

Humanity simulator (Készítette: Funk Gábor)

Készítette Doxygen 1.9.1

1. Névtérmutató	1
1.1. Névtérlista	1
2. Hierarchikus mutató	3
2.1. Osztályhierarchia	3
3. Osztálymutató	5
3.1. Osztálylista	5
4. Fájlmutató	9
4.1. Fájllista	9
5. Névterek dokumentációja	13
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. gtest_lite névtér-referencia	15
5.2.1. Részletes leírás	16
5.2.2. Függvények dokumentációja	16
5.2.2.1. almostEQ()	16
5.2.2.2. eq()	16
5.2.2.3. eqstr()	17
5.2.2.4. eqstrcase()	17
5.2.2.5. EXPECT_() [1/2]	17
5.2.2.6. EXPECT_() [2/2]	17
5.2.2.7. EXPECTSTR()	18
5.2.2.8. ge()	18
5.2.2.9. gt()	18
5.2.2.10. le()	18
5.2.2.11. lt()	18
5.2.2.12. ne()	19
5.2.2.13. nestr()	19
5.3. minerals névtér-referencia	19
5.3.1. Részletes leírás	20
5.3.2. Enumerációk dokumentációja	20
5.3.2.1. MINERAL_TYPE	20
5.3.3. Függvények dokumentációja	20
5.3.3.1. mineral_to_string()	20
5.4. sf névtér-referencia	20
5.4.1. Enumerációk dokumentációja	21

5.4.1.1. BlendMode	21
5.4.2. Függvények dokumentációja	22
5.4.2.1. file_exists_at_path()	22
5.4.2.2. to_string()	22
5.4.3. Változók dokumentációja	22
5.4.3.1. BlendAdd	22
5.5. tiles névtér-referencia	22
5.5.1. Részletes leírás	22
5.5.2. Enumerációk dokumentációja	22
5.5.2.1. TILETYPE	22
5.6. ui névtér-referencia	23
5.6.1. Részletes leírás	23
6. Osztályok dokumentációja	25
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. convertible	26
6.2. creature::AnglerMiner osztályreferencia	26
6.2.1. Részletes leírás	27
6.2.2. Konstruktorkok és destruktorkok 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. Konstruktorkok és destruktorkok 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. Konstruktorkok és destruktorkok 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.	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. Konstruktorok 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. Konstruktorok 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. Konstruktorok 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. Konstruktorok és destruktorok dokumentációja	51
6.14.2.1. ~EntityBase()	51

6.14.3. Tagfüggvények dokumentációja	51
6.14.3.1. apply_age()	51
6.14.3.2. die()	52
6.14.3.3. get_gender()	52
6.14.3.4. get_state()	52
6.14.3.5. get_type()	52
6.14.3.6. set_attack_texture()	52
6.14.3.7. set_death_texture()	53
6.14.3.8. set_health()	53
6.14.3.9. set_idle_texture()	53
6.14.3.10. set_run_texture()	54
6.14.3.11. set_state()	54
6.14.3.12. set_walk_texture()	54
6.14.4. Adattagok dokumentációja	54
6.14.4.1. death_timer	54
6.14.4.2. facing	55
6.14.4.3. gender	55
6.14.4.4. health	55
6.14.4.5. hit_timer	55
6.14.4.6. inner_timer	55
6.14.4.7. max_age	55
6.14.4.8. posx	56
6.14.4.9. posy	56
6.14.4.10. run_speed_modifier	56
6.14.4.11. save_name	56
6.14.4.12. speed	56
6.14.4.13. state	56
6.14.4.14. texture_data	57
6.15. EntityPlacer osztályreferencia	57
6.15.1. Részletes leírás	57
6.15.2. Konstruktorkok és destruktorkok dokumentációja	57
6.15.2.1. EntityPlacer()	57
6.15.3. Tagfüggvények dokumentációja	58
6.15.3.1. reset_mouse()	58
6.15.3.2. select_entity()	58
6.15.3.3. setup_factory()	58
6.15.3.4. toggle_placing()	58
6.15.3.5. try_place_entity()	58
6.15.4. Adattagok dokumentációja	58
6.15.4.1. spacePreviouslyPressed	59
6.16. sf::Event osztályreferencia	59
6.16.1. Enumeráció-tagok dokumentációja	59

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

6.20.3.5. <code>get_max_spawn_tries()</code>	68
6.20.3.6. <code>get_resource_scarcity()</code>	69
6.20.3.7. <code>get_screen_height()</code>	69
6.20.3.8. <code>get_screen_width()</code>	69
6.20.3.9. <code>get_sfml_lang()</code>	69
6.20.3.10. <code>get_target_fps()</code>	69
6.20.3.11. <code>get_world_size()</code>	69
6.20.3.12. <code>operator=()</code>	70
6.20.3.13. <code>read_from_config_file()</code>	70
6.20.3.14. <code>set_config_level()</code>	70
6.20.3.15. <code>set_world_size()</code>	70
6.20.4. Adattagok dokumentációja	70
6.20.4.1. <code>day_length</code>	70
6.21. GameManager osztályreferencia	71
6.21.1. Részletes leírás	71
6.21.2. Konstruktork és destruktorok dokumentációja	71
6.21.2.1. <code>GameManager()</code>	71
6.21.2.2. <code>~GameManager()</code>	72
6.21.3. Tagfüggvények dokumentációja	72
6.21.3.1. <code>draw_buttons()</code>	72
6.21.3.2. <code>game_loop()</code>	72
6.21.3.3. <code>get_elapsed_time()</code>	72
6.21.3.4. <code>handle_unit_placement()</code>	72
6.21.3.5. <code>is_valid()</code>	72
6.21.3.6. <code>run()</code>	73
6.21.3.7. <code>setup_buttons()</code>	73
6.21.3.8. <code>simulate_tick()</code>	73
6.21.3.9. <code>update_buttons()</code>	73
6.22. creature::Goat osztályreferencia	73
6.22.1. Részletes leírás	74
6.22.2. Konstruktork és destruktorok dokumentációja	74
6.22.2.1. <code>Goat()</code>	74
6.22.2.2. <code>~Goat()</code>	74
6.22.3. Tagfüggvények dokumentációja	75
6.22.3.1. <code>die()</code>	75
6.22.3.2. <code>draw_logic()</code>	75
6.22.3.3. <code>get_type()</code>	75
6.22.3.4. <code>update_logic()</code>	76
6.23. creature::HostileInterface osztályreferencia	76
6.23.1. Részletes leírás	77
6.23.2. Konstruktork és destruktorok dokumentációja	77
6.23.2.1. <code>~HostileInterface()</code>	77

6.23.3. Tagfüggvények dokumentációja	77
6.23.3.1. check_aggroed()	78
6.23.3.2. hostile_run()	78
6.23.3.3. hostile_walk()	78
6.23.3.4. retarget()	78
6.23.3.5. select_target()	79
6.23.3.6. set_hostile_config()	79
6.23.3.7. try_attack()	79
6.23.4. Adattagok dokumentációja	79
6.23.4.1. attack_speed	79
6.23.4.2. damage	80
6.23.4.3. goal	80
6.23.4.4. target	80
6.24. minerals::House osztályreferencia	80
6.24.1. Részletes leírás	81
6.24.2. Konstruktorkok és destruktorok dokumentációja	81
6.24.2.1. House()	81
6.24.3. Tagfüggvények dokumentációja	81
6.24.3.1. get_type()	81
6.24.3.2. update_logic()	81
6.24.4. Adattagok dokumentációja	82
6.24.4.1. iron_req	82
6.24.4.2. level	82
6.24.4.3. stone_req	82
6.24.4.4. wood_req	82
6.25. creature::Human osztályreferencia	83
6.25.1. Részletes leírás	84
6.25.2. Konstruktorkok és destruktorok dokumentációja	84
6.25.2.1. Human() [1/2]	84
6.25.2.2. Human() [2/2]	84
6.25.2.3. ~Human()	85
6.25.3. Tagfüggvények dokumentációja	85
6.25.3.1. die()	85
6.25.3.2. draw_logic()	85
6.25.3.3. get_profession_string()	85
6.25.3.4. get_type()	86
6.25.3.5. initialize()	86
6.25.3.6. select_texture()	86
6.25.3.7. update_logic()	87
6.25.4. Adattagok dokumentációja	87
6.25.4.1. goal	87
6.25.4.2. needs_promotion	87

6.25.4.3. needs_to_be_royal	87
6.25.4.4. profession	88
6.26. HumanResources osztályreferencia	88
6.26.1. Részletes leírás	88
6.26.2. Tagfüggvények dokumentációja	88
6.26.2.1. add_resources()	88
6.26.2.2. get_count_from()	89
6.26.2.3. is_there_enough_resource()	89
6.26.2.4. remove_resources()	89
6.26.2.5. set_resources()	90
6.27. ImportInvalidEntityException osztályreferencia	90
6.27.1. Részletes leírás	90
6.27.2. Konstruktork és destruktorok dokumentációja	91
6.27.2.1. ImportInvalidEntityException()	91
6.28. ImportInvalidHousingLevelException osztályreferencia	91
6.28.1. Részletes leírás	91
6.28.2. Konstruktork és destruktorok dokumentációja	91
6.28.2.1. ImportInvalidHousingLevelException()	92
6.29. ImportInvalidHumanProfessionException osztályreferencia	92
6.29.1. Részletes leírás	92
6.29.2. Konstruktork és destruktorok dokumentációja	92
6.29.2.1. ImportInvalidHumanProfessionException()	92
6.30. ImportInvalidResourceException osztályreferencia	93
6.30.1. Részletes leírás	93
6.30.2. Konstruktork és destruktorok dokumentációja	93
6.30.2.1. ImportInvalidResourceException()	93
6.31. sf::IntRect osztályreferencia	93
6.31.1. Konstruktork és destruktorok dokumentációja	94
6.31.1.1. IntRect() [1/2]	94
6.31.1.2. IntRect() [2/2]	94
6.31.2. Adattagok dokumentációja	94
6.31.2.1. height	94
6.31.2.2. left	94
6.31.2.3. top	94
6.31.2.4. width	95
6.32. InvalidBorderSizeException osztályreferencia	95
6.32.1. Részletes leírás	95
6.32.2. Konstruktork és destruktorok dokumentációja	95
6.32.2.1. InvalidBorderSizeException()	95
6.33. minerals::Iron osztályreferencia	96
6.33.1. Részletes leírás	96
6.33.2. Konstruktork és destruktorok dokumentációja	96

6.33.2.1. Iron()	96
6.33.3. Tagfüggvények dokumentációja	97
6.33.3.1. get_type()	97
6.33.3.2. harvest()	97
6.33.3.3. update_logic()	97
6.34. sf::Keyboard osztályreferencia	97
6.34.1. Enumeráció-tagok dokumentációja	98
6.34.1.1. Key	98
6.34.2. Tagfüggvények dokumentációja	98
6.34.2.1. isKeyPressed()	98
6.34.2.2. simulate_key_press()	99
6.34.2.3. simulate_key_release()	99
6.35. creature::KillerRobot osztályreferencia	99
6.35.1. Részletes leírás	100
6.35.2. Konstruktork és destruktorok dokumentációja	100
6.35.2.1. KillerRobot()	100
6.35.2.2. ~KillerRobot()	100
6.35.3. Tagfüggvények dokumentációja	100
6.35.3.1. die()	100
6.35.3.2. draw_logic()	101
6.35.3.3. get_type()	101
6.35.3.4. select_target()	101
6.35.3.5. update_logic()	102
6.36. creature::King osztályreferencia	102
6.36.1. Részletes leírás	103
6.36.2. Konstruktork és destruktorok dokumentációja	103
6.36.2.1. King()	103
6.36.2.2. ~King()	103
6.36.3. Tagfüggvények dokumentációja	103
6.36.3.1. update_logic()	103
6.37. creature::Living osztályreferencia	104
6.37.1. Részletes leírás	105
6.37.2. Konstruktork és destruktorok dokumentációja	106
6.37.2.1. ~Living()	106
6.37.3. Tagfüggvények dokumentációja	106
6.37.3.1. check_aggroed()	106
6.37.3.2. damage()	106
6.37.3.3. draw()	106
6.37.3.4. draw_logic()	107
6.37.3.5. get_width()	107
6.37.3.6. init_spritesheet_data()	107
6.37.3.7. look_left()	108

6.37.3.8. look_right()	108
6.37.3.9. needs_drawn()	108
6.37.3.10. retarget()	108
6.37.3.11. set_state()	109
6.37.3.12. setPosition()	109
6.37.3.13. setTexture()	109
6.37.3.14. setTheShadow()	110
6.37.3.15. shadow_logic()	110
6.37.3.16. update_logic()	110
6.37.3.17. update_spritesheet()	111
6.37.4. Adattagok dokumentációja	111
6.37.4.1. damaged_by	111
6.37.4.2. MAX_CREATURE_SIZE	111
6.38. creature::LivingTexture osztályreferencia	111
6.38.1. Részletes leírás	112
6.38.2. Adattagok dokumentációja	112
6.38.2.1. animation_speed	112
6.38.2.2. attack_texture_path	112
6.38.2.3. current_animation_time	112
6.38.2.4. death_texture	113
6.38.2.5. frame_count	113
6.38.2.6. idle_texture_path	113
6.38.2.7. run_texture_path	113
6.38.2.8. walk_texture_path	113
6.39. sf::Mouse osztályreferencia	113
6.39.1. Enumeráció-tagok dokumentációja	114
6.39.1.1. Mousedowntype	114
6.39.2. Tagfüggvények dokumentációja	114
6.39.2.1. getPosition()	114
6.39.2.2. isButtonPressed()	114
6.39.2.3. simulate_key_press()	115
6.39.2.4. simulate_key_release()	115
6.40. sf::Music osztályreferencia	115
6.40.1. Konstruktorok és destruktorok dokumentációja	115
6.40.1.1. Music()	115
6.40.2. Tagfüggvények dokumentációja	115
6.40.2.1. getStatus()	116
6.40.2.2. openFromFile()	116
6.40.2.3. play()	116
6.40.2.4. setLoop()	116
6.40.2.5. setVolume()	116
6.40.2.6. stop()	116

6.41. MusicLoadException osztályreferencia	117
6.41.1. Részletes leírás	117
6.41.2. Konstruktork és destruktorok dokumentációja	117
6.41.2.1. MusicLoadException()	117
6.42. MusicPlayer osztályreferencia	117
6.42.1. Részletes leírás	118
6.42.2. Konstruktork és destruktorok dokumentációja	118
6.42.2.1. MusicPlayer()	118
6.42.2.2. ~MusicPlayer()	118
6.42.3. Tagfüggvények dokumentációja	118
6.42.3.1. load_music()	118
6.42.3.2. set_volume()	119
6.42.3.3. toggle_music()	119
6.43. ObjectRegistry osztályreferencia	119
6.43.1. Részletes leírás	119
6.43.2. Tagfüggvények dokumentációja	120
6.43.2.1. register_type()	120
6.43.2.2. spawn()	120
6.44. gtest_lite::ostreamRedir osztályreferencia	120
6.44.1. Részletes leírás	120
6.44.2. Konstruktork és destruktorok dokumentációja	120
6.44.2.1. ostreamRedir()	121
6.44.2.2. ~ostreamRedir()	121
6.45. PostProcessor osztályreferencia	121
6.45.1. Részletes leírás	121
6.45.2. Konstruktork és destruktorok dokumentációja	122
6.45.2.1. PostProcessor()	122
6.45.3. Tagfüggvények dokumentációja	122
6.45.3.1. draw()	122
6.45.3.2. setColorOverlay()	122
6.45.3.3. setRenderSize()	123
6.45.3.4. setTextureFor()	123
6.45.3.5. toggle_chromatic_aberration()	123
6.45.3.6. toggle_noise()	124
6.45.3.7. toggle_vignette()	124
6.46. Profession osztályreferencia	124
6.46.1. Részletes leírás	125
6.46.2. Konstruktork és destruktorok dokumentációja	125
6.46.2.1. Profession()	125
6.46.3. Tagfüggvények dokumentációja	125
6.46.3.1. draw()	125
6.46.3.2. load_profession()	126

6.46.3.3. setPosition()	126
6.46.3.4. setTexture()	126
6.46.3.5. to_string()	127
6.47. RandomGenerator osztályreferencia	127
6.47.1. Részletes leírás	127
6.47.2. Konstruktork és destruktorok dokumentációja	128
6.47.2.1. RandomGenerator()	128
6.47.3. Tagfüggvények dokumentációja	128
6.47.3.1. get_instance()	128
6.47.3.2. get_random_int()	128
6.47.3.3. operator=()	129
6.48. ReadSaveFileFail osztályreferencia	129
6.48.1. Részletes leírás	129
6.48.2. Konstruktork és destruktorok dokumentációja	129
6.48.2.1. ReadSaveFileFail()	129
6.49. sf::RectangleShape osztályreferencia	130
6.49.1. Tagfüggvények dokumentációja	130
6.49.1.1. setFillColor()	130
6.49.1.2. setPosition()	130
6.49.1.3. setSize()	130
6.49.2. Adattagok dokumentációja	130
6.49.2.1. position	131
6.50. sf::RenderStates osztályreferencia	131
6.50.1. Konstruktork és destruktorok dokumentációja	131
6.50.1.1. RenderStates()	131
6.50.2. Tagfüggvények dokumentációja	131
6.50.2.1. setBlendMode()	131
6.50.2.2. setTransform()	132
6.50.3. Adattagok dokumentációja	132
6.50.3.1. blendMode	132
6.50.3.2. transform	132
6.51. sf::RenderWindow osztályreferencia	132
6.51.1. Konstruktork és destruktorok dokumentációja	133
6.51.1.1. RenderWindow() [1/3]	133
6.51.1.2. RenderWindow() [2/3]	133
6.51.1.3. RenderWindow() [3/3]	133
6.51.2. Tagfüggvények dokumentációja	133
6.51.2.1. clear() [1/2]	133
6.51.2.2. clear() [2/2]	133
6.51.2.3. close()	133
6.51.2.4. create()	134
6.51.2.5. display()	134

