](#)
  - Num7, [99](#)
  - Num8, [99](#)
  - Num9, [99](#)
  - Right, [99](#)
  - simulate\_key\_press, [100](#)
  - simulate\_key\_release, [100](#)
  - Space, [99](#)
  - Up, [99](#)
- sf::Mouse, [115](#)
  - getPosition, [116](#)
  - isButtonPressed, [116](#)
  - Left, [116](#)
  - Mousedowntype, [115](#)
  - Right, [116](#)
  - simulate\_key\_press, [116](#)
  - simulate\_key\_release, [116](#)
- sf::Music, [116](#)
  - getStatus, [117](#)
  - Music, [117](#)
  - openFromFile, [117](#)
  - play, [117](#)
  - setLoop, [117](#)
  - setVolume, [117](#)
  - stop, [118](#)
- sf::RectangleShape, [131](#)
  - position, [132](#)
  - setFillColor, [131](#)
  - setPosition, [131](#)
  - setSize, [131](#)
- sf::RenderStates, [132](#)
  - blendMode, [133](#)
  - RenderStates, [132](#)
  - setBlendMode, [133](#)
  - setTransform, [133](#)
  - transform, [133](#)
- sf::RenderWindow, [133](#)
  - clear, [134](#), [135](#)
  - close, [135](#)
  - create, [135](#)
  - display, [135](#)
  - draw, [135](#)
  - isOpen, [136](#)
  - pollEvent, [136](#)
  - RenderWindow, [134](#)
  - setFramerateLimit, [136](#)
- sf::Sound, [150](#)
  - ~Sound, [151](#)
  - play, [151](#)
  - setBuffer, [151](#)
  - stop, [151](#)
- sf::SoundBuffer, [151](#)
  - loadFromFile, [152](#)
- sf::SoundSource, [153](#)
  - ~SoundSource, [154](#)
  - Paused, [154](#)
  - Playing, [154](#)
  - SoundSource, [154](#)
  - SoundSourceType, [154](#)
  - Stopped, [154](#)
  - type, [154](#)
- sf::Sprite, [155](#)
  - ~Sprite, [155](#)
  - draw, [155](#)
  - getGlobalBounds, [155](#), [156](#)
  - getLocalBounds, [156](#)
  - getPosition, [156](#)
  - getTexture, [156](#)
  - setColor, [156](#)
  - setOrigin, [156](#)
  - setPosition, [156](#)
  - setRotation, [157](#)
  - setScale, [157](#)
  - setTexture, [157](#)
  - setTextureRect, [157](#)
  - Sprite, [155](#)
- sf::Texture, [178](#)
  - ~Texture, [178](#)
  - getSize, [178](#)
  - loadFromFile, [178](#)
  - operator=, [179](#)
  - Texture, [178](#)
- sf::Transform, [185](#)
  - combine, [186](#)

- matrix, [187](#)
- Transform, [186](#)
- transformPoint, [186](#)
- translate, [187](#)
- sf::Vector2f, [189](#)
  - Vector2f, [189](#)
  - x, [189](#)
  - y, [190](#)
- sf::Vector2i, [190](#)
  - Vector2i, [190](#)
  - x, [191](#)
  - y, [191](#)
- sf::VideoMode, [191](#)
  - bitsPerPixel, [192](#)
  - getDesktopMode, [192](#)
  - height, [192](#)
  - isValid, [192](#)
  - VideoMode, [191](#)
  - width, [192](#)
- shadow\_logic
  - creature::Living, [112](#)
- Shadowable, [143](#)
  - ~Shadowable, [144](#)
  - drawShadow, [144](#)
  - get\_height\_offset, [145](#)
  - get\_shadow\_strength, [145](#)
  - get\_skew\_offset, [145](#)
  - height\_offset, [147](#)
  - set\_height\_offset, [145](#)
  - set\_shadow\_strength, [146](#)
  - set\_skew\_offset, [146](#)
  - setShadow, [146](#)
  - setShadowDayNightCycle, [146](#)
  - setShadowPosition, [147](#)
  - setShadowTexture, [147](#)
- simulate\_key\_press
  - sf::Keyboard, [100](#)
  - sf::Mouse, [116](#)
- simulate\_key\_release
  - sf::Keyboard, [100](#)
  - sf::Mouse, [116](#)
- simulate\_tick
  - GameManager, [74](#)
- SimulationException, [148](#)
  - SimulationException, [148](#)
- Soldier
  - creature::Soldier, [149](#)
- sound\_player
  - WorldBase, [206](#)
- SoundPlayer, [152](#)
  - load\_sound, [152](#)
  - play\_sound, [153](#)
  - stop\_sound, [153](#)
- SoundSource
  - sf::SoundSource, [154](#)
- SoundSourceType
  - sf::SoundSource, [154](#)
- Space
  - sf::Keyboard, [99](#)
- spacePreviouslyPressed
  - EntityPlacer, [59](#)
- spawn
  - ObjectRegistry, [121](#)
- spawn\_entity
  - WorldBase, [204](#)
- spawn\_entity\_at\_pos
  - World, [198](#)
- spawn\_human
  - World, [198](#)
- spawn\_structure
  - WorldBase, [204](#)
- spawn\_structure\_at
  - WorldBase, [205](#)
- speed
  - creature::EntityBase, [57](#)
- Sprite
  - sf::Sprite, [155](#)
- src/creatures/EntityBase.cpp, [211](#)
- src/creatures/EntityBase.d, [211](#)
- src/creatures/EntityBase.h, [211](#)
- src/creatures/EntityUtils.h, [212](#)
- src/creatures/Goat.cpp, [212](#)
- src/creatures/Goat.d, [212](#)
- src/creatures/Goat.h, [212](#)
- src/creatures/HostileInterface.cpp, [213](#)
- src/creatures/HostileInterface.d, [213](#)
- src/creatures/HostileInterface.h, [213](#)
- src/creatures/hostiles/Bear.cpp, [213](#)
- src/creatures/hostiles/Bear.d, [214](#)
- src/creatures/hostiles/Bear.h, [214](#)
- src/creatures/hostiles/Crocodile.cpp, [214](#)
- src/creatures/hostiles/Crocodile.d, [214](#)
- src/creatures/hostiles/Crocodile.h, [214](#)
- src/creatures/hostiles/KillerRobot.cpp, [215](#)
- src/creatures/hostiles/KillerRobot.d, [215](#)
- src/creatures/hostiles/KillerRobot.h, [215](#)
- src/creatures/humans/AnglerMiner.cpp, [215](#)
- src/creatures/humans/AnglerMiner.d, [216](#)
- src/creatures/humans/AnglerMiner.h, [216](#)
- src/creatures/humans/Builder.cpp, [216](#)
- src/creatures/humans/Builder.d, [216](#)
- src/creatures/humans/Builder.h, [216](#)
- src/creatures/humans/Farmer.cpp, [217](#)
- src/creatures/humans/Farmer.d, [217](#)
- src/creatures/humans/Farmer.h, [217](#)
- src/creatures/humans/Fisherman.cpp, [217](#)
- src/creatures/humans/Fisherman.d, [218](#)
- src/creatures/humans/Fisherman.h, [218](#)
- src/creatures/humans/Human.cpp, [218](#)
- src/creatures/humans/Human.d, [218](#)
- src/creatures/humans/Human.h, [218](#)
- src/creatures/humans/King.cpp, [219](#)
- src/creatures/humans/King.d, [219](#)
- src/creatures/humans/King.h, [219](#)
- src/creatures/humans/Soldier.cpp, [219](#)
- src/creatures/humans/Soldier.d, [220](#)



src/creatures/humans/Soldier.h, 220  
 src/creatures/humans/Stonemason.cpp, 220  
 src/creatures/humans/Stonemason.d, 220  
 src/creatures/humans/Stonemason.h, 220  
 src/creatures/humans/Woodcutter.cpp, 221  
 src/creatures/humans/Woodcutter.d, 221  
 src/creatures/humans/Woodcutter.h, 221  
 src/creatures/Living.cpp, 221  
 src/creatures/Living.d, 222  
 src/creatures/Living.h, 222  
 src/EntityPlacer.cpp, 222  
 src/EntityPlacer.d, 222  
 src/EntityPlacer.h, 222  
 src/exceptions/FileExceptions.h, 223  
 src/exceptions/MusicLoadException.h, 223  
 src/exceptions/SimulationException.h, 224  
 src/exceptions/WorldExceptions.h, 224  
 src/external/memtrace.cpp, 224  
 src/external/memtrace.h, 224  
 src/external/modified\_gtest\_lite.h, 224  
 src/fake\_sfml/fake\_sfml.cpp, 233  
 src/fake\_sfml/fake\_sfml.d, 234  
 src/fake\_sfml/fake\_sfml.h, 234  
 src/GameConfig.cpp, 235  
 src/GameConfig.d, 235  
 src/GameConfig.h, 235  
 src/GameManager.cpp, 235  
 src/GameManager.d, 236  
 src/GameManager.h, 236  
 src/HumanResources.cpp, 236  
 src/HumanResources.d, 236  
 src/HumanResources.h, 236  
 src/main.cpp, 236  
 src/main.d, 237  
 src/MusicPlayer.cpp, 237  
 src/MusicPlayer.d, 237  
 src/MusicPlayer.h, 237  
 src/PostProcessor.cpp, 237  
 src/PostProcessor.d, 237  
 src/PostProcessor.h, 237  
 src/Profession.cpp, 238  
 src/Profession.d, 238  
 src/Profession.h, 238  
 src/Random\_Gen.cpp, 238  
 src/Random\_Gen.d, 238  
 src/Random\_Gen.h, 238  
 src/SaveHelpers.cpp, 238  
 src/SaveHelpers.d, 239  
 src/SaveHelpers.h, 239  
 src/SaveManager.cpp, 239  
 src/SaveManager.d, 240  
 src/SaveManager.h, 240  
 src/Shadowable.cpp, 240  
 src/Shadowable.d, 240  
 src/Shadowable.h, 240  
 src/SoundPlayer.cpp, 240  
 src/SoundPlayer.d, 240  
 src/SoundPlayer.h, 240  
 src/terrain\_tiles/Tile.cpp, 241  
 src/terrain\_tiles/Tile.d, 241  
 src/terrain\_tiles/Tile.h, 241  
 src/TerrainContainer.hpp, 241  
 src/TerrainContainer\_def.hpp, 242  
 src/Textureable.h, 242  
 src/TextureManager.cpp, 242  
 src/TextureManager.d, 242  
 src/TextureManager.h, 242  
 src/ui/button.cpp, 243  
 src/ui/button.d, 243  
 src/ui/button.h, 243  
 src/Utils.cpp, 243  
 src/Utils.d, 244  
 src/Utils.h, 244  
 src/World.cpp, 246  
 src/World.d, 246  
 src/World.hpp, 246  
 src/world\_object/BerryBush.cpp, 247  
 src/world\_object/BerryBush.d, 247  
 src/world\_object/BerryBush.h, 247  
 src/world\_object/CityCenter.cpp, 247  
 src/world\_object/CityCenter.d, 248  
 src/world\_object/CityCenter.h, 248  
 src/world\_object/House.cpp, 248  
 src/world\_object/House.d, 248  
 src/world\_object/House.h, 248  
 src/world\_object/Iron.cpp, 249  
 src/world\_object/Iron.d, 249  
 src/world\_object/Iron.h, 249  
 src/world\_object/ResourceStructure.cpp, 249  
 src/world\_object/ResourceStructure.d, 249  
 src/world\_object/ResourceStructure.h, 249  
 src/world\_object/Stone.cpp, 250  
 src/world\_object/Stone.d, 250  
 src/world\_object/Stone.h, 250  
 src/world\_object/Structure.cpp, 250  
 src/world\_object/Structure.d, 250  
 src/world\_object/Structure.h, 250  
 src/world\_object/Tree.cpp, 251  
 src/world\_object/Tree.d, 251  
 src/world\_object/Tree.h, 251  
 src/WorldBase.cpp, 251  
 src/WorldBase.d, 252  
 src/WorldBaseSerializable.cpp, 252  
 src/WorldBaseSerializable.d, 252  
 src/YAMLParse.cpp, 252  
 src/YAMLParse.d, 252  
 src/YAMLParse.h, 252  
 state  
     creature::EntityBase, 57  
 status  
     gtest\_lite::Test, 177  
 STONE  
     minerals, 20  
 Stone  
     minerals::Stone, 158  
 Stonemason

- creature::Stonemason, 160
- stop
  - sf::Music, 118
  - sf::Sound, 151
- stop\_sound
  - SoundPlayer, 153
- Stopped
  - sf::SoundSource, 154
- Structure
  - minerals::Structure, 163
- StructureException, 166
  - StructureException, 167
- structures
  - WorldBase, 207
- SUCCEED
  - modified\_gtest\_lite.h, 233
- sum
  - gtest\_lite::Test, 177
- swap\_at
  - TerrainContainer< T >, 173
- target
  - creature::HostileInterface, 81
- terrain
  - WorldBase, 207
- TerrainContainer
  - TerrainContainer< T >, 168
- TerrainContainer< T >, 167
  - ~TerrainContainer, 169
  - clear, 169
  - clear\_at, 169
  - draw, 170
  - generate\_world, 170
  - get\_height, 170
  - get\_seed, 170
  - get\_width, 171
  - is\_on\_screen, 171
  - is\_valid\_coordinate, 171
  - operator[], 172
  - resize, 172
  - set\_seed, 173
  - swap\_at, 173
  - TerrainContainer, 168
  - TILE\_SIZE, 173
- TEST
  - modified\_gtest\_lite.h, 233
- Texture
  - sf::Texture, 178
- texture\_data
  - creature::EntityBase, 57
- Textureable, 179
  - ~Textureable, 180
  - draw, 180
  - setPosition, 180
  - setTexture, 181
- TextureManager, 181
  - clear, 182
  - getInstance, 182
  - getTexture, 182
  - loadTexture, 182
- TILE\_SIZE
  - TerrainContainer< T >, 173
- tiles, 22
  - GRASS, 23
  - MOUNTAIN, 23
  - TILETYPE, 22
  - WATER, 23
- tiles::Tile, 183
  - draw, 184
  - get\_type, 184
  - init, 184
  - setPosition, 185
  - setTexture, 185
- TILETYPE
  - tiles, 22
- tmp
  - gtest\_lite::Test, 177
- to\_string
  - Profession, 128
  - sf, 22
- toggle\_chromatic\_aberration
  - PostProcessor, 125
- toggle\_music
  - MusicPlayer, 120
- toggle\_noise
  - PostProcessor, 125
- toggle\_placing
  - EntityPlacer, 59
- toggle\_vignette
  - PostProcessor, 125
- top
  - sf::FloatRect, 67
  - sf::IntRect, 96
- Transform
  - sf::Transform, 186
- transform
  - sf::RenderStates, 133
- transformPoint
  - sf::Transform, 186
- translate
  - sf::Transform, 187
- Transparent
  - sf::Color, 45
- Tree
  - minerals::Tree, 188
- trim
  - GameConfig.cpp, 235
- trim\_brackets
  - SaveHelper, 141
- try\_attack
  - creature::HostileInterface, 80
- try\_develop\_random\_role
  - World, 198
- try\_fishing
  - creature::Fisherman, 64
- try\_generate\_config\_file
  - YAMLParse, 210