6.51.2.6. draw() [1/3]	134
6.51.2.7. draw() [2/3]	134
6.51.2.8. draw() [3/3]	134
6.51.2.9. isOpen()	134
6.51.2.10. pollEvent()	134
6.51.2.11. setFrameRateLimit()	135
6.52. minerals::ResourceStructure osztályreferencia	135
6.52.1. Részletes leírás	136
6.52.2. Konstruktorkok és destruktorkok dokumentációja	136
6.52.2.1. ResourceStructure()	136
6.52.2.2. ~ResourceStructure()	136
6.52.3. Tagfüggvények dokumentációja	136
6.52.3.1. get_harvested()	136
6.52.3.2. harvest()	136
6.52.4. Adattagok dokumentációja	137
6.52.4.1. harvested	137
6.52.4.2. inner_timer	137
6.53. RoleOption struktúráreferencia	137
6.53.1. Részletes leírás	137
6.53.2. Adattagok dokumentációja	137
6.53.2.1. create	138
6.53.2.2. requirements	138
6.54. SaveHelper osztályreferencia	138
6.54.1. Részletes leírás	138
6.54.2. Típusdefiníció-tagok dokumentációja	138
6.54.2.1. CreatureFactory	138
6.54.2.2. HumanFactory	139
6.54.2.3. ResourceFactory	139
6.54.3. Tagfüggvények dokumentációja	139
6.54.3.1. getCreatureFactory()	139
6.54.3.2. getHumanFactory()	139
6.54.3.3. getResourceFactory()	139
6.55. SaveManager osztályreferencia	139
6.55.1. Részletes leírás	140
6.55.2. Konstruktorkok és destruktorkok dokumentációja	140
6.55.2.1. SaveManager()	140
6.55.3. Tagfüggvények dokumentációja	140
6.55.3.1. deleteFile()	140
6.55.3.2. loadFile()	141
6.55.3.3. saveFile()	142
6.56. Shadowable osztályreferencia	142
6.56.1. Részletes leírás	143

6.56.2. Konstruktork és destruktorok dokumentációja	143
6.56.2.1. ~Shadowable()	143
6.56.3. Tagfüggvények dokumentációja	143
6.56.3.1. drawShadow()	143
6.56.3.2. get_height_offset()	144
6.56.3.3. get_shadow_strength()	144
6.56.3.4. get_skew_offset()	144
6.56.3.5. set_height_offset()	144
6.56.3.6. set_shadow_strength()	145
6.56.3.7. set_skew_offset()	145
6.56.3.8. setShadow()	145
6.56.3.9. setShadowDayNightCycle()	146
6.56.3.10. setShadowPosition()	146
6.56.3.11. setShadowTexture()	146
6.56.4. Adattagok dokumentációja	146
6.56.4.1. height_offset	147
6.57. SimulationException osztályreferencia	147
6.57.1. Részletes leírás	147
6.57.2. Konstruktork és destruktorok dokumentációja	147
6.57.2.1. SimulationException()	147
6.58. creature::Soldier osztályreferencia	148
6.58.1. Részletes leírás	148
6.58.2. Konstruktork és destruktorok dokumentációja	148
6.58.2.1. Soldier()	148
6.58.2.2. ~Soldier()	149
6.58.3. Tagfüggvények dokumentációja	149
6.58.3.1. update_logic()	149
6.59. sf::Sound osztályreferencia	149
6.59.1. Konstruktork és destruktorok dokumentációja	150
6.59.1.1. ~Sound()	150
6.59.2. Tagfüggvények dokumentációja	150
6.59.2.1. play()	150
6.59.2.2. setBuffer()	150
6.59.2.3. stop()	150
6.60. sf::SoundBuffer osztályreferencia	150
6.60.1. Tagfüggvények dokumentációja	151
6.60.1.1. loadFromFile()	151
6.61. SoundPlayer osztályreferencia	151
6.61.1. Részletes leírás	151
6.61.2. Tagfüggvények dokumentációja	151
6.61.2.1. load_sound()	151
6.61.2.2. play_sound()	152

6.61.2.3. stop_sound()	152
6.62. sf::SoundSource osztályreferencia	152
6.62.1. Enumeráció-tagok dokumentációja	153
6.62.1.1. SoundSourceType	153
6.62.2. Konstruktorok és destruktorkok dokumentációja	153
6.62.2.1. SoundSource()	153
6.62.2.2. ~SoundSource()	153
6.62.3. Adattagok dokumentációja	153
6.62.3.1. type	153
6.63. sf::Sprite osztályreferencia	154
6.63.1. Konstruktorok és destruktorkok dokumentációja	154
6.63.1.1. Sprite()	154
6.63.1.2. ~Sprite()	154
6.63.2. Tagfüggvények dokumentációja	154
6.63.2.1. draw()	154
6.63.2.2. getGlobalBounds() [1/2]	155
6.63.2.3. getGlobalBounds() [2/2]	155
6.63.2.4. getLocalBounds()	155
6.63.2.5. getPosition()	155
6.63.2.6. getTexture()	155
6.63.2.7. setColor()	155
6.63.2.8. setOrigin()	155
6.63.2.9. setPosition()	156
6.63.2.10. setRotation()	156
6.63.2.11. setScale()	156
6.63.2.12. setTexture()	156
6.63.2.13. setTextureRect()	156
6.64. minerals::Stone osztályreferencia	157
6.64.1. Részletes leírás	157
6.64.2. Konstruktorok és destruktorkok dokumentációja	157
6.64.2.1. Stone()	157
6.64.3. Tagfüggvények dokumentációja	158
6.64.3.1. get_type()	158
6.64.3.2. harvest()	158
6.64.3.3. update_logic()	158
6.65. creature::Stonemason osztályreferencia	158
6.65.1. Részletes leírás	159
6.65.2. Konstruktorok és destruktorkok dokumentációja	159
6.65.2.1. Stonemason()	159
6.65.2.2. ~Stonemason()	160
6.65.3. Tagfüggvények dokumentációja	160
6.65.3.1. try_mine()	160

6.65.3.2. update_logic()	160
6.65.4. Adattagok dokumentációja	161
6.65.4.1. mining_iron	161
6.66. minerals::Structure osztályreferencia	161
6.66.1. Részletes leírás	162
6.66.2. Konstruktorkok és destruktorok dokumentációja	162
6.66.2.1. Structure()	162
6.66.2.2. ~Structure()	162
6.66.3. Tagfüggvények dokumentációja	162
6.66.3.1. draw()	163
6.66.3.2. draw_logic()	163
6.66.3.3. get_type()	163
6.66.3.4. needs_drawn()	163
6.66.3.5. setPosition()	164
6.66.3.6. setTexture()	164
6.66.3.7. update_logic()	164
6.66.4. Adattagok dokumentációja	165
6.66.4.1. MAX_OBJECT_SIZE	165
6.66.4.2. posx	165
6.66.4.3. posy	165
6.67. StructureException osztályreferencia	165
6.67.1. Részletes leírás	166
6.67.2. Konstruktorkok és destruktorok dokumentációja	166
6.67.2.1. StructureException()	166
6.68. TerrainContainer< T > osztálysablon-referencia	166
6.68.1. Részletes leírás	167
6.68.2. Konstruktorkok és destruktorok dokumentációja	167
6.68.2.1. TerrainContainer() [1/2]	167
6.68.2.2. TerrainContainer() [2/2]	168
6.68.2.3. ~TerrainContainer()	168
6.68.3. Tagfüggvények dokumentációja	168
6.68.3.1. clear()	168
6.68.3.2. clear_at()	168
6.68.3.3. draw()	169
6.68.3.4. generate_world()	169
6.68.3.5. get_height()	169
6.68.3.6. get_seed()	170
6.68.3.7. get_width()	170
6.68.3.8. is_on_screen()	170
6.68.3.9. is_valid_coordinate()	170
6.68.3.10.operator[]() [1/2]	171
6.68.3.11.operator[]() [2/2]	171

6.68.3.12. <code>resize()</code>	171
6.68.3.13. <code>set_seed()</code>	172
6.68.3.14. <code>swap_at()</code>	172
6.68.4. Adattagok dokumentációja	172
6.68.4.1. <code>TILE_SIZE</code>	172
6.69. <code>gtest_lite::Test</code> struktúrareferencia	173
6.69.1. Részletes leírás	174
6.69.2. Konstruktork és destruktorok dokumentációja	174
6.69.2.1. <code>~Test()</code>	174
6.69.3. Tagfüggvények dokumentációja	174
6.69.3.1. <code>astatus()</code>	174
6.69.3.2. <code>begin()</code>	174
6.69.3.3. <code>end()</code>	174
6.69.3.4. <code>expect()</code>	175
6.69.3.5. <code>fail()</code>	175
6.69.3.6. <code>getTest()</code>	175
6.69.4. Adattagok dokumentációja	175
6.69.4.1. <code>ablocks</code>	175
6.69.4.2. <code>failed</code>	175
6.69.4.3. <code>name</code>	176
6.69.4.4. <code>null</code>	176
6.69.4.5. <code>os</code>	176
6.69.4.6. <code>status</code>	176
6.69.4.7. <code>sum</code>	176
6.69.4.8. <code>tmp</code>	176
6.70. <code>sf::Texture</code> osztályreferencia	177
6.70.1. Konstruktork és destruktorok dokumentációja	177
6.70.1.1. <code>Texture()</code> [1/2]	177
6.70.1.2. <code>~Texture()</code>	177
6.70.1.3. <code>Texture()</code> [2/2]	177
6.70.2. Tagfüggvények dokumentációja	177
6.70.2.1. <code>getSize()</code>	177
6.70.2.2. <code>loadFromFile()</code>	178
6.70.2.3. <code>operator=()</code>	178
6.71. <code>Textureable</code> osztályreferencia	178
6.71.1. Részletes leírás	179
6.71.2. Konstruktork és destruktorok dokumentációja	179
6.71.2.1. <code>~Textureable()</code>	179
6.71.3. Tagfüggvények dokumentációja	179
6.71.3.1. <code>draw()</code>	179
6.71.3.2. <code>setPosition()</code>	179
6.71.3.3. <code>setTexture()</code>	180

6.72. TextureManager osztályreferencia	180
6.72.1. Részletes leírás	181
6.72.2. Tagfüggvények dokumentációja	181
6.72.2.1. clear()	181
6.72.2.2. getInstance()	181
6.72.2.3. getTexture()	181
6.72.2.4. loadTexture()	182
6.73. tiles::Tile osztályreferencia	182
6.73.1. Részletes leírás	183
6.73.2. Tagfüggvények dokumentációja	183
6.73.2.1. draw()	183
6.73.2.2. get_type()	183
6.73.2.3. init()	183
6.73.2.4. setPosition()	184
6.73.2.5. setTexture()	184
6.74. sf::Transform osztályreferencia	184
6.74.1. Konstruktorkok és destruktorkok dokumentációja	185
6.74.1.1. Transform() [1/2]	185
6.74.1.2. Transform() [2/2]	185
6.74.2. Tagfüggvények dokumentációja	185
6.74.2.1. combine()	185
6.74.2.2. transformPoint()	186
6.74.2.3. translate() [1/2]	186
6.74.2.4. translate() [2/2]	186
6.74.3. Adattagok dokumentációja	186
6.74.3.1. matrix	186
6.75. minerals::Tree osztályreferencia	186
6.75.1. Részletes leírás	187
6.75.2. Konstruktorkok és destruktorkok dokumentációja	187
6.75.2.1. Tree()	187
6.75.3. Tagfüggvények dokumentációja	187
6.75.3.1. get_type()	187
6.75.3.2. harvest()	188
6.75.3.3. update_logic()	188
6.76. sf::Vector2f osztályreferencia	189
6.76.1. Konstruktorkok és destruktorkok 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. Konstruktorkok és destruktorok 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. Konstruktorkok és destruktorok 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. Konstruktorkok és destruktorok 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	196
6.80.2. Konstruktorkok és destruktorok 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
6.80.4. Barát és kapcsolódó függvények dokumentációja	199

6.80.4.1. operator<<	199
6.80.4.2. operator>>	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()	202
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()	203
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()	205
6.81.3.13.spawn_structure_at()	205
6.81.3.14.upgrade_house_at()	205
6.81.4. Adattagok dokumentációja	206
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	207
6.81.4.7. sound_player	207
6.81.4.8. structures	207
6.81.4.9. terrain	207
6.82. YAMLParser osztályreferencia	207
6.82.1. Részletes leírás	208
6.82.2. Konstruktork és destruktorok dokumentációja	208
6.82.2.1. YAMLParser()	208
6.82.3. Tagfüggvények dokumentációja	208
6.82.3.1. get_value_of_key()	208
6.82.3.2. parse_file()	208
7. Fájlok dokumentációja	209
7.1. src/creatures/EntityBase.cpp fájlreferencia	209
7.2. src/creatures/EntityBase.d fájlreferencia	209

7.3. src/creatures/EntityBase.hpp fájlreferencia	209
7.4. src/creatures/EntityUtils.hpp fájlreferencia	210
7.5. src/creatures/Goat.cpp fájlreferencia	210
7.6. src/creatures/Goat.d fájlreferencia	210
7.7. src/creatures/Goat.hpp fájlreferencia	210
7.7.1. Részletes leírás	211
7.8. src/creatures/HostileInterface.cpp fájlreferencia	211
7.9. src/creatures/HostileInterface.d fájlreferencia	211
7.10. src/creatures/HostileInterface.hpp fájlreferencia	211
7.10.1. Részletes leírás	212
7.11. src/creatures/hostiles/Bear.cpp fájlreferencia	212
7.12. src/creatures/hostiles/Bear.d fájlreferencia	212
7.13. src/creatures/hostiles/Bear.hpp fájlreferencia	212
7.13.1. Részletes leírás	213
7.14. src/creatures/hostiles/Crocodile.cpp fájlreferencia	213
7.15. src/creatures/hostiles/Crocodile.d fájlreferencia	213
7.16. src/creatures/hostiles/Crocodile.hpp fájlreferencia	213
7.16.1. Részletes leírás	214
7.17. src/creatures/hostiles/KillerRobot.cpp fájlreferencia	214
7.18. src/creatures/hostiles/KillerRobot.d fájlreferencia	214
7.19. src/creatures/hostiles/KillerRobot.hpp fájlreferencia	214
7.19.1. Részletes leírás	215
7.20. src/creatures/humans/AnglerMiner.cpp fájlreferencia	215
7.21. src/creatures/humans/AnglerMiner.d fájlreferencia	215
7.22. src/creatures/humans/AnglerMiner.hpp fájlreferencia	215
7.22.1. Részletes leírás	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.hpp fájlreferencia	216
7.25.1. Részletes leírás	217
7.26. src/creatures/humans/Farmer.cpp fájlreferencia	217
7.27. src/creatures/humans/Farmer.d fájlreferencia	217
7.28. src/creatures/humans/Farmer.hpp fájlreferencia	217
7.28.1. Részletes leírás	218
7.29. src/creatures/humans/Fisherman.cpp fájlreferencia	218
7.30. src/creatures/humans/Fisherman.d fájlreferencia	218
7.31. src/creatures/humans/Fisherman.hpp fájlreferencia	218
7.31.1. Részletes leírás	219
7.32. src/creatures/humans/Human.cpp fájlreferencia	219
7.33. src/creatures/humans/Human.d fájlreferencia	219
7.34. src/creatures/humans/Human.hpp fájlreferencia	219
7.34.1. Részletes leírás	220

7.35. src/creatures/humans/King.cpp fájlreferencia	220
7.36. src/creatures/humans/King.d fájlreferencia	220
7.37. src/creatures/humans/King.hpp fájlreferencia	220
7.37.1. Részletes leírás	221
7.38. src/creatures/humans/Soldier.cpp fájlreferencia	221
7.39. src/creatures/humans/Soldier.d fájlreferencia	221
7.40. src/creatures/humans/Soldier.hpp fájlreferencia	221
7.40.1. Részletes leírás	222
7.41. src/creatures/humans/Stonemason.cpp fájlreferencia	222
7.42. src/creatures/humans/Stonemason.d fájlreferencia	222
7.43. src/creatures/humans/Stonemason.hpp fájlreferencia	222
7.43.1. Részletes leírás	223
7.44. src/creatures/humans/Woodcutter.cpp fájlreferencia	223
7.45. src/creatures/humans/Woodcutter.d fájlreferencia	223
7.46. src/creatures/humans/Woodcutter.hpp fájlreferencia	223
7.46.1. Részletes leírás	224
7.47. src/creatures/Living.cpp fájlreferencia	224
7.48. src/creatures/Living.d fájlreferencia	224
7.49. src/creatures/Living.hpp fájlreferencia	224
7.49.1. Részletes leírás	225
7.50. src/EntityPlacer.cpp fájlreferencia	225
7.51. src/EntityPlacer.d fájlreferencia	225
7.52. src/EntityPlacer.hpp fájlreferencia	225
7.52.1. Részletes leírás	226
7.53. src/exceptions/FileExceptions.hpp fájlreferencia	226
7.53.1. Részletes leírás	226
7.54. src/exceptions/MusicLoadException.hpp fájlreferencia	227
7.54.1. Részletes leírás	227
7.55. src/exceptions/SimulationException.hpp fájlreferencia	227
7.55.1. Részletes leírás	227
7.56. src/exceptions/WorldExceptions.hpp fájlreferencia	228
7.56.1. Részletes leírás	228
7.57. src/external/gtest_lite.h fájlreferencia	228
7.57.1. Részletes leírás	231
7.57.2. Makródefiníciók dokumentációja	231
7.57.2.1. ADD_FAILURE	231
7.57.2.2. ASSERT_	232
7.57.2.3. ASSERT_EQ	232
7.57.2.4. ASSERT_NO_THROW [1/2]	232
7.57.2.5. ASSERT_NO_THROW [2/2]	232
7.57.2.6. ASSERTTHROW	232
7.57.2.7. CREATE_Has_	233

7.57.2.8. CREATE_Has_fn_	233
7.57.2.9. END	233
7.57.2.10.ENDM	233
7.57.2.11.ENDMsg	233
7.57.2.12.EXPECT_ANY_THROW	233
7.57.2.13.EXPECT_DOUBLE_EQ	233
7.57.2.14.EXPECT_ENVCASEEQ	234
7.57.2.15.EXPECT_ENVEQ	234
7.57.2.16.EXPECT_EQ	234
7.57.2.17.EXPECT_FALSE	234
7.57.2.18.EXPECT_FLOAT_EQ	234
7.57.2.19.EXPECT_GE	234
7.57.2.20.EXPECT_GT	235
7.57.2.21.EXPECT_LE	235
7.57.2.22.EXPECT_LT	235
7.57.2.23.EXPECT_NE	235
7.57.2.24.EXPECT_NO_THROW	235
7.57.2.25.EXPECT_STRCASEEQ	235
7.57.2.26.EXPECT_STRCASENE	235
7.57.2.27.EXPECT_STREQ	236
7.57.2.28.EXPECT_STRNE	236
7.57.2.29.EXPECT_THROW	236
7.57.2.30.EXPECT_THROW_THROW	236
7.57.2.31.EXPECT_TRUE	236
7.57.2.32.EXPECTTHROW	236
7.57.2.33.Nem célszerű közvetlenül használni, vagy módosítani	237
7.57.2.34.FAIL	237
7.57.2.35.GTEND	237
7.57.2.36.GTINIT	237
7.57.2.37.SUCCEED	237
7.57.2.38.TEST	237
7.57.3. Függvények dokumentációja	237
7.57.3.1. hasMember()	237
7.58. src/external/memtrace.cpp fájlreferencia	238
7.59. src/external/memtrace.h fájlreferencia	238
7.60. src/fake_sfml/fake_sfml.cpp fájlreferencia	238
7.61. src/fake_sfml/fake_sfml.d fájlreferencia	238
7.62. src/fake_sfml/fake_sfml.hpp fájlreferencia	238
7.63. src/GameConfig.cpp fájlreferencia	239
7.63.1. Függvények dokumentációja	239
7.63.1.1. trim()	239
7.64. src/GameConfig.d fájlreferencia	239

7.65. src/GameConfig.hpp fájlreferencia	239
7.65.1. Részletes leírás	240
7.65.2. Enumerációk dokumentációja	240
7.65.2.1. Language	240
7.66. src/GameManager.cpp fájlreferencia	240
7.67. src/GameManager.d fájlreferencia	240
7.68. src/GameManager.hpp fájlreferencia	240
7.68.1. Részletes leírás	241
7.69. src/HumanResources.cpp fájlreferencia	241
7.70. src/HumanResources.d fájlreferencia	241
7.71. src/HumanResources.hpp fájlreferencia	241
7.72. src/main.cpp fájlreferencia	241
7.72.1. Függvények dokumentációja	241
7.72.1.1. main()	241
7.73. src/main.d fájlreferencia	242
7.74. src/MusicPlayer.cpp fájlreferencia	242
7.75. src/MusicPlayer.d fájlreferencia	242
7.76. src/MusicPlayer.hpp fájlreferencia	242
7.76.1. Részletes leírás	242
7.77. src/PostProcessor.cpp fájlreferencia	242
7.78. src/PostProcessor.d fájlreferencia	242
7.79. src/PostProcessor.hpp fájlreferencia	242
7.79.1. Részletes leírás	243
7.80. src/Profession.cpp fájlreferencia	243
7.81. src/Profession.d fájlreferencia	243
7.82. src/Profession.hpp fájlreferencia	243
7.82.1. Részletes leírás	243
7.83. src/Random_Gen.cpp fájlreferencia	243
7.84. src/Random_Gen.d fájlreferencia	243
7.85. src/Random_Gen.hpp fájlreferencia	243
7.85.1. Részletes leírás	244
7.86. src/SaveHelpers.cpp fájlreferencia	244
7.87. src/SaveHelpers.d fájlreferencia	244
7.88. src/SaveHelpers.hpp fájlreferencia	244
7.88.1. Részletes leírás	245
7.89. src/SaveManager.cpp fájlreferencia	245
7.90. src/SaveManager.d fájlreferencia	245
7.91. src/SaveManager.hpp fájlreferencia	245
7.91.1. Részletes leírás	246
7.92. src/Shadowable.cpp fájlreferencia	246
7.93. src/Shadowable.d fájlreferencia	246
7.94. src/Shadowable.hpp fájlreferencia	246

7.94.1. Részletes leírás	246
7.95. src/SoundPlayer.cpp fájlreferencia	246
7.96. src/SoundPlayer.d fájlreferencia	247
7.97. src/SoundPlayer.hpp fájlreferencia	247
7.97.1. Részletes leírás	247
7.98. src/terrain_tiles/Tile.cpp fájlreferencia	247
7.99. src/terrain_tiles/Tile.d fájlreferencia	247
7.100src/terrain_tiles/Tile.hpp fájlreferencia	247
7.100.1.Részletes leírás	248
7.101src/TerrainContainer.hpp fájlreferencia	248
7.101.1.Részletes leírás	248
7.102src/TerrainContainer_def.hpp fájlreferencia	248
7.103src/Textureable.hpp fájlreferencia	248
7.103.1.Részletes leírás	249
7.104src/TextureManager.cpp fájlreferencia	249
7.105src/TextureManager.d fájlreferencia	249
7.106src/TextureManager.hpp fájlreferencia	249
7.106.1.Részletes leírás	249
7.107src/ui/button.cpp fájlreferencia	249
7.108src/ui/button.d fájlreferencia	250
7.109src/ui/button.hpp fájlreferencia	250
7.109.1.Részletes leírás	250
7.110src/Utils.cpp fájlreferencia	250
7.110.1.Függvények dokumentációja	251
7.110.1.1.distance_to()	251
7.110.1.2.log_text()	251
7.110.1.3.warn_text()	251
7.111src/Utils.d fájlreferencia	251
7.112src/Utils.hpp fájlreferencia	251
7.112.1.Részletes leírás	252
7.112.2.Makródefiníciók dokumentációja	252
7.112.2.1.WITH_SFML_RENDER	252
7.112.3.Függvények dokumentációja	252
7.112.3.1.distance_to()	252
7.112.3.2.log_text()	253
7.112.3.3.warn_text()	253
7.113src/World.cpp fájlreferencia	253
7.113.1.Függvények dokumentációja	253
7.113.1.1.operator<<()	253
7.113.1.2.operator>>()	253
7.114src/World.d fájlreferencia	254
7.115src/World.hpp fájlreferencia	254

7.115.1.Részletes leírás	254
7.116src/world_object/BerryBush.cpp fájlreferencia	255
7.117src/world_object/BerryBush.d fájlreferencia	255
7.118src/world_object/BerryBush.hpp fájlreferencia	255
7.118.1.Részletes leírás	255
7.119src/world_object/CityCenter.cpp fájlreferencia	255
7.120src/world_object/CityCenter.d fájlreferencia	256
7.121src/world_object/CityCenter.hpp fájlreferencia	256
7.121.1.Részletes leírás	256
7.122src/world_object/House.cpp fájlreferencia	256
7.123src/world_object/House.d fájlreferencia	256
7.124src/world_object/House.hpp fájlreferencia	256
7.124.1.Részletes leírás	257
7.125src/world_object/Iron.cpp fájlreferencia	257
7.126src/world_object/Iron.d fájlreferencia	257
7.127src/world_object/Iron.hpp fájlreferencia	257
7.127.1.Részletes leírás	257
7.128src/world_object/ResourceStructure.cpp fájlreferencia	258
7.129src/world_object/ResourceStructure.d fájlreferencia	258
7.130src/world_object/ResourceStructure.hpp fájlreferencia	258
7.130.1.Részletes leírás	258
7.131src/world_object/Stone.cpp fájlreferencia	258
7.132src/world_object/Stone.d fájlreferencia	259
7.133src/world_object/Stone.hpp fájlreferencia	259
7.133.1.Részletes leírás	259
7.134src/world_object/Structure.cpp fájlreferencia	259
7.135src/world_object/Structure.d fájlreferencia	259
7.136src/world_object/Structure.hpp fájlreferencia	259
7.136.1.Részletes leírás	260
7.137src/world_object/Tree.cpp fájlreferencia	260
7.138src/world_object/Tree.d fájlreferencia	260
7.139src/world_object/Tree.hpp fájlreferencia	260
7.139.1.Részletes leírás	261
7.140src/WorldBase.cpp fájlreferencia	261
7.141src/WorldBase.d fájlreferencia	261
7.142src/YAMLParse.cpp fájlreferencia	261
7.143src/YAMLParse.d fájlreferencia	261
7.144src/YAMLParse.hpp fájlreferencia	261
7.144.1.Részletes leírás	261

1. fejezet

Névtérmutató

1.1. Névtérlista

Az összes névtér listája rövid leírásokkal:

creature	Az összes élőlény és entitás ebben a névtérben van	13
gtest_lite	Gtest_lite: a keretrendszer függvényinek és objektumainak névtére	15
minerals	Az összes struktúra ebben a névtérben van	19
sf	20
tiles	Az összes terepkocka elem ebben a névtérben van	22
ui	Az összes UI elem ebben a névtérben van	23

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	104
creature::Goat	73
creature::HostileInterface	76
creature::Bear	28
creature::Crocodile	46
creature::KillerRobot	99
creature::Human	83
creature::Builder	34
creature::Farmer	60
creature::Fisherman	62
creature::AnglerMiner	26
creature::King	102
creature::Soldier	148
creature::Stonemason	158
creature::AnglerMiner	26
creature::Woodcutter	193
EntityPlacer	57
sf::Event	59
sf::FloatRect	64
GameConfig	66
GameManager	71
HumanResources	88
sf::IntRect	93
sf::Keyboard	97
creature::LivingTexture	111
std::logic_error	
ImportInvalidHumanProfessionException	92
InvalidBorderSizeException	95

sf::Mouse	113
sf::Music	115
MusicPlayer	117
ObjectRegistry	119
gtest_lite::ostreamRedir	120
PostProcessor	121
RandomGenerator	127
sf::RectangleShape	130
sf::RenderStates	131
sf::RenderWindow	132
RoleOption	137
std::runtime_error	
SimulationException	147
CityCenterException	41
ImportInvalidEntityException	90
ImportInvalidHousingLevelException	91
ImportInvalidResourceException	93
MusicLoadException	117
ReadSaveFileFail	129
StructureException	165
SaveHelper	138
SaveManager	139
Shadowable	142
creature::Living	104
minerals::Structure	161
minerals::CityCenter	39
minerals::House	80
minerals::ResourceStructure	135
minerals::BerryBush	31
minerals::Iron	96
minerals::Stone	157
minerals::Tree	186
sf::Sound	149
sf::SoundBuffer	150
SoundPlayer	151
sf::SoundSource	152
sf::Sprite	154
TerrainContainer< T >	166
TerrainContainer< tiles::Tile * >	166
gtest_lite::Test	173
sf::Texture	177
Textureable	178
Profession	124
creature::Living	104
minerals::Structure	161
tiles::Tile	182
ui::Button	35
TextureManager	180
sf::Transform	184
sf::Vector2f	189
sf::Vector2i	190
sf::VideoMode	191
WorldBase	199
World	194
YAMLParse	207

3. fejezet

Osztálymutató

3.1. Osztálylista

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

_Is_Types< F, T >	25
Segédsablon típuskonverzió futás közbeni ellenőrzésere	
creature::AnglerMiner	26
Az "AnglerMiner" szakmájú ember osztály leírása	
creature::Bear	28
A medve osztály leírása	
minerals::BerryBush	31
A bokor osztály leírása. Ételt ad, ha kitermelik	
sf::Bound	33
creature::Builder	34
Az építész szakmájú ember osztály leírása	
ui::Button	35
A gomb osztály leírása. Tárolja a gomb méretét és azt, hogy mit csinál, ha rákattintanak	
minerals::CityCenter	39
A városközpont osztály leírása. E köré épülnek a házak	
CityCenterException	41
Akkor kell dobni, ha a városközpont hibásan működött	
sf::Clock	42
sf::ClockTime	42
sf::Color	43
creature::Crocodile	46
A krokodil osztály leírása	
creature::EntityBase	49
Egy alap, nem rajzolható entitás osztálya	
EntityPlacer	57
Az entitások a kattintással való lerakása	
sf::Event	59
creature::Farmer	60
A farmer szakmájú ember osztály leírása	
creature::Fisherman	62
A halász szakmájú ember osztály leírása	
sf::FloatRect	64
GameConfig	66
A világ szimulációjának leírása	
GameManager	71
A világ szimulálásáért és a kirazolás irányításáért felelős osztály	

creature::Goat	
A kecske osztály leírása	73
creature::HostileInterface	
A vadállat entitások interface leírása	76
minerals::House	
A ház osztály leírása. Szinttől függően idéz embereket	80
creature::Human	
Az alap ember osztály leírása. Minden fajta szakmájú ember innen öröklődik	83
HumanResources	
Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva	88
ImportInvalidEntityException	
Akkor kell dobni, ha egy entitás hibásan lett beolvasva	90
ImportInvalidHousingLevelException	
Akkor kell dobni, ha egy ház hibásan lett beolvasva	91
ImportInvalidHumanProfessionException	
Akkor kell dobni, ha egy szakma hibásan lett beolvasva	92
ImportInvalidResourceException	
Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva	93
sf::IntRect	93
InvalidBorderSizeException	
Akkor kell dobni, ha egy világhatárnak nem jó értéket akarnak beállítani	95
minerals::Iron	
A vasérc osztály leírása. Vasat ad, amikor kitermelik	96
sf::Keyboard	97
creature::KillerRobot	
A gyilkos robot osztály leírása	99
creature::King	
A király szakmájú ember osztály leírása	102
creature::Living	
Az élő entitások interface leírása	104
creature::LivingTexture	
Az élő entitások kinézetének adatai	111
sf::Mouse	113
sf::Music	115
MusicLoadException	
Akkor kell dobni, ha egy zene hibásan lett lejátszva vagy hibás a beolvasott zene	117
MusicPlayer	
A zene játészó osztály leírása	117
ObjectRegistry	
Az entitások és más világ objektumok lerakásának intézéséért felelős osztály	119
gtest_lite::ostreamRedir	120
PostProcessor	
A grafikus szépítő osztály leírása	121
Profession	
A szakma osztály leírása	124
RandomGenerator	
Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály	127
ReadSaveFileFail	
Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás	129
sf::RectangleShape	130
sf::RenderStates	131
sf::RenderWindow	132
minerals::ResourceStructure	
Az erőforrás struktúra osztály leírása	135
RoleOption	
Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni	137
SaveHelper	
Factory-k	138

SaveManager	
A fájl menedzseléshez szolgáló osztály leírása	139
Shadowable	
Az árnyékoláshoz szükséges interface	142
SimulationException	
Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik	147
creature::Soldier	
A katona szakmájú ember osztály leírása	148
sf::Sound	149
sf::SoundBuffer	150
SoundPlayer	
A hanglejátszó osztály leírása	151
sf::SoundSource	152
sf::Sprite	154
minerals::Stone	
A kő osztály leírása. követ ad, amikor kitermelik	157
creature::Stonemason	
A bányász szakmájú ember osztály leírása	158
minerals::Structure	
A struktúra osztály leírása	161
StructureException	
Akkor kell dobni, ha egy struktúra hibásan működött	165
TerrainContainer< T >	
A világ terepét tároló osztály	166
gtest_lite::Test	173
sf::Texture	177
Textureable	
Egy interface, ami a textúrázáshoz kell	178
TextureManager	
A Textúra kezelő osztály	180
tiles::Tile	
A terepkocka osztály leírása	182
sf::Transform	184
minerals::Tree	
A fa osztály leírása. Fát ad, ha kitermelik	186
sf::Vector2f	189
sf::Vector2i	190
sf::VideoMode	191
creature::Woodcutter	
A favágó szakmájú ember osztály leírása	193
World	
A világ osztály leírása	194
WorldBase	
A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza	199
YAMLParse	
Egy YAML (Yet Another Markup Language) fájl beolvasó osztály	207