- try\_hover\_animation
  - ui::Button, [38](#)
- try\_mine
  - creature::Stonemason, [161](#)
- try\_place\_entity
  - EntityPlacer, [59](#)
- try\_upgrade
  - minerals::House, [83](#)
- type
  - sf::Event, [61](#)
  - sf::SoundSource, [154](#)
- ui, [23](#)
- ui::Button, [35](#)
  - Button, [36](#)
  - draw, [37](#)
  - onClick, [37](#)
  - setCallback, [37](#)
  - setPosition, [37](#)
  - setTexture, [38](#)
  - try\_hover\_animation, [38](#)
- Up
  - sf::Keyboard, [99](#)
- update\_buttons
  - GameManager, [74](#)
- update\_logic
  - creature::AnglerMiner, [27](#)
  - creature::Bear, [30](#)
  - creature::Builder, [35](#)
  - creature::Crocodile, [48](#)
  - creature::Farmer, [62](#)
  - creature::Fisherman, [65](#)
  - creature::Goat, [77](#)
  - creature::Human, [88](#)
  - creature::KillerRobot, [103](#)
  - creature::King, [105](#)
  - creature::Living, [112](#)
  - creature::Soldier, [150](#)
  - creature::Stonemason, [161](#)
  - creature::Woodcutter, [194](#)
  - minerals::BerryBush, [32](#)
  - minerals::CityCenter, [40](#)
  - minerals::House, [83](#)
  - minerals::Iron, [98](#)
  - minerals::Stone, [159](#)
  - minerals::Structure, [165](#)
  - minerals::Tree, [188](#)
- update\_spritesheet
  - creature::Living, [112](#)
- update\_world
  - World, [199](#)
- upgrade\_house
  - minerals::House, [83](#)
- upgrade\_house\_at
  - WorldBase, [205](#)
- Utils.cpp
  - distance\_to, [243](#)
  - log\_text, [244](#)
  - warn\_text, [244](#)
- Utils.h
  - distance\_to, [245](#)
  - log\_text, [245](#)
  - warn\_text, [245](#)
  - WITH\_SFML\_RENDER, [245](#)
- Vector2f
  - sf::Vector2f, [189](#)
- Vector2i
  - sf::Vector2i, [190](#)
- VideoMode
  - sf::VideoMode, [191](#)
- WALK
  - creature, [15](#)
- walk\_texture\_path
  - creature::LivingTexture, [115](#)
- warn\_text
  - Utils.cpp, [244](#)
  - Utils.h, [245](#)
- WATER
  - tiles, [23](#)
- White
  - sf::Color, [46](#)
- width
  - sf::Bound, [33](#)
  - sf::FloatRect, [67](#)
  - sf::IntRect, [96](#)
  - sf::VideoMode, [192](#)
- WITH\_SFML\_RENDER
  - Utils.h, [245](#)
- WOOD
  - minerals, [20](#)
- Woodcutter
  - creature::Woodcutter, [193](#)
- World, [194](#)
  - ~World, [196](#)
  - clear, [196](#)
  - draw, [196](#)
  - get\_border\_height, [197](#)
  - get\_border\_width, [197](#)
  - populate\_world, [197](#)
  - regenerate, [197](#)
  - set\_border\_height, [197](#)
  - set\_border\_width, [198](#)
  - spawn\_entity\_at\_pos, [198](#)
  - spawn\_human, [198](#)
  - try\_develop\_random\_role, [198](#)
  - update\_world, [199](#)
  - World, [196](#)
- WorldBase, [199](#)
  - ~WorldBase, [201](#)
  - build\_city\_center\_at, [201](#)
  - camp\_needs\_spawn, [205](#)
  - current\_city\_center, [206](#)
  - entities, [206](#)
  - get\_current\_city\_center, [201](#)
  - get\_excluded\_entities, [201](#)
  - get\_position\_nearby\_town, [202](#)

- get\_random\_house\_pos, [202](#)
- get\_random\_suitable\_position, [202](#)
- get\_resources, [203](#)
- get\_structure\_type, [203](#)
- getTileAt, [203](#)
- houses, [206](#)
- humans, [206](#)
- MAX\_OBJECT\_SIZE, [206](#)
- remove\_structure\_at, [204](#)
- sound\_player, [206](#)
- spawn\_entity, [204](#)
- spawn\_structure, [204](#)
- spawn\_structure\_at, [205](#)
- structures, [207](#)
- terrain, [207](#)
- upgrade\_house\_at, [205](#)
- WorldBaseSerialiazble, [207](#)
  - clear, [208](#)
  - elapsed\_time, [209](#)
  - operator<<, [208](#)
  - operator>>, [209](#)
  - reinitialize\_self, [208](#)
  - saved\_size, [209](#)
- WorldBaseSerializable.cpp
  - operator<<, [252](#)
  - operator>>, [252](#)
- x
  - sf::Vector2f, [189](#)
  - sf::Vector2i, [191](#)
- y
  - sf::Vector2f, [190](#)
  - sf::Vector2i, [191](#)
- YAMLParse, [209](#)
  - get\_value\_of\_key, [210](#)
  - parse\_file, [210](#)
  - try\_generate\_config\_file, [210](#)
  - YAMLParse, [210](#)

## Házi feladat

Programozás alapjai 2.

Funk Gábor

YSDDH7

### Feladatválasztás (1. rész)

---

Valós idejű ember csoport szimulátor specifikációja

## Feladat

Egy valós idejű, felülnézetes, ember csoport szimuláció, ahol az emberek együttműködnek, erőforrásokat, nyersanyagokat szereznek és várost építenek.

A feladat egy szimulálható világot készíteni (hasonlót a Worldbox nevű játékhoz, csak egyszerűbbet), ahol ezt a szimulációt végre lehet hajtani.

## Feladatspecifikáció

### Külső források:

A grafikus megjelenítéshez és hangokhoz a program az SFML könyvtárat fogja használni.

### Írányítás:

A világban a felülnézetes kamerát a nyilakkal (jobb, bal, fel, le-vel) lehet mozgatni.

A menü 4 gombból fog állni: mentés, betöltés, új szimuláció.

A gombokat az egér bal gombjával lehet aktiválni.

A mentés gomb menti az állapotot, a betöltés pedig betölti, az új szimuláció pedig új állapotot hoz létre ami nem írja át automatikusan a mentett állapotot.

### Mentés:

Az állapotot a program a save\_data.dat nevű fájlból olvassa és menti. A mentés tartalmazni fogja:

- Élőlényeket
- Erőforrás lelő helyeket
- A világ terepéhez szükséges seedet
- A világban eltelt időt
- Az emberek által termelt erőforrásokat
- A napszakot
- Az emberek által épített házakat

A mentés fájl formátuma:

1. Sor: <a világ nagysága szám>|<az eltelt idő>|<a világ generálásához szükséges seed>
2. Sor: <emberek által gyűjtött vas száma>|<kő száma>|<étel száma>|<farönk száma>
3. Sor: entitások és attribútumaik ;-vel elválasztva. Példa (Név;X pozíció;Y pozíció; kor évben): <"Kecske";4;6;7>|<"Ember";20;41;5>...
4. Sor: az emberek által épített házakat és erőforrás lelő helyeket tárolja ugyanabban a formátumban, mint a 3. Sor.

## Világ:

A világ egy előre meghatározott méretű grid alapú szimuláció less, aminek a terepe véletlen generálást használ. Az élőlényeknek van egy maximum élettartamuk. Ha ezt eléri akkor meghalnak. Az élőlényeknek rengeteg célja lehet, amit a belső működésük határoz meg, de ahhoz, hogy ezt végre tudják hajtani, oda kell menniük ahol végre akarják a célt hajtani. Példa: A favágó ahhoz, hogy fát tudjon vágni, oda kell menni a fához.

## A világ terepe:

A grid 3 féle tile-ből fog állni:

-Víz: Itt lelhetőek halak, a halászok mindig ezeket a tile-okat fogják keresni.

-Mező: Itt lelhető fa és étel, ezen kívül itt medvék fognak idéződni.

-Hegy: Itt lelhető vas és kő.