4. fejezet

Fájlmutató

4.1. Fájllista

Az összes fájl listája rövid leírásokkal:

src/EntityPlacer.cpp	225
src/EntityPlacer.d	225
src/EntityPlacer.hpp	
Az entitások lerakásáért felelős osztály itt van deklarálva	225
src/GameConfig.cpp	239
src/GameConfig.d	239
src/GameConfig.hpp	
A Szimuláció konfigurációja itt érhető el	239
src/GameManager.cpp	240
src/GameManager.d	240
src/GameManager.hpp	
A játékmenedzser osztály itt van deklarálva	240
src/HumanResources.cpp	241
src/HumanResources.d	241
src/HumanResources.hpp	241
src/main.cpp	241
src/main.d	242
src/MusicPlayer.cpp	242
src/MusicPlayer.d	242
src/MusicPlayer.hpp	
A zene lejátszó osztály itt van deklarálva	242
src/PostProcessor.cpp	242
src/PostProcessor.d	242
src/PostProcessor.hpp	
A grafikus szépítő osztály deklarációját tartalmazza	242
src/Profession.cpp	243
src/Profession.d	243
src/Profession.hpp	
Ebben a fájlban van deklarálva a szakma indikátor osztály	243
src/Random_Gen.cpp	243
src/Random_Gen.d	243
src/Random_Gen.hpp	
A véletlen generátor osztályt tároló fájl	243
src/SaveHelpers.cpp	244
src/SaveHelpers.d	244

src/SaveHelpers.hpp	
A mentést segítő factory-k és segédfüggvények	244
src/SaveManager.cpp	245
src/SaveManager.d	245
src/SaveManager.hpp	
A fájl menedzseléshez szolgáló osztály itt van deklarálva	245
src/Shadowable.cpp	246
src/Shadowable.d	246
src/Shadowable.hpp	
Az árnyék szimulálásához való osztályt tartalmazza	246
src/SoundPlayer.cpp	246
src/SoundPlayer.d	247
src/SoundPlayer.hpp	
Ebben a fájlba vannak a hanglejátszó osztályhoz szükséges deklarációk	247
src/TerrainContainer.hpp	
A Világ terepének a deklarálása ebben a fájlba van	248
src/TerrainContainer_def.hpp	248
src/Textureable.hpp	
Ebbe a fájlba van a textúrázáshoz szükséges osztály	248
src/TextureManager.cpp	249
src/TextureManager.d	249
src/TextureManager.hpp	
Ebbe a fájlba van az az osztály, ami a textúrák betöltéséért, kiosztásáért és tárolásáért felelős	249
src/Utils.cpp	250
src/Utils.d	251
src/Utils.hpp	
Ebben a fájlba vannak a segéd függvények	251
src/World.cpp	253
src/World.d	254
src/World.hpp	
A Világ osztály, ami a fő szimulációs elemek tárolásáért felelős	254
src/WorldBase.cpp	261
src/WorldBase.d	261
src/YAMLParse.cpp	261
src/YAMLParse.d	261
src/YAMLParse.hpp	
A config beolvasó osztály itt található	261
src/creatures/EntityBase.cpp	209
src/creatures/EntityBase.d	209
src/creatures/EntityBase.hpp	209
src/creatures/EntityUtils.hpp	210
src/creatures/Goat.cpp	210
src/creatures/Goat.d	210
src/creatures/Goat.hpp	
A Kecske osztály itt van deklarálva	210
src/creatures/HostileInterface.cpp	211
src/creatures/HostileInterface.d	211
src/creatures/HostileInterface.hpp	
A vadállat interface itt van deklarálva	211
src/creatures/Living.cpp	224
src/creatures/Living.d	224
src/creatures/Living.hpp	
Az élő interface itt van deklarálva	224
src/creatures/hostiles/Bear.cpp	212
src/creatures/hostiles/Bear.d	212
src/creatures/hostiles/Bear.hpp	
A Medve osztály itt van deklarálva	212
src/creatures/hostiles/Crocodile.cpp	213

src/creatures/hostiles/Crocodile.d	213
src/creatures/hostiles/Crocodile.hpp	
A krokodil osztály itt van deklarálva	213
src/creatures/hostiles/KillerRobot.cpp	214
src/creatures/hostiles/KillerRobot.d	214
src/creatures/hostiles/KillerRobot.hpp	
A Gyilkos Robot osztály itt van deklarálva	214
src/creatures/humans/AnglerMiner.cpp	215
src/creatures/humans/AnglerMiner.d	215
src/creatures/humans/AnglerMiner.hpp	
Az "AnglerMiner" szakmájú ember osztály itt van deklarálva	215
src/creatures/humans/Builder.cpp	216
src/creatures/humans/Builder.d	216
src/creatures/humans/Builder.hpp	
Az építész szakmájú ember osztály itt van deklarálva	216
src/creatures/humans/Farmer.cpp	217
src/creatures/humans/Farmer.d	217
src/creatures/humans/Farmer.hpp	
A farmer szakmájú ember osztály itt van deklarálva	217
src/creatures/humans/Fisherman.cpp	218
src/creatures/humans/Fisherman.d	218
src/creatures/humans/Fisherman.hpp	
A halász szakmájú ember osztály itt van deklarálva	218
src/creatures/humans/Human.cpp	219
src/creatures/humans/Human.d	219
src/creatures/humans/Human.hpp	
Az alap ember osztály itt van deklarálva	219
src/creatures/humans/King.cpp	220
src/creatures/humans/King.d	220
src/creatures/humans/King.hpp	
A király szakmájú ember osztály itt van deklarálva	220
src/creatures/humans/Soldier.cpp	221
src/creatures/humans/Soldier.d	221
src/creatures/humans/Soldier.hpp	
A katona szakmájú ember osztály itt van deklarálva	221
src/creatures/humans/Stonemason.cpp	222
src/creatures/humans/Stonemason.d	222
src/creatures/humans/Stonemason.hpp	
A bányász szakmájú ember osztály itt van deklarálva	222
src/creatures/humans/Woodcutter.cpp	223
src/creatures/humans/Woodcutter.d	223
src/creatures/humans/Woodcutter.hpp	
A favágó szakmájú ember osztály itt van deklarálva	223
src/exceptions/FileExceptions.hpp	
Ebben a fájlban vannak deklarálva azok a hibák, amik az IO-hoz kapcsolódnak	226
src/exceptions/MusicLoadException.hpp	
Ebben a fájlban vannak deklarálva azok a hibák, amik az Zenéhez kapcsolódnak	227
src/exceptions/SimulationException.hpp	
Ebben a fájlban vannak deklarálva azok a hibák, amik az alap Szimulációhoz kapcsolódnak.	
Ezekből a hibákból öröklődik minden olyan hiba, amit a program kezel és dob	227
src/exceptions/WorldExceptions.hpp	
Ebben a fájlban vannak deklarálva azok a hibák, amik az Világhoz kapcsolódnak	228
src/external/gtest_lite.h	228
src/external/memtrace.cpp	238
src/external/memtrace.h	238
src/fake_sfml/fake_sfml.cpp	238
src/fake_sfml/fake_sfml.d	238
src/fake_sfml/fake_sfml.hpp	238

src/terrain_tiles/Tile.cpp	247
src/terrain_tiles/Tile.d	247
src/terrain_tiles/Tile.hpp	
A Terepkocka osztály itt van deklarálva	247
src/ui/button.cpp	249
src/ui/button.d	250
src/ui/button.hpp	
A gomb osztály itt van deklarálva	250
src/world_object/BerryBush.cpp	255
src/world_object/BerryBush.d	255
src/world_object/BerryBush.hpp	
A bokor osztály itt van deklarálva	255
src/world_object/CityCenter.cpp	255
src/world_object/CityCenter.d	256
src/world_object/CityCenter.hpp	
A városközpont osztály itt van deklarálva	256
src/world_object/House.cpp	256
src/world_object/House.d	256
src/world_object/House.hpp	
A Ház osztály itt van deklarálva	256
src/world_object/Iron.cpp	257
src/world_object/Iron.d	257
src/world_object/Iron.hpp	
A Vasérc osztály itt van deklarálva	257
src/world_object/ResourceStructure.cpp	258
src/world_object/ResourceStructure.d	258
src/world_object/ResourceStructure.hpp	
A kibányászható osztály itt van deklarálva	258
src/world_object/Stone.cpp	258
src/world_object/Stone.d	259
src/world_object/Stone.hpp	
A Kő osztály itt van deklarálva	259
src/world_object/Structure.cpp	259
src/world_object/Structure.d	259
src/world_object/Structure.hpp	
A struktúra osztály itt van deklarálva	259
src/world_object/Tree.cpp	260
src/world_object/Tree.d	260
src/world_object/Tree.hpp	
A fa osztály itt van deklarálva	260

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 entitások 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. 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`)

5.2.1. Részletes leírás

`gtest_lite`: a keretrendszer függvényinek és objektumainak névtére

5.2.2. Függvények dokumentációja

5.2.2.1. `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>

5.2.2.2. `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

5.2.2.3. eqstr()

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

5.2.2.4. eqstrcase()

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

5.2.2.5. 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.

5.2.2.6. 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.

5.2.2.7. 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.

5.2.2.8. ge()

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

5.2.2.9. gt()

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

5.2.2.10. le()

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

5.2.2.11. lt()

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


5.2.2.12. ne()

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

5.2.2.13. nestr()

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

5.3. 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.

5.3.1. Részletes leírás

Az összes struktúra ebben a névtérben van.

5.3.2. Enumerációk dokumentációja

5.3.2.1. MINERAL_TYPE

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

Enumeráció-értékek

STONE	
WOOD	
IRON	
FOOD	
HOUSING	
CITY_CENTER	

5.3.3. Függvények dokumentációja

5.3.3.1. mineral_to_string()

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

Mentést elősegítő függvények.

5.4. 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

5.4.1. Enumerációk dokumentációja

5.4.1.1. BlendMode

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

Enumeráció-értékek

None	
Alpha	
Additive	
Multiply	
BlendAdd	

5.4.2. Függvények dokumentációja

5.4.2.1. file_exists_at_path()

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

5.4.2.2. to_string()

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

5.4.3. Változók dokumentációja

5.4.3.1. BlendAdd

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

5.5. 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](#) }

5.5.1. Részletes leírás

Az összes terepkocka elem ebben a névtérben van.

5.5.2. Enumerációk dokumentációja

5.5.2.1. TILETYPE

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

Enumeráció-értékek

GRASS	
WATER	
MOUNTAIN	

5.6. 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.

5.6.1. 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 <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:

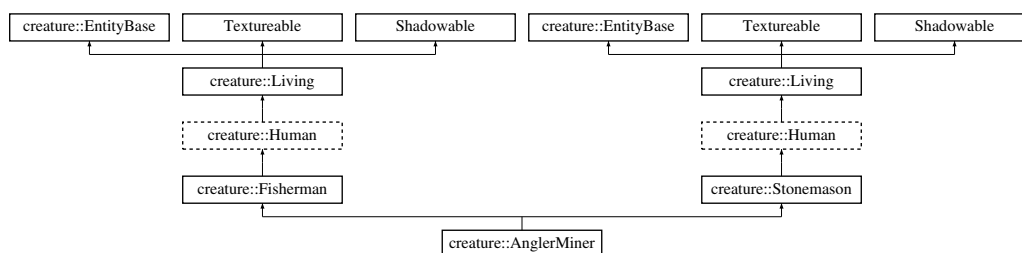
- [src/external/gtest_lite.h](#)

6.2. creature::AnglerMiner osztályreferencia

Az "AnglerMiner" szakmájú ember osztály leírása.

```
#include <AnglerMiner.hpp>
```

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

<code>x</code>	Az x koordináta.
<code>y</code>	Az y koordináta.
<code>gender_modifier</code>	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:

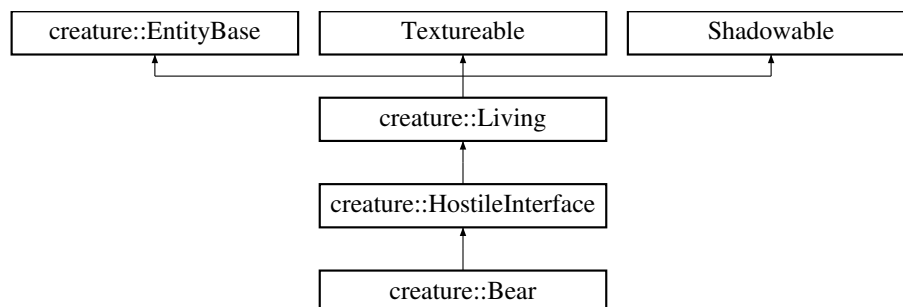
- src/creatures/humans/[AnglerMiner.hpp](#)
- src/creatures/humans/[AnglerMiner.cpp](#)

6.3. creature::Bear osztályreferencia

A medve osztály leírása.

```
#include <Bear.hpp>
```

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:

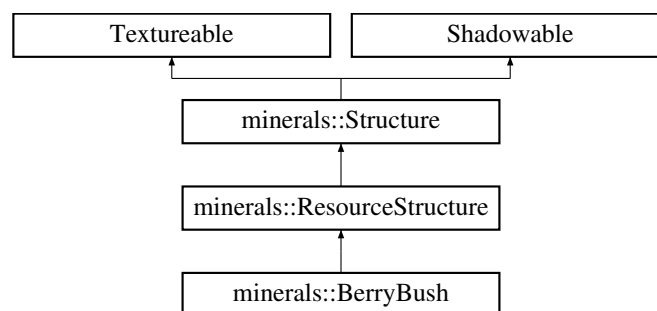
- [src/creatures/hostiles/Bear.hpp](#)
- [src/creatures/hostiles/Bear.cpp](#)

6.4. minerals::BerryBush osztályreferencia

A bokr osztály leírása. Ételt ad, ha kitermelik.

```
#include <BerryBush.hpp>
```

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

További örökölt tagok

6.4.1. Részletes leírás

A bokr osztály leírása. Ételt ad, ha kitermelik.

6.4.2. Konstruktorkok és destruktorkok dokumentációja

6.4.2.1. BerryBush()

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

Konstruktork 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]
```

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

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

6.4.3.3. 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.hpp](#)
- [src/world_object/BerryBush.cpp](#)

6.5. sf::Bound osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

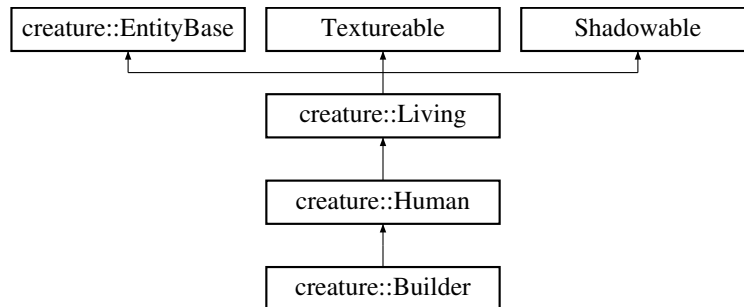
- [src/fake_sfml/fake_sfml.hpp](#)

6.6. creature::Builder osztályreferencia

Az építész szakmájú ember osztály leírása.

```
#include <Builder.hpp>
```

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:

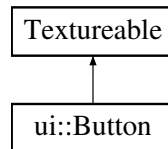
- [src/creatures/humans/Builder.hpp](#)
- [src/creatures/humans/Builder.cpp](#)

6.7. ui::Button osztályreferencia

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

```
#include <button.hpp>
```

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 leszármozottat.

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:

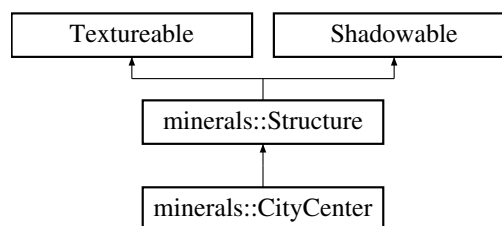
- [src/ui/button.hpp](#)
- [src/ui/button.cpp](#)

6.8. minerals::CityCenter osztályreferencia

A városközpont osztály leírása. E köré épülnek a házak.

```
#include <CityCenter.hpp>
```

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:

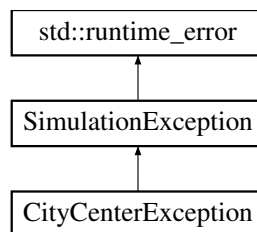
- [src/world_object/CityCenter.hpp](#)
- [src/world_object/CityCenter.cpp](#)

6.9. CityCenterException osztályreferencia

Akkor kell dobni, ha a városközpont hibásan működött.

```
#include <WorldExceptions.hpp>
```

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.hpp](#)

6.10. sf::Clock osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.11. sf::ClockTime osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.12. sf::Color osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

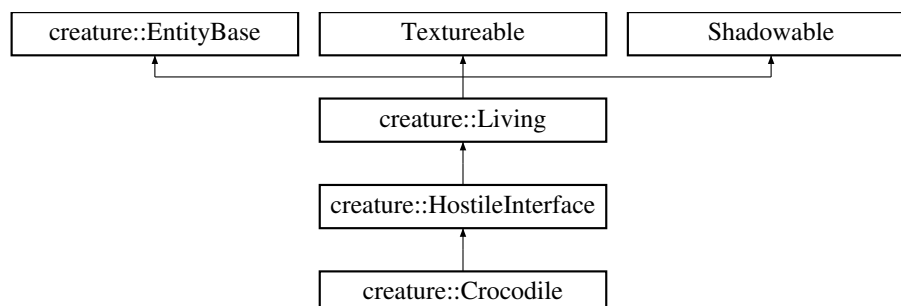
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.13. creature::Crocodile osztályreferencia

A krokodil osztály leírása.