A tile-ok típusa nem befolyásolja, hogy az élőlények milyen gyorsak azokra lépve.

## Emberek:

Az emberek speciális élőlények, amik képesek város létrehozására és építésére is.

Amennyiben már van létező városeelem akkor megpróbálnak egy szakmát felvenni. Egy szakma felvétele a városnak erőforrásba fog kerülni. Egy ember csak 1 szakmát vehet fel.

## Szakmák:

Minden ember feje felett ott fog lebegni egy ikon, ami mutatja, hogy milyen szakmája van. Ha nincs szakmája akkor egy "Zzz" alvó ikon lesz a feje fölött. Ezek a választható szakmák:

**Király:** Ezt a kasztot akkor kapja meg egy ember, ha ő hozta létre a várost. Létrehozás után csak sétál össze-vissza, így "felügyeli" a királyságát. Szakma ikon: arany korona.

**Harcos:** Ha egy ember ezt a kasztot viseli, akkor az a feladata, hogy vadásszon állatokat és megvédje a többi embert az ellenséges lényektől. Szakma ikon: kard

**Építész:** Ha van építésre elég erőforrás akkor a város közepe köré megpróbál új házakat építeni. Ha vannak régi házak, azokat is megpróbálja korszerűsíteni. Szakma ikon: téglá.

**Farmer:** Bogyóbokrokat keres és leszedi őket. Szakma ikon: kasza.

**Halász:** Vizet keres és a víz tile-okon halászik időnként. Szakma ikon: horgászbót.

**Bányász:** Vasércet és követ termel. Szakma ikon: csákány.

**Favágó:** Fákat keres és kivágja őket. Szakma ikon: fejsze.

**Angler-Miner:** Ez egy speciális kaszt, mivel akinek ez a szakmája az halászni is tud és bányászni is. Szakma ikon: csákány amin halak lógnak.

## **Más élőlények: (kecske,krokodil,medve,gyilkos robot)**

**Kecske:** Ártalmatlan állat.

**Krokodil:** Lassú de erős vadállat.

**Medve:** Gyors és nagyon ellenséges vadállat.

**Gyilkos robot:** Nagyon ritkán idéződik, az a célja, hogy mindenkit elpusztítson. 999 évig él, így szinte csak akkor tűnik el, ha az emberek elpusztítják.

## **Erőforrások:**

4 féle erőforrás van:

**stone** (kő): Házak és bizonyos szakmákhoz kell. Kőhegyből lehet szerezni.

**wood** (fa): Fa vágásával szerezhető. Szinte mindenhez kell.

**food** (étel): Bogyóbokorból, halászatból és vadászatból szerezhető. Szükséges, hogy az emberek életben maradjanak.

**iron** (vas): Fejlettebb szakmákhoz és modern házakhoz kell. Vasércből lehet kinyerni.

## **Erőforrás menedzsment:**

A világ nyilvántart egy globális erőforrástárolót, amibe minden ember bele tud nézni, bele tud rakni és ki is tud venni akármennyi erőforrást.

A világ bizonyos mennyiségű időnként "megpróbálja megetetni" az embereket. Ha egy ember nem kap ételt akkor meghal.

## **Épületek:**

Az épületek ahhoz, kell, hogy időnként új emberek idéződjenek. Épületet az építész tud készíteni, kivéve a városközép (egy kút) épületet. Azt egy ember akkor épít, ha nem talál létező várost. Az építész fejleszteni fogja a lakóházakat, ha van erőforrás maximum 2-szer. A fejlesztett házakból gyorsabban idéződnek új emberek.

## Kikötések:

- A városközép lerakásakor a globális erőforrástárolóba 10-10 erőforrás bekerül és 5 ember leideződik, hogy életképes legyen a szimuláció hosszú távon is. (Ez azért kell, hogy a király ne haljon meg egyből, amikor megcsinálja a várost).
- Időnként a világ idéz erőforrásokat, hogy ne foggyon ki belőlük.
- Csak 1 város lesz és minden ember eléri a többi ember által szerzett erőforrásokat.

## Tesztek:

A tesztelés az SFML könyvtár nélkül fog történni így:

Az SFML könyvtár helyett csinállok egy billboard SFML könyvtárat, ami megvalósítja az SFML-ből használt metódusokat, classokat, viszont ezeknek a funkcióját megváltoztatja. A rajzolások és egyéb SFML működések helyett ezek a billboard megvalósítások csak kiírják, hogy milyen művelet történt. Példa a setPosition-ra: "setPosition meghívva az x és y pozícióra" fog kiíródni meghíváskor.



## Házi feladat

# Programozás alapjai 2.

## Funk Gábor

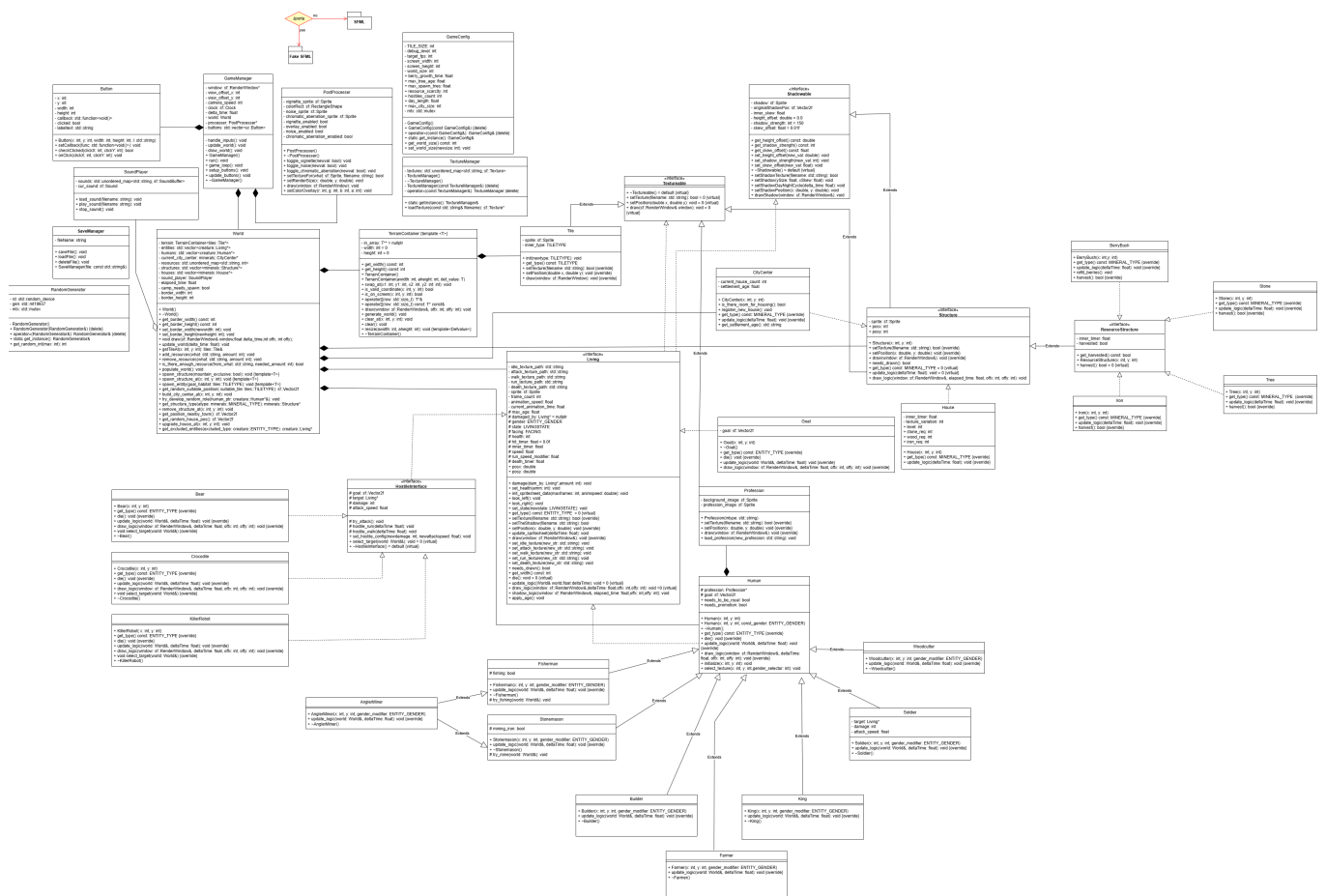
YSDDH7

## NHF Terv (2. rész)

## Valós idejű ember csoport szimulátor terve

## Terv és pontosított specifikáció

## Az osztálydiagram (UML):



A diagram tartalmazza az osztályokat, annak tagfüggvényeit és adattagjait. Sajnos a PDF formátumba exportálás elrontja a minőségét a nagy képeknek.

Az osztálydiagram elérhető nagy felbontásban ezen a linken: <https://bugfr.ee>

A Ctrl és “+” lenyomásával nagyítani lehet a képet, a Ctrl és “-” lenyomásával pedig kicsinyíteni. A görgő egyszeres lenyomásával és az egér mozgatásával fel,le,jobbra,balra lehet navigálni.

## Külső források:

A grafikus megjelenítéshez és hangokhoz a program az SFML könyvtárat fogja használni.

Annak érdekében, hogy a Jportán tesztelhető legyen a program, készíteni fogok egy Kamu\_SFML könyvtárat, amibe hasonló osztályok és metódusok lesznek, mint az igazi SFML grafikus könyvtárban, csak nem rajzolnak ki semmit. Ezek a saját osztályok csak kiabáló osztályok lesznek, például, ha betöltök egy textúrát egy fájlból egy változóba, akkor kiíródik, hogy **“Textúra betöltve innen: ./kepek/kep.png”**. Hibakezelés is lesz, például, ha nem találja a képet a program, akkor a **“Textúra betöltése sikertelen innen: ./kepek/hianyzo.png”**.

## Játékmenedzser:

Ez lesz a fő osztály, ez fog felelni a világ szimulációjának elindításához, a grafikus ablak létrehozásához és majd ha a program befejeződik, felszabadítja a világot. Ez az osztály fogja végezni a gombok tárolását és frissítését is, valamint az irányításkezelést is.

A világban a felülnézetes kamerát a nyilakkal (jobb, bal, fel, le-vel) lehet mozgatni. A menü 4 gombból fog állni: mentés, betöltés, új szimuláció. A gombokat az egér bal gombjával lehet aktiválni.

A következő oldalon található a játékmenedzserhez (Game Manager) tartozó classok UML osztálydiagramja. Ezen csak a az egyszerűbb és átláthatóbb képért a szomszédos osztályok láthatóak.



világ x:0 y:0-tól kezdődik) csak akkor frissíti a kamera helyét, ha az nem megy ki a világból.

## Világ:

A világ egy előre meghatározott méretű grid alapú szimuláció less, aminek a terepe véletlen generálást használ. Az élőlényeknek van egy maximum élettartamuk. Ha ezt eléri akkor meghalnak. Az élőlényeknek rengeteg célja lehet, amit a belső működésük határoz meg, de ahhoz, hogy ezt végre tudják hajtani, oda kell menniük ahol végre akarják a célt hajtani. Példa: A favágó ahhoz, hogy fát tudjon vágni, oda kell menni a fához.

A világ fontosabb adattagjai:

- Terrain: Ez tárolja a terep kockákat. Ez egy speciális dinamikus 2 dimenziós tömb, amely képes föld, víz, hegy kockákat tárolni
- Entities: Ez a heterogén kollekció tárolja az összes entitást az emberek kivételével
- Humans: Ez a heterogén kollekció tárolja az összes embert.
- Jelenlegi városközpont: Egy pointer a jelenlegi város központra.
- Structures: Eltárolja a fákat, bokrokat, követ és minden erőforrás struktúrát egy heterogén kollekcióba.
- Houses: A házak tárolására alkalmas dinamikus tömb.
- Resources: Az emberek által bányászott erőforrások tárolása.

A világ fontosabb metódusai:

- Draw(): Kirajzol mindent, ami a világba van, a terepkockákat, entitásokat és az erőforrás lelőhelyeket.
- Update(): Frissít minden entitást és struktúrát az előző frissítés óta eltelt idő függvényébe.
- Populate\_world(): Új állatokat és erőforrásokat idéz a világba. Amikor a világ létrejön (A konstruktor meghívja) akkor is meghívódik ez. Ezen kívül időnként meghívódik, hogy ez emberek sose fussanak ki a fából, kőből, vasból vagy ételből.

## Textureable és Shadowable interface:

2 Fontos interface-t tervezek, melyek egyszerűsítik a kirajzolást:

«interface» Textureable
<pre>+ ~Textureable() = default {virtual} + setTexture(filename: std::string): bool = 0 {virtual} + setPosition(double x, double y): void = 0 {virtual} + draw(sf::RenderWindow&amp; window): void = 0 {virtual}</pre>

«interface» Shadowable
<pre>- shadow: sf::Sprite - originalShadowPos: sf::Vector2f - inner_skew: float - height_offset: double = 0.0 - shadow_strength: int = 150 - skew_offset: float = 0.01f</pre>
<pre>+ get_height_offset() const: double + get_shadow_strength() const: int + get_skew_offset() const: float + set_height_offset(new_val: double): void + set_shadow_strength(new_val: int): void + set_skew_offset(new_val: float): void + ~Shadowable() = default {virtual} + setShadowTexture(filename: std::string): bool + setShadow(ySize: float, xSkew: float): void + setShadowDayNightCycle(delta_time: float): void + setShadowPosition(x: double, y: double): void + drawShadow(window: sf::RenderWindow&amp;): void</pre>

### Textureable:

- setTexture(): Egy textúra beállítása. A textureManager nevű class loadTexture() nevű metódusát használva megnézi, hogy be van-e töltve már a keresett textúra. Ha van akkor azt beállítja magának, ha nincs akkor betölti a fájlból és azután állítja be magának. Fontos, hogy az, hogy hova állítja be a textúrát majd a megvalósítástól függ.

- setPosition(x,y): megvalósítástól függően beállítja a pozíciót, hogy hova kell kirajzolódnia.

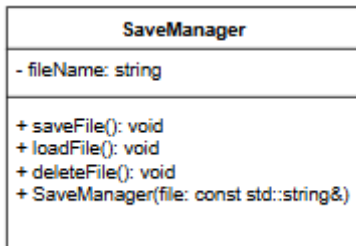
-draw(): Kirajzolja a sprite-okat vagy sprite-ot megvalósítástól függően.

### Shadowable:

Ez az interface azért fontos, mert a világba lesz napszak és ez fogja megvalósítani, hogy például az emberek árnyéka időtől függően merre álljon. Ha dél van akkor az árnyék nem látszik, ha este van, akkor se.

Egy kirajzolható dolognak lehet a kinézetétől függetlenül másik árnyéka, és lehet egy olyan entitás is, ami láthatatlan, ezért fontos, hogy ez a 2 interface külön legyen.

### Mentés:



A mentés

A mentés gomb menti az állapotot, a betöltés pedig betölti, az új szimuláció pedig új állapotot hoz létre ami nem írja át automatikusan a mentett állapotot.

Az állapotot a program a save\_data.dat nevű fájlból olvassa és menti. A mentés tartalmazni fogja:

- Élőlényeket
- Erőforrás lelő helyeket
- A világ terepéhez szükséges seedet
- A világban eltelt időt
- Az emberek által termelt erőforrásokat
- A napszakot
- Az emberek által épített házakat

A mentés fájl formátuma:

1. Sor: **<a világ nagysága szám>|<az eltelt idő>|<a világ generálásához szükséges seed>**
2. Sor: **<emberek által gyűjtött vas száma>|<kő száma>|<étel száma>|<farönk száma>**
3. Sor: entitások és attribútumaik ;-vel elválasztva. Példa (*Név;X pozíció;Y pozíció; kor évben*): **<"Kecske";4;6;7>|<"Ember";20;41;5>...**
4. Sor: az emberek által épített házakat és erőforrás lelő helyeket tárolja ugyanabban a formátumban, mint a 3. Sor.