```
#include <Crocodile.hpp>
```

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 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.
- `~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:

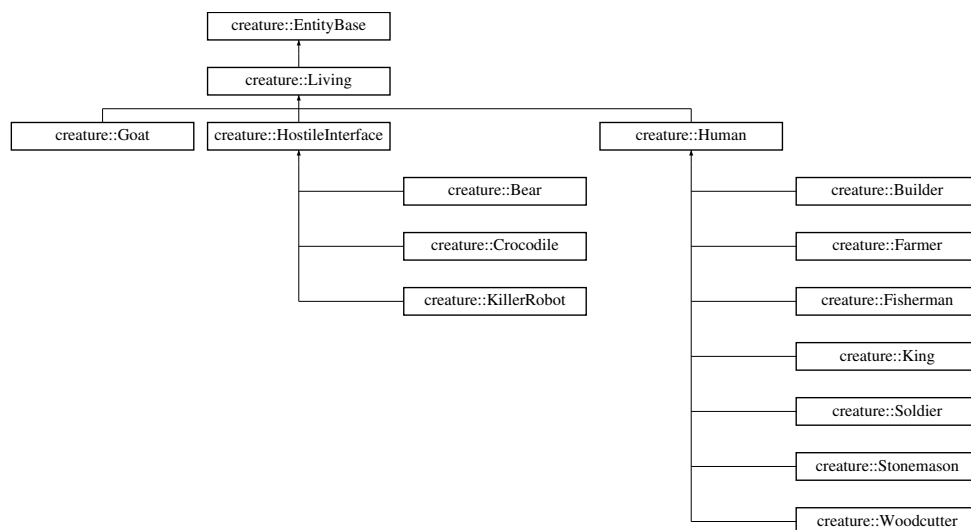
- [src/creatures/hostiles/Crocodile.hpp](#)
- [src/creatures/hostiles/Crocodile.cpp](#)

6.14. creature::EntityBase osztályreferencia

Egy alap, nem rajzolható entitás osztálya.

```
#include <EntityBase.hpp>
```

A creature::EntityBase osztály származási diagramja:



Publikus tagfüggvények

- `ENTITY_GENDER get_gender () const`
Visszaadja az entitás nemét.
- `LIVINGSTATE get_state () const`
Visszaadja az entitás belső állapotát.
- `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

- `std::string save_name = "?"`
Milyen szimbóluma legyen a mentés fájlba.
- `float death_timer =0.1f`
A halál animáció hátralévő idejét méri. Ha ez 0 akkor az entitás felszabadul és megsemmisül.
- `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 ≤ 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](#)

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_gender()

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

Visszaadja az entitás nemét.

Visszatérési érték

Az entitás neve.

6.14.3.4. get_state()

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

Visszaadja az entitás belső állapotát.

Visszatérési érték

Az entitás belső állapota.

6.14.3.5. 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`.

6.14.3.6. 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.
----------------	---------------------------------

6.14.3.7. 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.
----------------	---------------------------------

6.14.3.8. set_health()

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

Beállítja az entitás életét ez bizonyos értékre.

Paraméterek

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

6.14.3.9. 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.
----------------	---------------------------------

6.14.3.10. 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.
----------------	---------------------------------

6.14.3.11. set_state()

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

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

6.14.3.12. 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
```

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 ≤ 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 = "?"
```

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

6.15. EntityPlacer osztályreferencia

Az entitások a kattintással való lerakása.

```
#include <EntityPlacer.hpp>
```

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

6.16. sf::Event osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

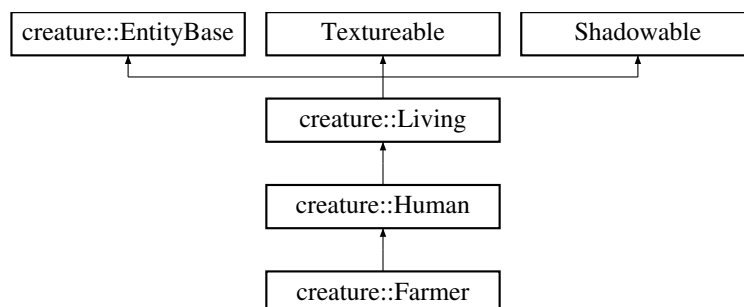
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.17. creature::Farmer osztályreferencia

A farmer szakmájú ember osztály leírása.

```
#include <Farmer.hpp>
```

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:

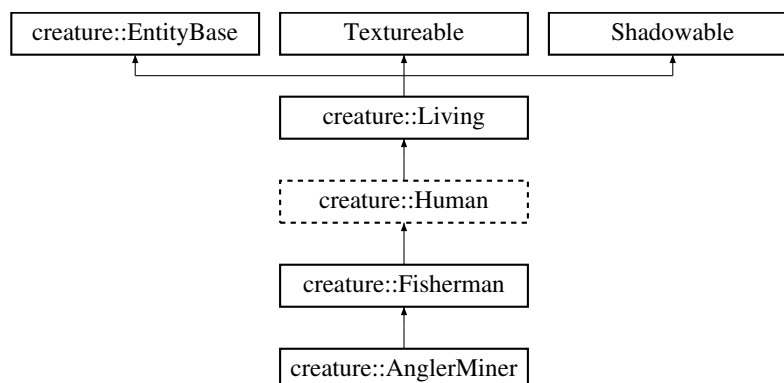
- `src/creatures/humans/Farmer.hpp`
- `src/creatures/humans/Farmer.cpp`

6.18. creature::Fisherman osztályreferencia

A halász szakmájú ember osztály leírása.

```
#include <Fisherman.hpp>
```

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 halászhat.

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

<i>x</i>	Az x koordináta.
<i>y</i>	Az y koordináta.
<i>gender_modifier</i>	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.hpp](#)
- [src/creatures/humans/Fisherman.cpp](#)

6.19. sf::FloatRect osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.20. GameConfig osztályreferencia

A világ szimulációjának leírása.

```
#include <GameConfig.hpp>
```


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`
- `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. operator=()

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

Nem szükséges a singleton pattern miatt.

6.20.3.13. read_from_config_file()

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

Beolvassa a filepath elérési útvonalról a konfigurációt.

6.20.3.14. set_config_level()

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

Beállítható, hogy mennyire bőbeszédű legyen a hibakezelés / figyelmeztetések.

6.20.3.15. 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.hpp](#)
- [src/GameConfig.cpp](#)

6.21. GameManager osztályreferencia

A világ szimulálásáért és a kirazolás irányításáért felelős osztály.

```
#include <GameManager.hpp>
```

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. Konstruktorok és destruktorok 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:

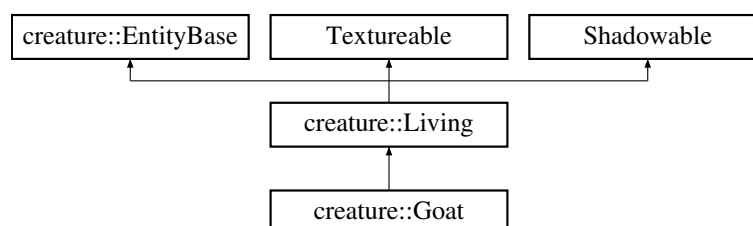
- [src/GameManager.hpp](#)
- [src/GameManager.cpp](#)

6.22. creature::Goat osztályreferencia

A kecske osztály leírása.

```
#include <Goat.hpp>
```

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 ofx, int ofy)` 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. Konstruktorkok és destruktorkok 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

x	Az x koordináta.
y	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:

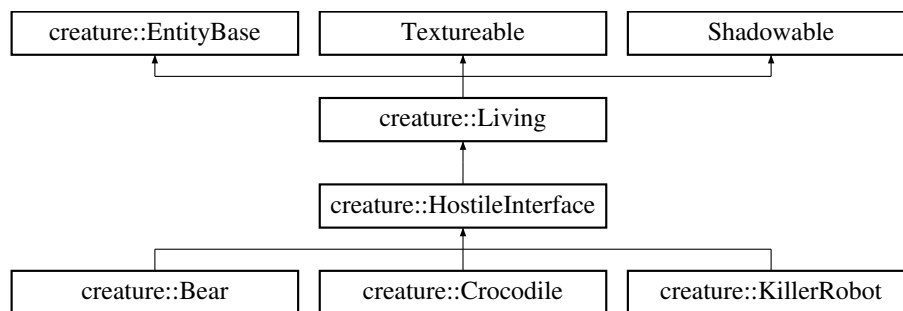
- [src/creatures/Goat.hpp](#)
- [src/creatures/Goat.cpp](#)

6.23. creature::HostileInterface osztályreferencia

A vadállat entitások interface leírása.

```
#include <HostileInterface.hpp>
```

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 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.

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.
-------------------	---

Újraimplementált 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:

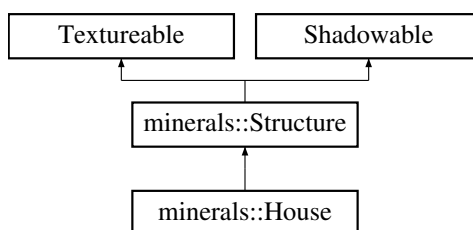
- src/creatures/[HostileInterface.hpp](#)
- src/creatures/[HostileInterface.cpp](#)

6.24. minerals::House osztályreferencia

A ház osztály leírása. Szinttől függően idéz embereket.

```
#include <House.hpp>
```

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.

Publikus attribútumok

- int [level](#)
Milyen modern a ház (1-3 -ig).
- int [stone_req](#)
Mennyi kő kell, hogy a ház elérje a következő szintet.
- int [wood_req](#)
Mennyi fa kell, hogy a ház elérje a következő szintet.
- int [iron_req](#)
Mennyi vas kell, hogy a ház elérje a következő szintet.

További örökölt tagok

6.24.1. Részletes leírás

A ház osztály leírása. Szinttől függően idéz embereket.

6.24.2. Konstruktork és destruktork dokumentációja

6.24.2.1. House()

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

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

6.24.3. Tagfüggvények dokumentációja

6.24.3.1. get_type()

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

Szimbólum, ami a fájlba mentéshez kell.

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

6.24.3.2. 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](#).

6.24.4. Adattagok dokumentációja

6.24.4.1. iron_req

```
int minerals::House::iron_req
```

Mennyi vas kell, hogy a ház elérje a következő szintet.

6.24.4.2. level

```
int minerals::House::level
```

Milyen modern a ház (1-3 -ig).

6.24.4.3. stone_req

```
int minerals::House::stone_req
```

Mennyi kő kell, hogy a ház elérje a következő szintet.

6.24.4.4. wood_req

```
int minerals::House::wood_req
```

Mennyi fa kell, hogy a ház elérje a következő szintet.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

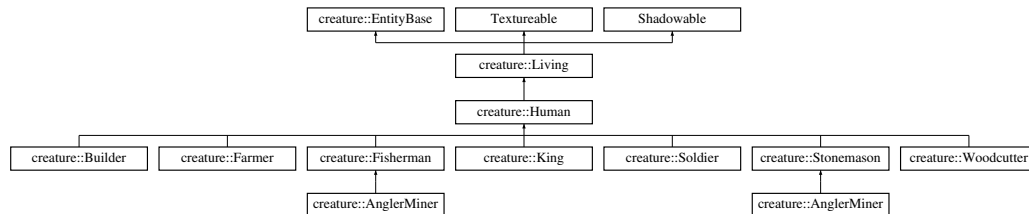
- [src/world_object/House.hpp](#)
- [src/world_object/House.cpp](#)

6.25. creature::Human osztályreferencia

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

```
#include <Human.hpp>
```

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 kirajzolási 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 emberenként 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 városa.

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 destruktorkok dokumentációja

6.25.2.1. Human() [1/2]

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

Az alap konstruktor ami leidézi 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 leidézi 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 destruktor, 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. 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.

6.25.3.6. select_texture()

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

Beállít egy textúrát ami nagyon különböző lehet emberenénkt 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.

6.25.3.7. 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](#).

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

6.26. HumanResources osztályreferencia

Az emberek által összegyűjtött erőforrások itt vannak nyilvántartva.

```
#include <HumanResources.hpp>
```

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:

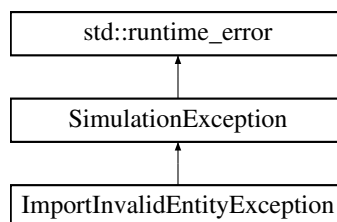
- [src/HumanResources.hpp](#)
- [src/HumanResources.cpp](#)

6.27. ImportInvalidEntityException osztályreferencia

Akkor kell dobni, ha egy entitás hibásan lett beolvasva.

```
#include <FileExceptions.hpp>
```

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:

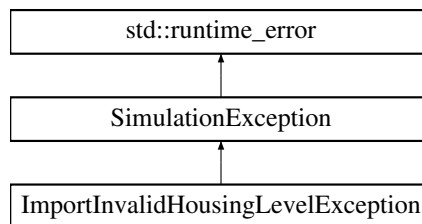
- [src/exceptions/FileExceptions.hpp](#)

6.28. ImportInvalidHousingLevelException osztályreferencia

Akkor kell dobni, ha egy ház hibásan lett beolvasva.

```
#include <FileExceptions.hpp>
```

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:

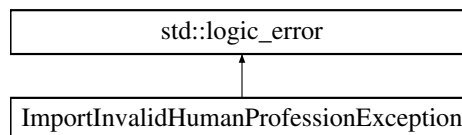
- src/exceptions/[FileExceptions.hpp](#)

6.29. ImportInvalidHumanProfessionException osztályreferencia

Akkor kell dobni, ha egy szakma hibásan lett beolvasva.

```
#include <FileExceptions.hpp>
```

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. Konstruktork és destruktorok dokumentációja

6.29.2.1. ImportInvalidHumanProfessionException()

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

Ez a dokumentáció az osztályról a következő fájl alapján készült:

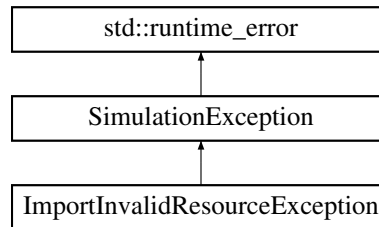
- src/exceptions/[FileExceptions.hpp](#)

6.30. ImportInvalidResourceException osztályreferencia

Akkor kell dobni, ha egy erőforrás hibásan lett beolvasva.

```
#include <FileExceptions.hpp>
```

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. Konstruktorkok és destruktorkok 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.hpp](#)

6.31. sf::IntRect osztályreferencia

```
#include <fake_sfml.hpp>
```

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. Konstruktorkok és destruktorkok 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:

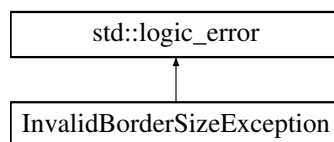
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.32. InvalidBorderSizeException osztályreferencia

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

```
#include <WorldExceptions.hpp>
```

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:

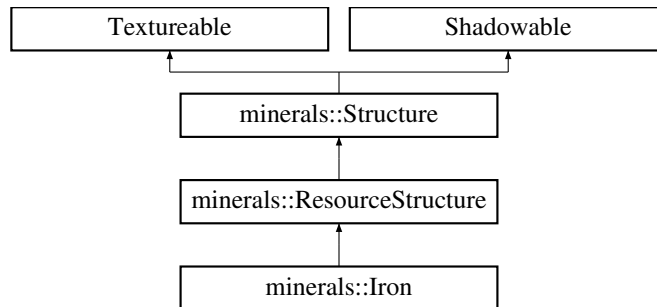
- [src/exceptions/WorldExceptions.hpp](#)

6.33. minerals::Iron osztályreferencia

A vasérc osztály leírása. Vasat ad, amikor kitermelik.

```
#include <Iron.hpp>
```

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.
- bool [harvest](#) () override
Tisztán virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.

További örökölt tagok

6.33.1. Részletes leírás

A vasérc osztály leírása. Vasat ad, amikor kitermelik.

6.33.2. Konstruktorok és destruktorok dokumentációja

6.33.2.1. Iron()

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

Konstruktor 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. harvest()

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

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

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.hpp](#)
- [src/world_object/Iron.cpp](#)

6.34. sf::Keyboard osztályreferencia

```
#include <fake_sfml.hpp>
```

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	
Num8	
Num9	
Num0	

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:

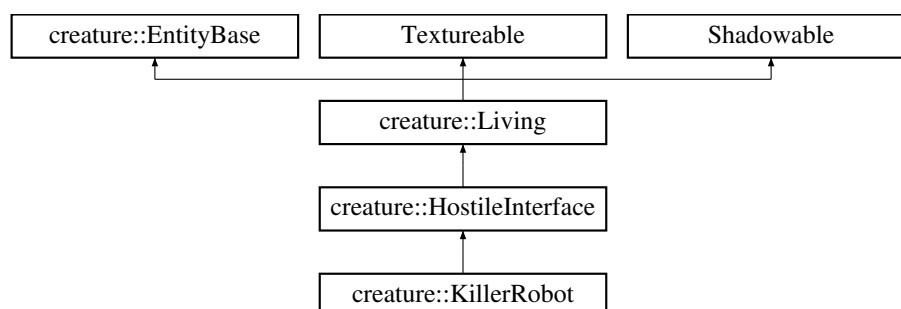
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.35. creature::KillerRobot osztályreferencia

A gyilkos robot osztály leírása.

```
#include <KillerRobot.hpp>
```

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 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.
- [~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. Konstruktorkok és destruktorkok 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 destruktork.

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:

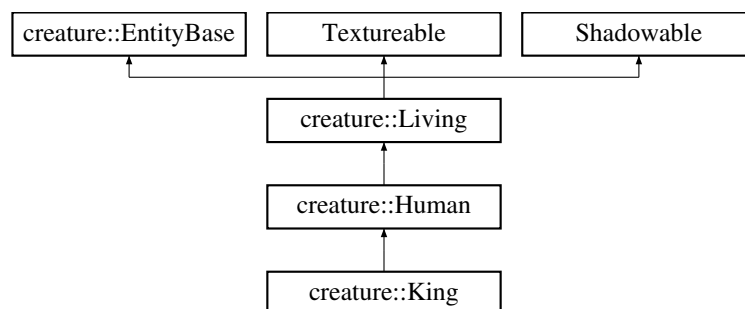
- [src/creatures/hostiles/KillerRobot.hpp](#)
- [src/creatures/hostiles/KillerRobot.cpp](#)

6.36. creature::King osztályreferencia

A király szakmájú ember osztály leírása.