## A világ terepe:

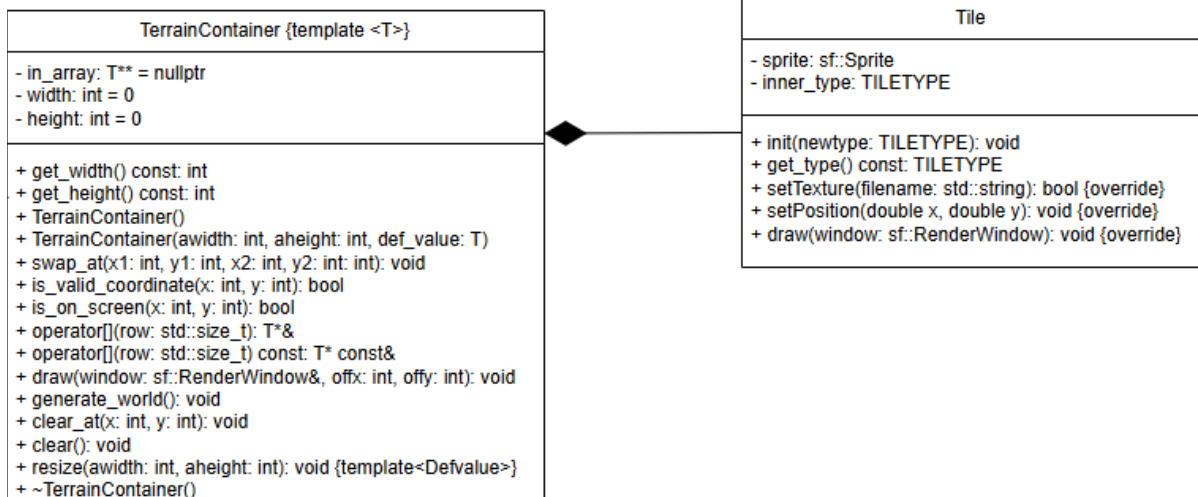
A világ egy 2 dimenziós, speciális dinamikus tömb lesz, ami képes tárolni terepkockákat.

Inicializálásnál meg kell adni egy N számot, amely a világ nagysága lesz. N-től függően a világ N\*N terepkockából fog állni.

A terep tároló felelős a saját terepkockáiért, ha megszűnik ez a tároló, akkor megszünteti a tárolt terepkockákat is.

A terep tároló képes lesz egy véletlen világot generálni, ahol a tengerek, hegyek és tavak véletlenül fognak elhelyezkedni.

Az alábbi UML diagram részlet bemutatja, körülbelül hogy fog kinézni a terep tároló a végleges fázisban.



A grid 3 féle tile-ból fog állni:

-Víz: Itt lelhetőek halak, a halászok mindig ezeket a tile-okat fogják keresni.

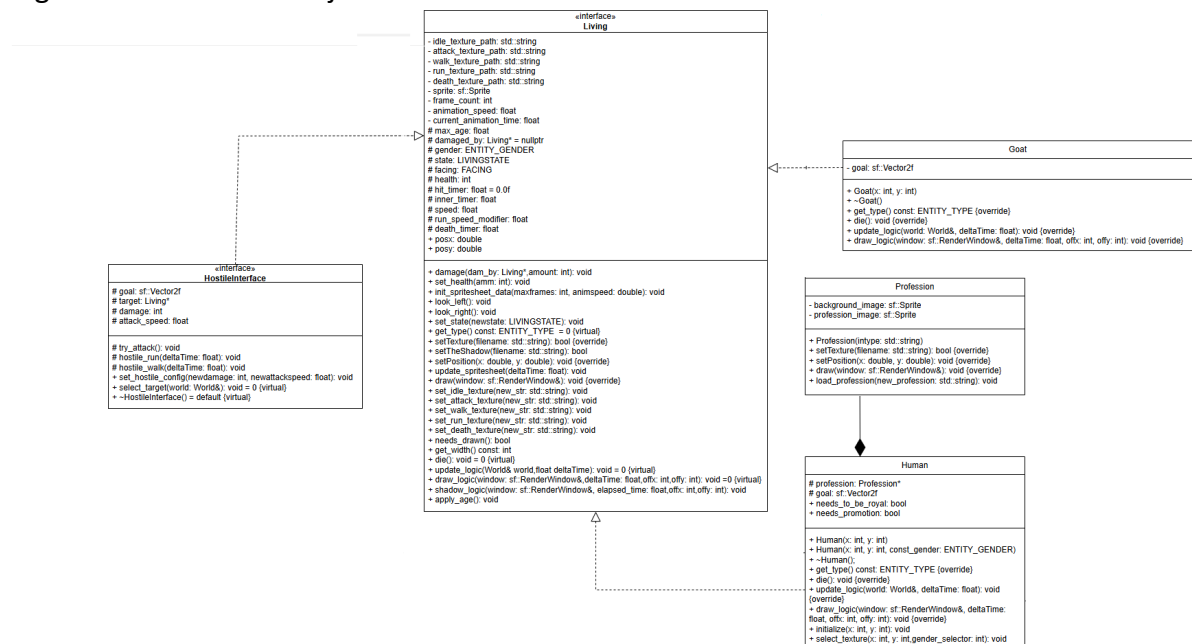
-Mező: Itt lelhető fa és étel, ezen kívül itt medvék fognak idéződni.

-Hegy: Itt lelhető vas és kő.

A tile-ok típusa nem befolyásolja, hogy az élőlények milyen gyorsak azokra lépve.

## Entitások:

Az entitások generalizálására készíték egy Living interface-t, amely megköti, hogy mi kötelező egy entitásnak. Ez a diagram tartalmazza a fontosabb interfaceket és osztályokat, amelyeknek még lesz sok leszármozottja:



Az ember osztályból fognak a különböző emberek szakmától függően öröklődni. A szakma (Profession) osztály az csak egy jelző lesz, ami kirajzolja annak a szakmának az ikonját, amit az ember művel. Az ellenséges interface-ből (HostileInterface) fognak azok az állatok / entitások öröklődni, amelyek más állatokat vagy embereket képesek megtámadni. A kecske egy semleges entitás, ezért ő csak a Living interfaceből örökl. A

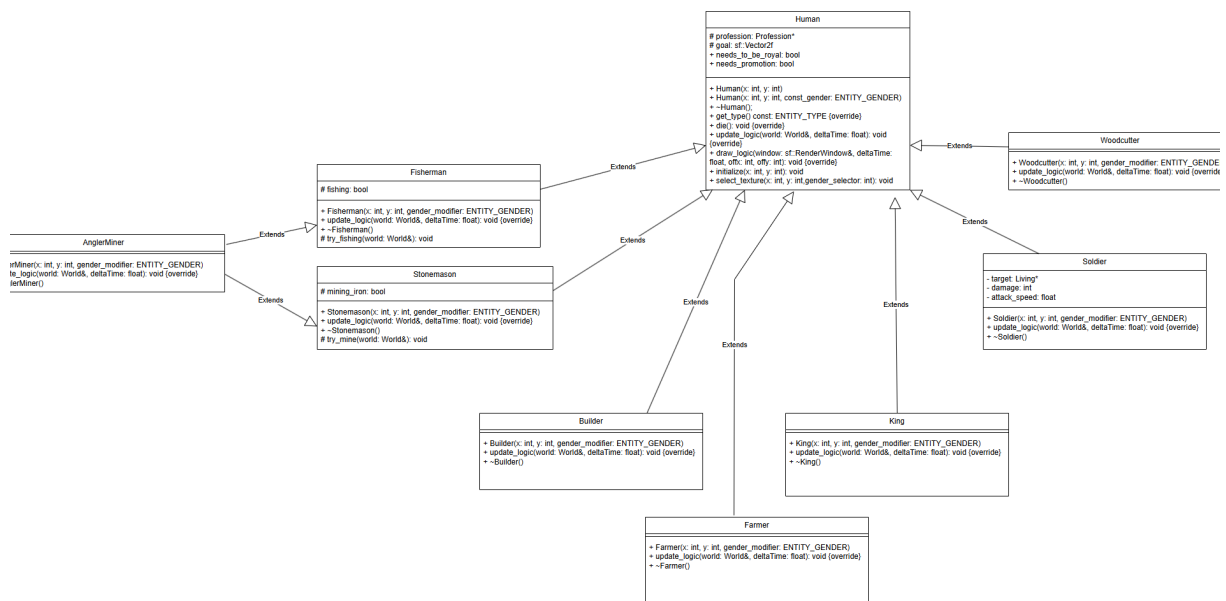
living interface megkívánja, hogy a belőle öröklődő osztályok megvalósítsák ezeket a tagfüggvényeket:

- Die(): Mi történjen, ha meghal az entitás?
- Get\_type(): Ez egy enumerációt ad vissza. "HUMAN" (Ember)-t, ha ember az élőlény, vadállatot.
- Update\_logic(): Itt kell leírni egy élőlény viselkedését. Például a krokodil odamegy más állatokhoz és megeszi őket. A halász pedig tavat keres, hogy tudjon halászni.
- Draw\_logic(): Hogyan rajzolódjon ki az állat? Például az embernél a szakma ikon-t is ki kell rajzolni.

## Emberek:

Az emberek speciális élőlények, amik képesek város létrehozására és építésére is. Amennyiben már van létező városeelem akkor megpróbálnak egy szakmát felvenni. Egy szakma felvétele a városnak erőforrásba fog kerülni. Egy ember csak 1 szakmát vehet fel.

## Szakmák:



Az emberek szakmától függően fognak viselkedni. Minden szakma az ember osztályból öröklődik. Az "Angler Miner" kivétel, mert ő a bányászból és a horgászból fog öröklődni. Az ő esetében fennáll a gyémánt öröklődés.

Minden ember feje felett ott fog lebegni egy ikon, ami mutatja, hogy milyen szakmája van. Ha nincs szakmája akkor egy "Zzz" alvó ikon lesz a feje fölött. Ezek a választható szakmák:

**Király:** Ezt a kasztot akkor kapja meg egy ember, ha ő hozta létre a várost. Létrehozás után csak sétál össze-vissza, így "felügyeli" a királyságát. Szakma ikon: arany korona.

**Harcos:** Ha egy ember ezt a kasztot viseli, akkor az a feladata, hogy vadásson állatokat és megvédje a többi embert az ellenséges lényektől. Szakma ikon: kard



**Építész:** Ha van építésre elég erőforrás akkor a város közepe köré megpróbál új házakat építeni. Ha vannak régi házak, azokat is megpróbálja korszerűsíteni. Szakma ikon: téglá.

**Farmer:** Bogyóbokrokat keres és leszedi őket. Szakma ikon: kasza.

**Halász:** Vízet keres és a víz tile-okon halászik időnként. Szakma ikon: horgászbót.

**Bányász:** Vasércet és követ termel. Szakma ikon: csákány.

**Favágó:** Fákat keres és kivágja őket. Szakma ikon: fejsze.

**Angler-Miner:** Ez egy speciális kaszt, mivel akinek ez a szakmája az halászni is tud és bányászni is. Szakma ikon: csákány amin halak lógnak.

## Más élőlények: (kecske,krokodil,medve,gyilkos robot)

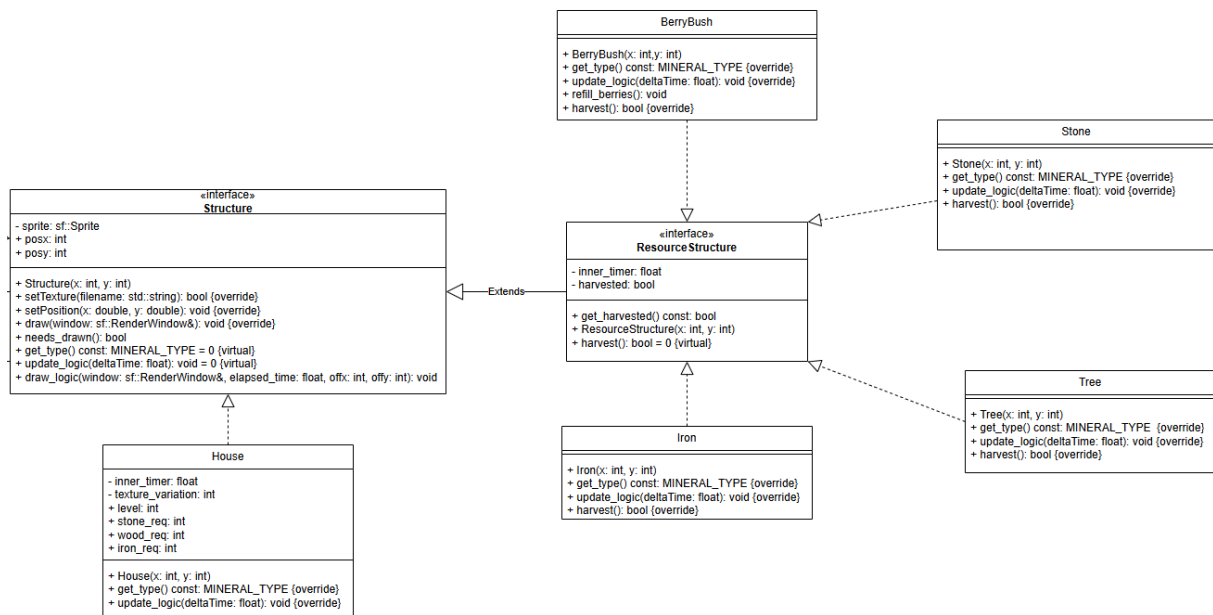
**Kecske:** Ártalmatlan állat.

**Krokodil:** Lassú de erős vadállat.

**Medve:** Gyors és nagyon ellenséges vadállat.

**Gyilkos robot:** Nagyon ritkán idéződik, az a célja, hogy mindenkit elpusztítson. 999 évig él, így szinte csak akkor tűnik el, ha az emberek elpusztítják.

## Erőforrások:



Az erőforrás struktúra interface-ből öröklődik 4 darab “épület”, ami a világba megjelenik. Ha ezeket az emberek lebontják, akkor különféle erőforrásokat kapnak. Létezik a sima struktúra is, amelyet nem akarnak az emberek lebontani. Ilyen a városközpont és a ház.

4 féle erőforrás van:

**stone** (kő): Házak és bizonyos szakmákhoz kell. Kőhegyből lehet szerezni.

**wood** (fa): Fa vágásával szerezhető. Szinte mindenhez kell.

**food** (étel): Bogyóbokorból, halászatból és vadászatból szerezhető. Szükséges, hogy az emberek életben maradjanak.

**iron** (vas): Fejlettebb szakmákhoz és modern házakhoz kell. Vasércből lehet kinyerni.

## **Erőforrás menedzsment:**

A világ nyilvántart egy globális erőforrástárolót, amibe minden ember bele tud nézni, bele tud rakni és ki is tud venni akármennyi erőforrást.

A világ bizonyos mennyiségű időnként "megpróbálja megetetni" az embereket. Ha egy ember nem kap ételt akkor meghal.