```
#include <King.hpp>
```

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 örülni bolyong a világba.

6.36.2. Konstruktorok és destruktorok 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 neme.

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:

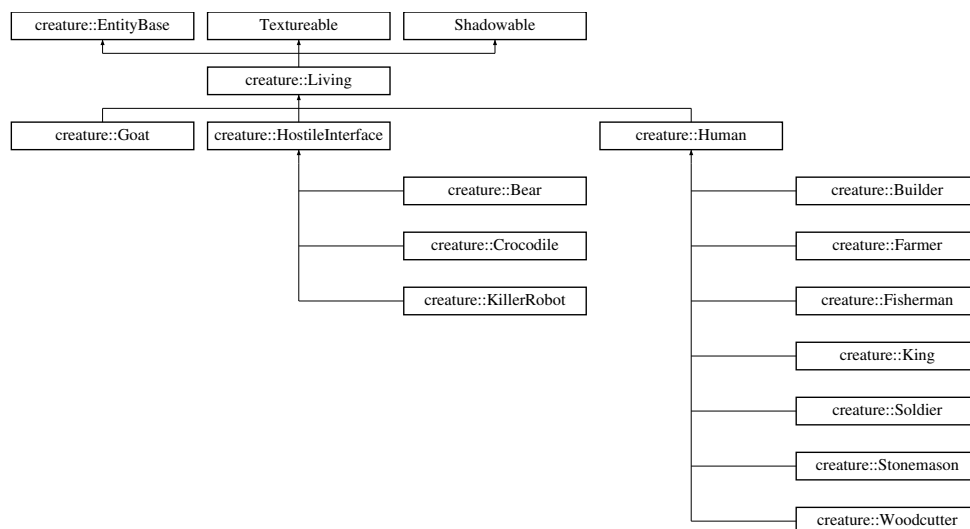
- [src/creatures/humans/King.hpp](#)
- [src/creatures/humans/King.cpp](#)

6.37. creature::Living osztályreferencia

Az élő entitások interface leírása.

```
#include <Living.hpp>
```

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
Kirajzolja 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". Eltávolja, 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.

Újraimplementáló 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.
-------------------	---

Újrimplementáló 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

<i>deltaTime</i>	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.hpp](#)
- [src/creatures/Living.cpp](#)

6.38. creature::LivingTexture osztályreferencia

Az élő entitások kinézetének adatai.

```
#include <EntityUtils.hpp>
```

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 entitások 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.hpp](#)

6.39. sf::Mouse osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.40. sf::Music osztályreferencia

```
#include <fake_sfml.hpp>
```

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. Konstruktork és destruktork 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:

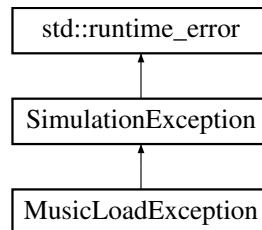
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.41. MusicLoadException osztályreferencia

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

```
#include <MusicLoadException.hpp>
```

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.hpp](#)

6.42. MusicPlayer osztályreferencia

A zene játszó osztály leírása.

```
#include <MusicPlayer.hpp>
```

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 játszó 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.hpp](#)
- [src/MusicPlayer.cpp](#)

6.43. ObjectRegistry osztályreferencia

Az entitások és más világ objektumok lerakásának intézéséért felelős osztály.

```
#include <EntityPlacer.hpp>
```

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

6.44. gtest_lite::ostreamRedir osztályreferencia

```
#include <gtest_lite.h>
```

Publikus tagfüggvények

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

6.44.1. Részletes leírás

Segédsablon ostream átirányításához A destruktorkor visszaállít

6.44.2. Konstruktorkor és destruktorkor 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/[gtest_lite.h](#)

6.45. PostProcessor osztályreferencia

A grafikus szépítő osztály leírása.

```
#include <PostProcessor.hpp>
```

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)
Beállítja egy képnek egy új textúráját.
- 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. Konstruktork és destruktork 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:

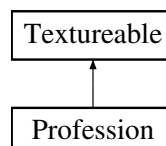
- [src/PostProcessor.hpp](#)
- [src/PostProcessor.cpp](#)

6.46. Profession osztályreferencia

A szakma osztály leírása.

```
#include <Profession.hpp>
```

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
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.
- 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.hpp](#)
- [src/Profession.cpp](#)

6.47. RandomGenerator osztályreferencia

Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály.

```
#include <Random_Gen.hpp>
```

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:

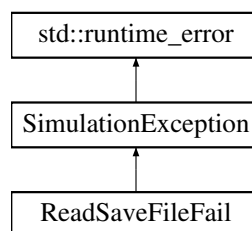
- [src/Random_Gen.hpp](#)
- [src/Random_Gen.cpp](#)

6.48. ReadSaveFileFail osztályreferencia

Akkor kell dobni, ha egy IO mentés, importálás vagy törlés hibás.

```
#include <FileExceptions.hpp>
```

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. Konstruktorkok és destruktorkok 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.hpp](#)

6.49. sf::RectangleShape osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.50. sf::RenderStates osztályreferencia

```
#include <fake_sfml.hpp>
```

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 destruktorkok 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.hpp`
- `src/fake_sfml/fake_sfml.cpp`

6.51. sf::RenderWindow osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

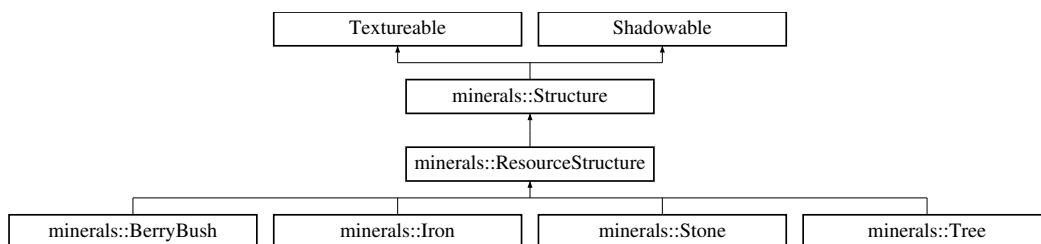
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.52. minerals::ResourceStructure osztályreferencia

Az erőforrás struktúra osztály leírása.

```
#include <ResourceStructure.hpp>
```

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](#) ()=0
Tisztán virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.
- virtual [~ResourceStructure](#) ()=default
Virtuális destruktork.

Védett attribútumok

- float [inner_timer](#)
Az idéződés óta eltelt idő.
- bool [harvested](#)
Ki van-e termelve az erőforrás.

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. Konstruktorkok és destruktorkok dokumentációja

6.52.2.1. ResourceStructure()

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

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

6.52.2.2. ~ResourceStructure()

```
virtual minerals::ResourceStructure::~~ResourceStructure ( ) [virtual], [default]
```

Virtuális destruktork.

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. harvest()

```
virtual bool minerals::ResourceStructure::harvest ( ) [pure virtual]
```

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

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 idéződés óta eltelt idő.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/world_object/[ResourceStructure.hpp](#)
- src/world_object/[ResourceStructure.cpp](#)

6.53. RoleOption struktúráreferencia

Segít abba, hogy OOP-sebben lehessen az embernek véletlenszerűen új szakmát adni.

```
#include <SaveHelpers.hpp>
```

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.hpp](#)

6.54. SaveHelper osztályreferencia

Factory-k.

```
#include <SaveHelpers.hpp>
```

Publikus típusok

- using [CreatureFactory](#) = std::function< [creature::Living](#) *(int, int)>
- using [HumanFactory](#) = std::function< [creature::Human](#) *(int, int, [creature::ENTITY_GENDER](#))>
- using [ResourceFactory](#) = std::function< [minerals::ResourceStructure](#) *(int, int)>

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](#) ()

6.54.1. Részletes leírás

Factory-k.

6.54.2. Típusdefiníció-tagok dokumentációja

6.54.2.1. CreatureFactory

```
using SaveHelper::CreatureFactory = std::function<creature::Living*(int, int)>
```


6.54.2.2. HumanFactory

```
using SaveHelper::HumanFactory = std::function<creature::Human*(int, int, creature::ENTITY_GENDER)>
```

6.54.2.3. ResourceFactory

```
using SaveHelper::ResourceFactory = std::function<minerals::ResourceStructure*(int, int)>
```

6.54.3. Tagfüggvények dokumentációja

6.54.3.1. getCreatureFactory()

```
const std::unordered_map< std::string, SaveHelper::CreatureFactory > & SaveHelper::getCreature↵  
Factory ( ) [static]
```

6.54.3.2. getHumanFactory()

```
const std::unordered_map< std::string, SaveHelper::HumanFactory > & SaveHelper::getHuman↵  
Factory ( ) [static]
```

6.54.3.3. getResourceFactory()

```
const std::unordered_map< std::string, SaveHelper::ResourceFactory > & SaveHelper::getResource↵  
Factory ( ) [static]
```

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- src/[SaveHelpers.hpp](#)
- src/[SaveHelpers.cpp](#)

6.55. SaveManager osztályreferencia

A fájl menedzseléshez szolgáló osztály leírása.

```
#include <SaveManager.hpp>
```

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 útvonala.
-------------	----------------------

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

<code>world</code>	Referencia a világra.
--------------------	-----------------------

6.55.3.3. saveFile()

```
void SaveManager::saveFile (
    World & world )
```

Elment egy világot a fájlba.

Paraméterek

<code>world</code>	Referencia a világra.
--------------------	-----------------------

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

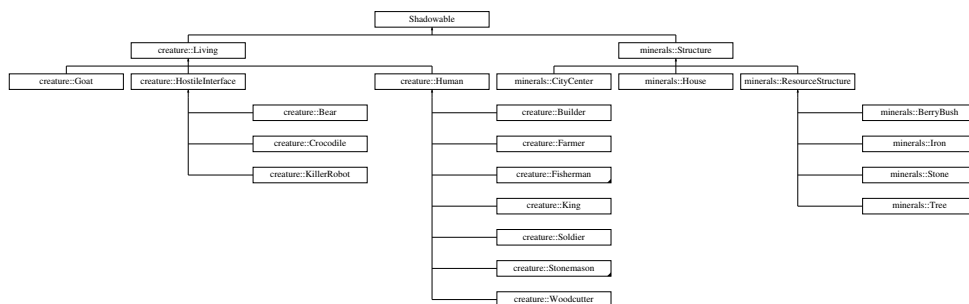
- [src/SaveManager.hpp](#)
- [src/SaveManager.cpp](#)

6.56. Shadowable osztályreferencia

Az árnyékoláshoz szükséges interface.

```
#include <Shadowable.hpp>
```

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 erősségre.
- 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 erősségéhez.
- 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:

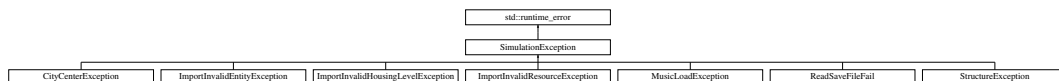
- [src/Shadowable.hpp](#)
- [src/Shadowable.cpp](#)

6.57. SimulationException osztályreferencia

Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik.

```
#include <SimulationException.hpp>
```

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:

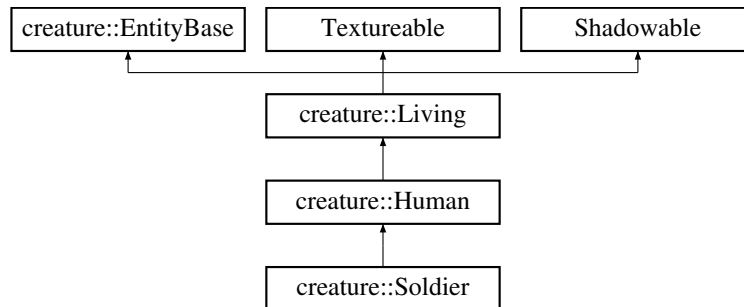
- [src/exceptions/SimulationException.hpp](#)

6.58. creature::Soldier osztályreferencia

A katona szakmájú ember osztály leírása.

```
#include <Soldier.hpp>
```

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. Konstruktorkok 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.hpp](#)
- [src/creatures/humans/Soldier.cpp](#)

6.59. sf::Sound osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp`
- `src/fake_sfml/fake_sfml.cpp`

6.60. `sf::SoundBuffer` osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp
- src/fake_sfml/fake_sfml.cpp

6.61. SoundPlayer osztályreferencia

A hanglejátszó osztály leírása.

```
#include <SoundPlayer.hpp>
```

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 útvonál.
-----------------	---------------------

6.61.2.2. play_sound()

```
void SoundPlayer::play_sound (
    const std::string & filename )
```

Lejátszik egy hangot az elérési útvonálról. Ha még nem volt ez betöltve akkor először betölti.

Paraméterek

<i>filename</i>	Az elérési útvonál.
-----------------	---------------------

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.hpp](#)
- [src/SoundPlayer.cpp](#)

6.62. sf::SoundSource osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.63. sf::Sprite osztályreferencia

```
#include <fake_sfml.hpp>
```

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. Konstruktorkok és destruktorkok 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:

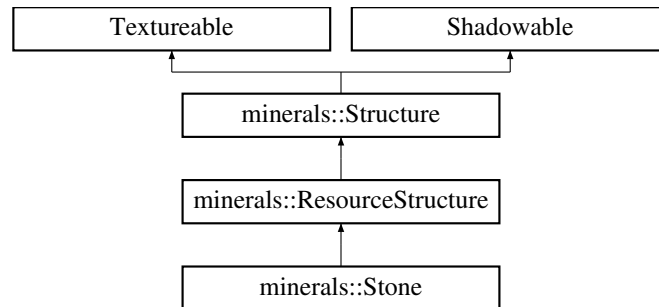
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.64. minerals::Stone osztályreferencia

A kő osztály leírása. követ ad, amikor kitermelik.

```
#include <Stone.hpp>
```

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.
- bool `harvest ()` override
Tisztán virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.

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. `harvest()`

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

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

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:

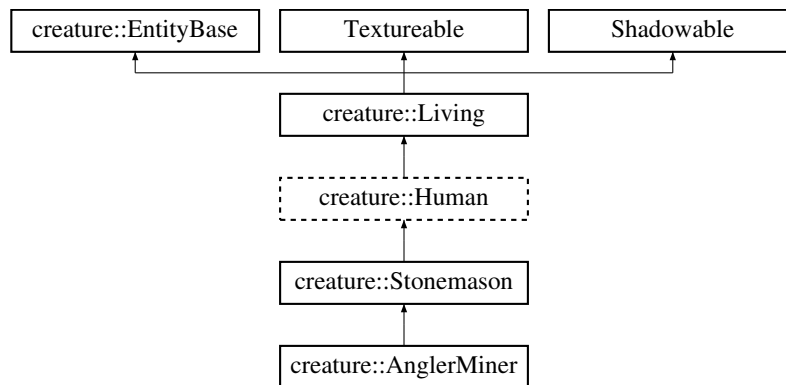
- [src/world_object/Stone.hpp](#)
- [src/world_object/Stone.cpp](#)

6.65. `creature::Stonemason` osztályreferencia

A bányász szakmájú ember osztály leírása.

```
#include <Stonemason.hpp>
```

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?

Újraimplementált 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:

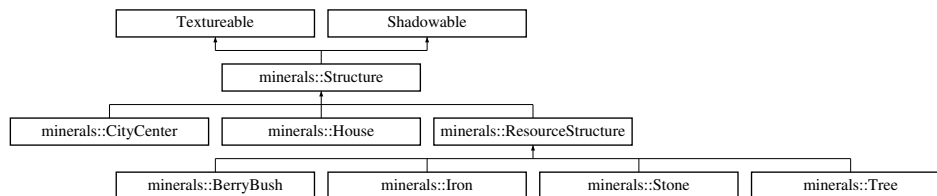
- [src/creatures/humans/Stonemason.hpp](#)
- [src/creatures/humans/Stonemason.cpp](#)

6.66. minerals::Structure osztályreferencia

A struktúra osztály leírása.

```
#include <Structure.hpp>
```

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

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

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:

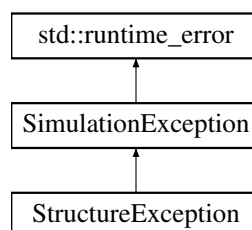
- [src/world_object/Structure.hpp](#)
- [src/world_object/Structure.cpp](#)

6.67. StructureException osztályreferencia

Akkor kell dobni, ha egy struktúra hibásan működött.

```
#include <WorldExceptions.hpp>
```

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.hpp](#)

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>ahight</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 )
```

Kirajolja 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 struktúráreferencia

```
#include <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/gtest_lite.h`

6.70. sf::Texture osztályreferencia

```
#include <fake_sfml.hpp>
```

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 destruktorkok 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:

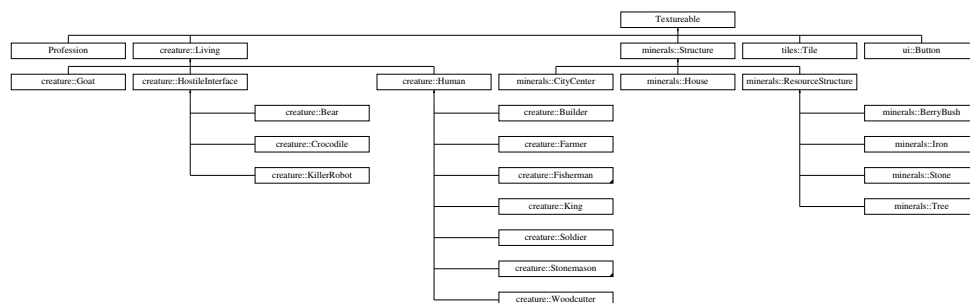
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.71. Textureable osztályreferencia

Egy interface, ami a textúrázáshoz kell.

```
#include <Textureable.hpp>
```

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.hpp](#)

6.72. TextureManager osztályreferencia

A Textúra kezelő osztály.

```
#include <TextureManager.hpp>
```

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:

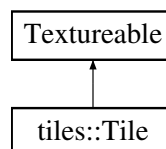
- [src/TextureManager.hpp](#)
- [src/TextureManager.cpp](#)

6.73. tiles::Tile osztályreferencia

A terepkocka osztály leírása.

```
#include <Tile.hpp>
```

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 leszármazottat.
---------------	---

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.hpp](#)
- src/terrain_tiles/[Tile.cpp](#)

6.74. sf::Transform osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

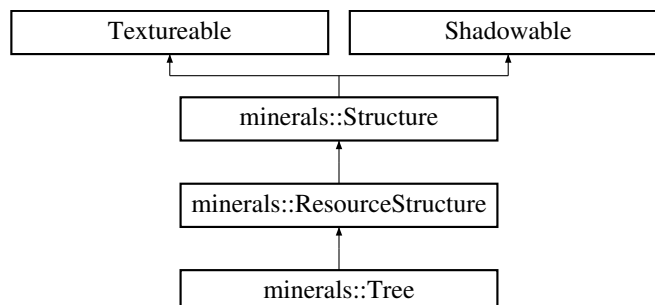
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.75. minerals::Tree osztályreferencia

A fa osztály leírása. Fát ad, ha kitermelik.

```
#include <Tree.hpp>
```

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.
- bool [harvest](#) () override
Tisztán virtuális metódus. Ez leírja mi történik, hogy ha kitermelik ezt az erőforrást.

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. harvest()

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

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

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.hpp](#)
- src/world_object/[Tree.cpp](#)

6.76. sf::Vector2f osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.77. sf::Vector2i osztályreferencia

```
#include <fake_sfml.hpp>
```

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.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.78. sf::VideoMode osztályreferencia

```
#include <fake_sfml.hpp>
```

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:

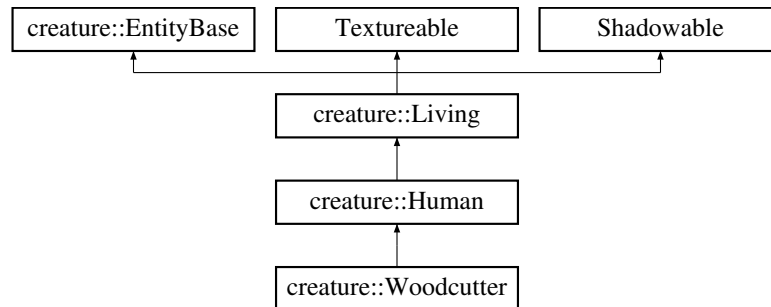
- [src/fake_sfml/fake_sfml.hpp](#)
- [src/fake_sfml/fake_sfml.cpp](#)

6.79. creature::Woodcutter osztályreferencia

A favágó szakmájú ember osztály leírása.

```
#include <Woodcutter.hpp>
```

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?

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

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

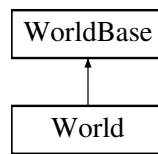
- [src/creatures/humans/Woodcutter.hpp](#)
- [src/creatures/humans/Woodcutter.cpp](#)

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` ()
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 entitásokat 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.

Barátok

- std::ostream & `operator<<` (std::ostream &os, `World` &w)
A világ adatait segíti kimenteni egy folyamba.
- std::ifstream & `operator>>` (std::ifstream &in, `World` &w)
Egy folyamból tölti fel a világot új adatokkal.

További örökölt tagok

6.80.1. Részletes leírás

A világ osztály leírása.

Tárolja az erőforrásokat, embereket, entitásokat, 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 ( )
```

Segédfüggvény a világ törléséhez. Felszabadítja az entitásokat, erőforrásokat.

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 nagysághoz.

Visszatérési érték

A határ mérete.

6.80.3.4. get_border_width()

```
int World::get_border_width ( ) const
```

Egy getter a horizontális kamera határ nagysághoz.

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 nagysághoz.

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

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

6.80.4. Barát és kapcsolódó függvények dokumentációja

6.80.4.1. operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    World & w ) [friend]
```

A világ adatait segíti kimenteni egy folyamba.

6.80.4.2. operator>>

```
std::ifstream& operator>> (
    std::ifstream & in,
    World & w ) [friend]
```

Egy folyamból tölti fel a világot új adatokkal.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

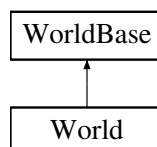
- src/[World.hpp](#)
- src/[World.cpp](#)

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, bokor, 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

<i>x</i>	Az x koordináta.
<i>y</i>	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

<i>excluded_type</i>	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

<i>suitable_tile</i>	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

<i>atype</i>	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

<i>x</i>	Oszlop index.
<i>y</i>	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 <i>x</i> koordináta.
<i>y</i>	Az <i>y</i> 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. YAMLParser osztályreferencia

Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.

```
#include <YAMLParser.hpp>
```

Publikus tagfüggvények

- [YAMLParser \(\)](#)
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)
Arra szolgál, hogy a beolvasott fájl tokenjeit kívülről is el lehet érni.

6.82.1. Részletes leírás

Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.

A YML specifikációnak megfelelően be tud olvasni YML fájlokat (.yaml).

6.82.2. Konstruktorok és destruktorok dokumentációja

6.82.2.1. YAMLParser()

```
YAMLParser::YAMLParser ( )
```

Az alap konstruktor, semmit nem csinál csak jelzi.

6.82.3. Tagfüggvények dokumentációja

6.82.3.1. get_value_of_key()

```
std::string YAMLParser::get_value_of_key (
    const std::string & key )
```

Arra szolgál, hogy a beolvasott fájl tokenjeit kívülről is el lehet érni.

6.82.3.2. parse_file()

```
bool YAMLParser::parse_file (
    const std::string & filepath )
```

Beolvassa a YML fájlt. Vigyáz a kommentekre.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- [src/YAMLParser.hpp](#)
- [src/YAMLParser.cpp](#)

7. fejezet

Fájlok dokumentációja

7.1. src/creatures/EntityBase.cpp fájlreferencia

```
#include "EntityBase.hpp"
```

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.2. src/creatures/EntityBase.d fájlreferencia

7.3. src/creatures/EntityBase.hpp fájlreferencia

```
#include "EntityUtils.hpp"  
#include "../Utils.hpp"
```

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.hpp fájlreferencia

```
#include <string>
#include <iostream>
#include "../GameConfig.hpp"
```

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.hpp"
#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.hpp fájlreferencia

A Kecske osztály itt van deklarálva.

```
#include "Living.hpp"
#include "../Random_Gen.hpp"
#include "../Utils.hpp"
```


Osztályok

- class `creature::Goat`

A kecske osztály leírása.

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.7.1. Részletes leírás

A Kecske osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.8. src/creatures/HostileInterface.cpp fájlreferencia

```
#include "HostileInterface.hpp"
#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.hpp fájlreferencia

A vadállat interface itt van deklarálva.

```
#include "Living.hpp"
```

Osztályok

- class [creature::HostileInterface](#)

A vadállat entitások interface leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.10.1. Részletes leírás

A vadállat interface itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.11. src/creatures/hostiles/Bear.cpp fájlreferencia

```
#include "Bear.hpp"
#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.hpp fájlreferencia

A Medve osztály itt van deklarálva.

```
#include "../HostileInterface.hpp"
#include "../Random_Gen.hpp"
#include "../Utils.hpp"
```

Osztályok

- class `creature::Bear`

A medve osztály leírása.

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.13.1. Részletes leírás

A Medve osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.14. src/creatures/hostiles/Crocodile.cpp fájlreferencia

```
#include "Crocodile.hpp"  
#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.hpp fájlreferencia

A krokodil osztály itt van deklarálva.

```
#include "../HostileInterface.hpp"  
#include "../Random_Gen.hpp"  
#include "../Utils.hpp"
```

Osztályok

- class [creature::Crocodile](#)

A krokodil osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.16.1. Részletes leírás

A krokodil osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.17. src/creatures/hostiles/KillerRobot.cpp fájlreferencia

```
#include "KillerRobot.hpp"  
#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.hpp fájlreferencia

A Gyilkos Robot osztály itt van deklarálva.

```
#include "../HostileInterface.hpp"  
#include "../Random_Gen.hpp"  
#include "../Utils.hpp"
```

Osztályok

- class `creature::KillerRobot`

A gyilkos robot osztály leírása.

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.19.1. Részletes leírás

A Gyilkos Robot osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.20. src/creatures/humans/AnglerMiner.cpp fájlreferencia

```
#include "AnglerMiner.hpp"  
#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.hpp fájlreferencia

Az "AnglerMiner" szakmájú ember osztály itt van deklarálva.

```
#include "Fisherman.hpp"  
#include "Stonemason.hpp"
```

Osztályok

- class [creature::AnglerMiner](#)

Az "AnglerMiner" szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.22.1. Részletes leírás

Az "AnglerMiner" szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.23. src/creatures/humans/Builder.cpp fájlreferencia

```
#include "Builder.hpp"  
#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.hpp fájlreferencia

Az építész szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```

Osztályok

- class `creature::Builder`

Az építész szakmájú ember osztály leírása.

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.25.1. Részletes leírás

Az építész szakmájú ember osztály itt van deklarálv.

Szerző

Funk Gábor

Dátum

2025-04-21

7.26. src/creatures/humans/Farmer.cpp fájlreferencia

```
#include "Farmer.hpp"  
#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.hpp fájlreferencia

A farmer szakmájú ember osztály itt van deklarálv.

```
#include "Human.hpp"
```

Osztályok

- class [creature::Farmer](#)

A farmer szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.28.1. Részletes leírás

A farmer szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.29. src/creatures/humans/Fisherman.cpp fájlreferencia

```
#include "Fisherman.hpp"  
#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.hpp fájlreferencia

A halász szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```


Osztályok

- class [creature::Fisherman](#)

A halász szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.31.1. Részletes leírás

A halász szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.32. src/creatures/humans/Human.cpp fájlreferencia

```
#include "Human.hpp"  
#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.hpp fájlreferencia

Az alap ember osztály itt van deklarálva.

```
#include "../Living.hpp"  
#include "../Random_Gen.hpp"  
#include "../Profession.hpp"  
#include "../world_object/CityCenter.hpp"  
#include "../Utils.hpp"
```

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.

7.34.1. Részletes leírás

Az alap ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.35. src/creatures/humans/King.cpp fájlreferencia

```
#include "King.hpp"  
#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.hpp fájlreferencia

A király szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```

Osztályok

- class [creature::King](#)

A király szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.37.1. Részletes leírás

A király szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.38. src/creatures/humans/Soldier.cpp fájlreferencia

```
#include "Soldier.hpp"  
#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.hpp fájlreferencia

A katona szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```

Osztályok

- class [creature::Soldier](#)

A katona szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.40.1. Részletes leírás

A katona szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.41. src/creatures/humans/Stonemason.cpp fájlreferencia

```
#include "Stonemason.hpp"  
#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.hpp fájlreferencia

A bányász szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```

Osztályok

- class [creature::Stonemason](#)

A bányász szakmájú ember osztály leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.43.1. Részletes leírás

A bányász szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.44. src/creatures/humans/Woodcutter.cpp fájlreferencia

```
#include "Woodcutter.hpp"  
#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.hpp fájlreferencia

A favágó szakmájú ember osztály itt van deklarálva.

```
#include "Human.hpp"
```

Osztályok

- class `creature::Woodcutter`

A favágó szakmájú ember osztály leírása.

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.46.1. Részletes leírás

A favágó szakmájú ember osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.47. src/creatures/Living.cpp fájlreferencia

```
#include "Living.hpp"
```

Névterek

- `creature`

Az összes élőlény és entitás ebben a névtérben van.

7.48. src/creatures/Living.d fájlreferencia

7.49. src/creatures/Living.hpp fájlreferencia

Az élő interface itt van deklarálva.

```
#include "../Textureable.hpp"
#include "../TextureManager.hpp"
#include "../GameConfig.hpp"
#include "../Shadowable.hpp"
#include <string>
#include <iostream>
#include "EntityBase.hpp"
```

Osztályok

- class [creature::Living](#)

Az élő entitások interface leírása.

Névterek

- [creature](#)

Az összes élőlény és entitás ebben a névtérben van.

7.49.1. Részletes leírás

Az élő interface itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.50. src/EntityPlacer.cpp fájlreferencia

```
#include "EntityPlacer.hpp"
```

7.51. src/EntityPlacer.d fájlreferencia

7.52. src/EntityPlacer.hpp fájlreferencia

Az entitások lerakásáért felelős osztály itt van deklarálva.

```
#include <string>
#include <unordered_map>
#include <algorithm>
#include "World.hpp"
#include "GameConfig.hpp"
#include "creatures/Living.hpp"
#include "creatures/humans/Human.hpp"
#include "creatures/humans/King.hpp"
#include "creatures/Goat.hpp"
#include "creatures/hostiles/Crocodile.hpp"
#include "creatures/hostiles/Bear.hpp"
#include "creatures/hostiles/KillerRobot.hpp"
#include "world_object/BerryBush.hpp"
#include "world_object/Stone.hpp"
#include "world_object/Tree.hpp"
#include "world_object/Iron.hpp"
#include "Utils.hpp"
```

Osztályok

- class [ObjectRegistry](#)
Az entitások és más világ objektumok lerakásának intézéséért felelős osztály.
- class [EntityPlacer](#)
Az entitások a kattintással való lerakása.

7.52.1. Részletes leírás

Az entitások lerakásáért felelős osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.53. src/exceptions/FileExceptions.hpp fájlreferencia

Ebben a fájlban vannak deklarálva azok a hibák, amik az IO-hoz kapcsolódnak.

```
#include "SimulationException.hpp"
```

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.

7.53.1. Részletes leírás

Ebben a fájlban vannak deklarálva azok a hibák, amik az IO-hoz kapcsolódnak.

Szerző

Funk Gábor

Dátum

2025-04-21

7.54. src/exceptions/MusicLoadException.hpp fájlreferencia

Ebben a fájlban vannak deklarálva azok a hibák, amik az Zenéhez kapcsolódnak.

```
#include "SimulationException.hpp"
```

Osztályok

- class [MusicLoadException](#)

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

7.54.1. Részletes leírás

Ebben a fájlban vannak deklarálva azok a hibák, amik az Zenéhez kapcsolódnak.

Szerző

Funk Gábor

Dátum

2025-04-21

7.55. src/exceptions/SimulationException.hpp fájlreferencia

Ebben a fájlban vannak deklarálva azok a hibák, amik az alap Szimulációhoz kapcsolódnak. Ezekből a hibákból öröklődik minden olyan hiba, amit a program kezel és dob.

```
#include <stdexcept>
#include <string>
```

Osztályok

- class [SimulationException](#)

Akkor kell dobni, ha egy szimulációs elem hibásan viselkedik.

7.55.1. Részletes leírás

Ebben a fájlban vannak deklarálva azok a hibák, amik az alap Szimulációhoz kapcsolódnak. Ezekből a hibákból öröklődik minden olyan hiba, amit a program kezel és dob.

Szerző

Funk Gábor

Dátum

2025-04-21

7.56. src/exceptions/WorldExceptions.hpp fájlreferencia

Ebben a fájlban vannak deklarálva azok a hibák, amik az Világhoz kapcsolódnak.

```
#include "SimulationException.hpp"
```

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.

7.56.1. Részletes leírás

Ebben a fájlban vannak deklarálva azok a hibák, amik az Világhoz kapcsolódnak.

Szerző

Funk Gábor

Dátum

2025-04-21

7.57. src/external/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 névtére

Makródefiníciók

- `#define TEST(C, N) do { gtest_lite::test.begin(#C"."#N);`
- `#define END gtest_lite::test.end(); } while (false);`
Tesztelés 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 *) eltéréset tesztelő makró*
- `#define EXPECT_STRCASEEQ(expected, actual) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::eqstrcase, __FILE__, __LINE__, "EXPECT_STRCASEEQ(" #expected ", " #actual ")", "etalon")`
*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 *) eltéréset 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 ")", "etalon")`
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 ")", "etalon")`
Környezeti változóhoz hasonlít – ilyen nincs a gtest-ben (kisbetű/nagybetű azonos)
- `#define ASSERT_EQ(expected, actual) gtest_lite::ASSERT_(expected, actual, gtest_lite::eq, "ASSERT_EQ")`
Azonosságot elváró makró
- `#define ASSERT_NO_THROW(statement)`
Nem várunk kivételt.
- `#define CREATE_Has_(X)`
- `#define CREATE_Has_fn_(X, S)`
- `#define EXPECT_THROW(statement, exp, act)`
EXPECT_THROW: kivételkezelés.
- `#define ASSERT_THROW(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="elvart", 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)`

7.57.1. Részletes leírás

(v4/2022)

Google gtest keretrendszerhez hasonló rendszer. Sz.l. 2015., 2016., 2017. (*Has_X*) Sz.l. 2018 (*template*), *ENDM*, *ENDMsg*, *nullptr_t* Sz.l. 2019 *singleton* Sz.l. 2021 *ASSERT...*, *STRCASE...* Sz.l. 2021 *EXPECT_REGEX*, *CREATE_Has_fn*, *cmp w. NULL*, *EXPECT_param* fix V.B., Sz.l. 2022 *almostEQ* fix, Sz.l. 2022. *EXPECT_THROW* fix

A tesztelés legalapvetőbb funkcióit támogató függvények és makrók. Nem szálbiztos megvalósítás.

Szabadon felhasználható, bővíthető.