## **Épületek:**

Az épületek ahhoz, kell, hogy időnként új emberek idéződjének. Épületet az építész tud készíteni, kivéve a városközép (egy kút) épületet. Azt egy ember akkor épít, ha nem talál létező várost. Az építész fejleszteni fogja a lakóházakat, ha van erőforrás maximum 2-szer. A fejlesztett házakból gyorsabban idéződnék új emberek.

## **TextureManager és PostProcessor:**

A gyorsabb fejlesztés érdekében létrehozok segédosztályokat, melyek arra szolgálnak, hogy gyorsítsák a programot:

### **TextureManager:**

Eltárolja a már betöltött textúrákat, hogy később, ha szükség van még egyszer ugyanarra a textúrára, ne kelljen kétszer betölteni.

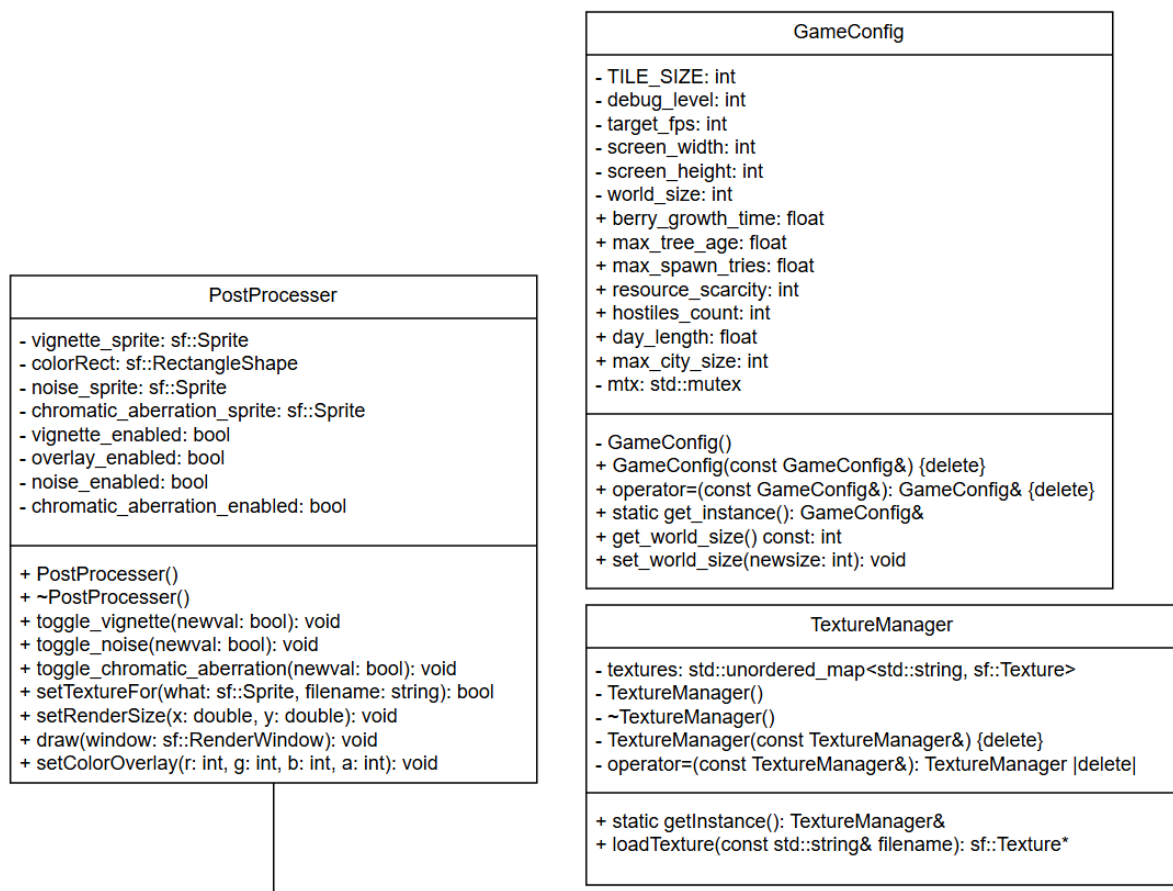
### **Postprocessor:**

Vizuális effektek beállítására való osztály. Lehet zajt, elmosódást és színhatás effekteket beállítani vele. Ezen felül még beállítható az, hogy a képernyő széle sötétebb legyen, ezzel az éjszaka napszakot szimulálva.

### **GameConfig:**

Ez egy Config osztály, ami azt segíti, hogy ne rendszertelenül legyenek szétszórva azok a változók, amelyek fontosak a szimulációhoz. Így könnyebb kísérletezni, hogy milyen beállításokkal kapok élethűbb szimulációt.

A kezdetleges UML diagramja ezeknek az osztályoknak:



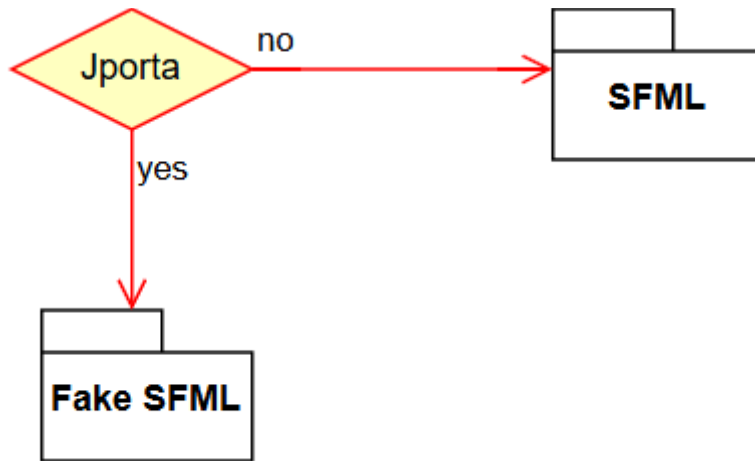
## Kikötések:

- A városközpont lerakásakor a globális erőforrástárolóba 10-10 erőforrás bekerül és 5 ember leidéződik, hogy életképes legyen a szimuláció hosszú távon is. (Ez azért kell, hogy a király ne haljon meg egyből, amikor megcsinálja a várost).
- Időnként a világ idéz erőforrásokat, hogy ne foggyon ki belőlük.
- Csak 1 város lesz és minden ember eléri a többi ember által szerzett erőforrásokat.

## Tesztek:

A tesztelés az SFML könyvtár nélkül fog történni így:

Az SFML könyvtár helyett csinállok egy billboard SFML könyvtárat, ami megvalósítja az SFML-ből használt metódusokat, classokat, viszont ezeknek a funkciójit megváltoztatja. A rajzolások és egyéb SFML működések helyett ezek a billboard megvalósítások csak kiírják, hogy milyen művelet történt. Példa a setPosition-ra: "setPosition meghívva az x és y pozícióra" fog kiíródni meghíváskor.



A jportára beadott változatba az én nem igazi SFML könyvtáram lesz beadva, így tesztelhetővé válik ott is.

#### **Első teszt tervek:**

- A tesztprogram betölt egy mentést, amiben van 5 ember és nincs semmilyen veszélyes állat. Futtatja a tesztet 10 szimulációs napig, hibát ad, ha már nem 5 ember van, mert nem szabadott volna meghalni 1-nek sem.
- A tesztprogram betölt egy mentést, ahol már van városközpont és 1 ember. Egy iterációt szimulál és megnézi, hogy van-e munkája az embernek (Kéne lennie, mert már van városközpont).
- Betölt egy világot amibe nincs étel, és futtatja a szimulációt. Az embereknek éhen kell halnia, ha életben maradnak, a teszt sikertelen.
- Teszteli azt, hogy lehet-e új erőforrást rakni a világba.
- Teszteli azt, hogy lehet-e új embert vagy állatot idézni a világba.
- Próbál kivenni több erőforrást az emberek erőforrást tárolójából, mint amennyi van.
- A tesztprogram megpróbálja a kamerát a pályára kívülre mozgatni.