Használati példa: Teszteljük az $f(x)=2*x$ függvényt: `int f(int x) { return 2*x; }`

```
int main() { TEST(TeszEsetNeve, TesztNeve) EXPECT_EQ(0, f(0)); EXPECT_EQ(4, f(2)) << "A függvény hibás
eredményt adott" << std::endl; ... END ... // Fatális hiba esetén a tesztelés nem fut tovább. Ezek az AS-
SERT... makrók. // Nem lehet a kiírásukhoz további üzenetet fűzni. PL: TEST(TeszEsetNeve, TesztNeve)
ASSERT_NO_THROW(f(0)); // itt nem lehet << "duma" EXPECT_EQ(4, f(2)) << "A függvény hibás eredményt
adott" << std::endl; ... END ...
```

A működés részleteinek megértése szorgalmi feladat.

7.57.2. Makródefiníciók dokumentációja

7.57.2.1. ADD_FAILURE

```
#define ADD_FAILURE( ) gtest_lite::test.expect(false, __FILE__, __LINE__, "ADD_FAILURE()",
true)
```

Sikertelen teszt makrója.

7.57.2.2. 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; }
```

7.57.2.3. ASSERT_EQ

```
#define ASSERT_EQ(
    expected,
    actual ) gtest_lite::ASSERT_(expected, actual, gtest_lite::eq, "ASER_EQ")
```

Azonosságot elváró makró

ASSERT típusú ellenőrzések. CSak 1-2 van megvalósítva. Nem ostream& -val térnek vissza !!! Kivételt várunk

7.57.2.4. ASSERT_NO_THROW [1/2]

```
#define ASSERT_NO_THROW(
    statement )
```

Érték:

```
try { gtest_lite::test.tmp = true; statement; } \
catch (...) { gtest_lite::test.tmp = false; }\
ASSERTTHROW(statement, "nem dob kivetelt.", "kivetelt dobott.")
```

Nem várunk kivételt.

7.57.2.5. ASSERT_NO_THROW [2/2]

```
#define ASSERT_NO_THROW(
    statement )
```

Érték:

```
try { gtest_lite::test.tmp = true; statement; } \
catch (...) { gtest_lite::test.tmp = false; }\
ASSERTTHROW(statement, "nem dob kivetelt.", "kivetelt dobott.")
```

Nem várunk kivételt.

7.57.2.6. 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; }
```

7.57.2.7. 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)...`

7.57.2.8. 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; \
};
```

7.57.2.9. END

```
#define END gtest_lite::test.end(); } while (false);
```

Tesztet vége.

7.57.2.10. ENDM

```
#define ENDM gtest_lite::test.end(true); } while (false);
```

Tesztet vége allokált blokkok számának összehasonlításával Ez az ellenőrzés nem bomba biztos.

7.57.2.11. 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.

7.57.2.12. 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.

7.57.2.13. 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

7.57.2.14. 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)

7.57.2.15. 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.

7.57.2.16. EXPECT_EQ

```

#define EXPECT_EQ(
    expected,
    actual ) gtest_lite::EXPECT_(expected, actual, gtest_lite::eq, __FILE__, __LINE↵
__, "EXPECT_EQ(" #expected ", " #actual ")") )

```

Azonosságot elváró makró

7.57.2.17. EXPECT_FALSE

```

#define EXPECT_FALSE(
    actual ) gtest_lite::EXPECT_(false, actual, gtest_lite::eq, __FILE__, __LINE__,
"EXPECT_FALSE(" #actual ")") )

```

Hamis értéket elváró makró

7.57.2.18. 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

7.57.2.19. 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

7.57.2.20. 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

7.57.2.21. 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

7.57.2.22. EXPECT_LT

```
#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

7.57.2.23. 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

7.57.2.24. 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 kivételt.", "kivetelt dobott.")
```

Nem várunk kivételt.

7.57.2.25. 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)

7.57.2.26. EXPECT_STRCASENE

```
#define EXPECT_STRCASENE(  
    expected,
```

```

        actual ) gtest_lite::EXPECTSTR(expected, actual, gtest_lite::nestr, __FILE__, ↵
_, __LINE__, "EXPECT_STRCASENE(" #expected ", " #actual ")", "etalon" )

```

C stringek (const char *) eltéréset tesztelő makró (kisbetű/nagybetű azonos)

7.57.2.27. 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

7.57.2.28. 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 *) eltéréset tesztelő makró

7.57.2.29. 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.

7.57.2.30. 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.

7.57.2.31. EXPECT_TRUE

```

#define EXPECT_TRUE(
    actual ) gtest_lite::EXPECT_(true, actual, gtest_lite::eq, __FILE__, __LINE__ ↵
, "EXPECT_TRUE(" #actual " )" )

```

Igaz értéket elváró makró

7.57.2.32. EXPECTTHROW

```

#define EXPECTTHROW(
    statement,
    exp,
    act )

```

Érték:

```
gtest_lite::test.expect(gtest_lite::test.tmp, __FILE__, __LINE__, #statement) \
« "*** Az utasítás " « (act) \
« "\n** Azt vártuk, hogy " « (exp) « std::endl
```

EXPECTTHROW; kivételkezelés.

Belső megvalósításhoz tartozó makrók, és osztályok.

7.57.2.33. Nem célszerű közvetlenül használni, vagy módosítani**7.57.2.34. FAIL**

```
#define FAIL( ) gtest_lite::test.expect(false, __FILE__, __LINE__, "FAIL()", true)
```

Sikertelen teszt fatális hiba makrója.

7.57.2.35. GTEND

```
#define GTEND(
    os )
```

7.57.2.36. GTINIT

```
#define GTINIT(
    IS )
```

7.57.2.37. SUCCEED

```
#define SUCCEED( ) gtest_lite::test.expect(true, __FILE__, __LINE__, "SUCCEED()", true)
```

Sikeres teszt makrója.

7.57.2.38. 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>	- teszt eset neve (csak a gtest kompatibilitás miatt van külön neve az eseteknek)
<i>N</i>	- teszt neve

7.57.3. Függvények dokumentációja**7.57.3.1. hasMember()**

```
void hasMember (
    ... ) [inline]
```

Segédfüggvény egy publikus adattag, vagy tagfüggvény létezésének tesztelésére fordítási időben

7.58. src/external/memtrace.cpp fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <ctype.h>
```

7.59. src/external/memtrace.h fájlreferencia

7.60. src/fake_sfml/fake_sfml.cpp fájlreferencia

```
#include "fake_sfml.hpp"
```

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.hpp fájlreferencia

```
#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include "../Utils.hpp"
```

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.hpp"
```

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.hpp fájlreferencia

A Szimuláció konfigurációja itt érhető el.

```
#include "YAMLParse.hpp"  
#include <mutex>  
#include <iostream>  
#include <string>
```

Osztályok

- class [GameConfig](#)
 A világ szimulációjának leírása.

Enumerációk

- enum class `Language` { `MAGYAR` , `ENGLISH` , `NONE` }

7.65.1. Részletes leírás

A Szimuláció konfigurációja itt érhető el.

Szerző

Funk Gábor

Dátum

2025-04-21

7.65.2. Enumerációk dokumentációja

7.65.2.1. Language

```
enum Language [strong]
```

Enumeráció-értékek

MAGYAR	
ENGLISH	
NONE	

7.66. src/GameManager.cpp fájlreferencia

```
#include "GameManager.hpp"
```

7.67. src/GameManager.d fájlreferencia

7.68. src/GameManager.hpp fájlreferencia

A játékmenedzser osztály itt van deklarálva.

```
#include "Utils.hpp"  
#include <SFML/Graphics.hpp>  
#include <iostream>  
#include <vector>  
#include "GameConfig.hpp"  
#include "ui/button.hpp"  
#include "World.hpp"  
#include "EntityPlacer.hpp"  
#include "PostProcessor.hpp"  
#include "MusicPlayer.hpp"  
#include "SaveManager.hpp"  
#include "SoundPlayer.hpp"
```

Osztályok

- class `GameManager`

A világ szimulálásáért és a kirazolás irányításáért felelős osztály.

7.68.1. Részletes leírás

A játékmenedzser osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.69. src/HumanResources.cpp fájlreferencia

```
#include "HumanResources.hpp"
```

7.70. src/HumanResources.d fájlreferencia

7.71. src/HumanResources.hpp 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.hpp"
#include "GameConfig.hpp"
#include "World.hpp"
#include "TextureManager.hpp"
#include "Random_Gen.hpp"
#include "external/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.hpp"
```

7.75. src/MusicPlayer.d fájlreferencia

7.76. src/MusicPlayer.hpp fájlreferencia

A zene lejátszó osztály itt van deklarálva.

```
#include "Utils.hpp"  
#include <SFML/Audio.hpp>  
#include <unordered_map>  
#include <string>  
#include <memory>  
#include "../exceptions/MusicLoadException.hpp"
```

Osztályok

- class [MusicPlayer](#)

A zene játsszó osztály leírása.

7.76.1. Részletes leírás

A zene lejátszó osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.77. src/PostProcessor.cpp fájlreferencia

```
#include "PostProcessor.hpp"
```

7.78. src/PostProcessor.d fájlreferencia

7.79. src/PostProcessor.hpp fájlreferencia

A grafikus szépítő osztály deklarációját tartalmazza.

```
#include "Utils.hpp"  
#include <SFML/Graphics.hpp>  
#include <string>  
#include "TextureManager.hpp"
```

Osztályok

- class [PostProcessor](#)

A grafikus szépítő osztály leírása.

7.79.1. Részletes leírás

A grafikus szépművész osztály deklarációját tartalmazza.

Szerző

Funk Gábor

Dátum

2025-04-21

7.80. src/Profession.cpp fájlreferencia

```
#include "Profession.hpp"
```

7.81. src/Profession.d fájlreferencia

7.82. src/Profession.hpp fájlreferencia

Ebben a fájlban van deklarálni a szakma indikátor osztály.

```
#include "Textureable.hpp"  
#include "TextureManager.hpp"
```

Osztályok

- class [Profession](#)

A szakma osztály leírása.

7.82.1. Részletes leírás

Ebben a fájlban van deklarálni a szakma indikátor osztály.

Szerző

Funk Gábor

Dátum

2025-04-21

7.83. src/Random_Gen.cpp fájlreferencia

```
#include "Random_Gen.hpp"
```

7.84. src/Random_Gen.d fájlreferencia

7.85. src/Random_Gen.hpp fájlreferencia

A véletlen generátor osztályt tároló fájl.

```
#include <iostream>  
#include <chrono>  
#include <random>  
#include <mutex>
```

Osztályok

- class [RandomGenerator](#)

Egy korszerűbb és konfigurálhatóbb véletlen szám generátor osztály.

7.85.1. Részletes leírás

A véletlen generátor osztályt tároló fájl.

Szerző

Funk Gábor

Dátum

2025-04-21

7.86. src/SaveHelpers.cpp fájlreferencia

```
#include "SaveHelpers.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

Függvények

- std::string [minerals::mineral_to_string](#) (MINERAL_TYPE type)

Mentést elősegítő függvények.

7.87. src/SaveHelpers.d fájlreferencia

7.88. src/SaveHelpers.hpp fájlreferencia

A mentést segítő factory-k és segédfüggvények.

```
#include <string>
#include <unordered_map>
#include <functional>
#include "creatures/Living.hpp"
#include "creatures/humans/Human.hpp"
#include "creatures/humans/Woodcutter.hpp"
#include "creatures/humans/Farmer.hpp"
#include "creatures/humans/Stonemason.hpp"
#include "creatures/humans/Fisherman.hpp"
#include "creatures/humans/Builder.hpp"
#include "creatures/humans/King.hpp"
#include "creatures/humans/AnglerMiner.hpp"
#include "creatures/humans/Soldier.hpp"
#include "creatures/Goat.hpp"
#include "creatures/hostiles/Crocodile.hpp"
#include "creatures/hostiles/Bear.hpp"
#include "creatures/hostiles/KillerRobot.hpp"
#include "world_object/Structure.hpp"
#include "world_object/ResourceStructure.hpp"
#include "world_object/BerryBush.hpp"
```

```
#include "world_object/Stone.hpp"
#include "world_object/Tree.hpp"
#include "world_object/Iron.hpp"
#include "world_object/CityCenter.hpp"
#include "world_object/House.hpp"
```

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.

7.88.1. Részletes leírás

A mentést segítő factory-k és segédfüggvények.

Szerző

Funk Gábor

Dátum

2025-04-21

7.89. src/SaveManager.cpp fájlreferencia

```
#include "SaveManager.hpp"
```

7.90. src/SaveManager.d fájlreferencia

7.91. src/SaveManager.hpp fájlreferencia

A fájl menedzseléshez szolgáló osztály itt van deklarálva.

```
#include <iostream>
#include <string>
#include <fstream>
#include <cstdio>
#include "Utils.hpp"
#include "World.hpp"
#include "SaveHelpers.hpp"
#include "GameConfig.hpp"
#include "../exceptions/FileExceptions.hpp"
```

Osztályok

- class [SaveManager](#)

A fájl menedzseléshez szolgáló osztály leírása.

7.91.1. Részletes leírás

A fájl menedzseléshez szolgáló osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.92. src/Shadowable.cpp fájlreferencia

```
#include "Shadowable.hpp"
```

7.93. src/Shadowable.d fájlreferencia

7.94. src/Shadowable.hpp fájlreferencia

Az árnyék szimulálásához való osztályt tartalmazza.

```
#include "TextureManager.hpp"  
#include "GameConfig.hpp"  
#include "Utils.hpp"  
#include <SFML/Graphics.hpp>  
#include <string>  
#include <cmath>
```

Osztályok

- class [Shadowable](#)

Az árnyékoláshoz szükséges interface.

7.94.1. Részletes leírás

Az árnyék szimulálásához való osztályt tartalmazza.

Szerző

Funk Gábor

Dátum

2025-04-21

7.95. src/SoundPlayer.cpp fájlreferencia

```
#include "SoundPlayer.hpp"
```

7.96. src/SoundPlayer.d fájlreferencia

7.97. src/SoundPlayer.hpp fájlreferencia

Ebben a fájlba vannak a hanglejátszó osztályhoz szükséges deklarációk.

```
#include "Utils.hpp"
#include <SFML/Audio.hpp>
#include <unordered_map>
#include <string>
#include <memory>
#include <iostream>
```

Osztályok

- class [SoundPlayer](#)
A hanglejátszó osztály leírása.

7.97.1. Részletes leírás

Ebben a fájlba vannak a hanglejátszó osztályhoz szükséges deklarációk.

Szerző

Funk Gábor

Dátum

2025-04-21

7.98. src/terrain_tiles/Tile.cpp fájlreferencia

```
#include "Tile.hpp"
```

Névterek

- [tiles](#)
Az összes terepkocka elem ebben a névtérben van.

7.99. src/terrain_tiles/Tile.d fájlreferencia

7.100. src/terrain_tiles/Tile.hpp fájlreferencia

A Terepkocka osztály itt van deklarálv.

```
#include "../Textureable.hpp"
#include "../TextureManager.hpp"
```

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` }

7.100.1. Részletes leírás

A Terepkocka osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.101. src/TerrainContainer.hpp fájlreferencia

A Világ terepének a deklarálása ebben a fájlba van.

```
#include "Utils.hpp"
#include <SFML/Graphics.hpp>
#include "GameManager.hpp"
#include "GameConfig.hpp"
#include "Random_Gen.hpp"
#include "terrain_tiles/Tile.hpp"
#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.hpp fájlreferencia

Ebbe a fájlba van a textúrázáshoz szükséges osztály.

```
#include "Utils.hpp"
#include <SFML/Graphics.hpp>
#include <string>
```

Osztályok

- class [Textureable](#)

Egy interface, ami a textúrázáshoz kell.

7.103.1. Részletes leírás

Ebbe a fájlba van a textúrázáshoz szükséges osztály.

Szerző

Funk Gábor

Dátum

2025-04-21

7.104. src/TextureManager.cpp fájlreferencia

```
#include "TextureManager.hpp"
```

7.105. src/TextureManager.d fájlreferencia

7.106. src/TextureManager.hpp fájlreferencia

Ebbe a fájlba van az az osztály, ami a textúrák betöltéséért, kiosztásáért és tárolásáért felelős.

```
#include "Utils.hpp"
#include <SFML/Graphics.hpp>
#include <unordered_map>
#include <string>
#include <memory>
#include <iostream>
```

Osztályok

- class [TextureManager](#)

A Textúra kezelő osztály.

7.106.1. Részletes leírás

Ebbe a fájlba van az az osztály, ami a textúrák betöltéséért, kiosztásáért és tárolásáért felelős.

A [TextureManager](#) osztály singleton pattern-t használ.

Szerző

Funk Gábor

Dátum

2025-04-21

7.107. src/ui/button.cpp fájlreferencia

```
#include "button.hpp"
```

Névterek

- [ui](#)

Az összes UI elem ebben a névtérben van.

7.108. src/ui/button.d fájlreferencia

7.109. src/ui/button.hpp fájlreferencia

A gomb osztály itt van deklarálva.

```
#include "../Textureable.hpp"
#include "../TextureManager.hpp"
#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.

7.109.1. Részletes leírás

A gomb osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.110. src/Utils.cpp fájlreferencia

```
#include "Utils.hpp"
#include "GameConfig.hpp"
```

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 nyelvre 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 nyelvre is lehetőség és arra is, hogy ne írjon semmit.

7.111. src/Utils.d fájlreferencia

7.112. src/Utils.hpp fájlreferencia

Ebben a fájlba vannak a segéd függvények.

```
#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 nyelvű 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 nyelvű is lehetőség és arra is, hogy ne írjon semmit.

7.112.1. Részletes leírás

Ebben a fájlba vannak a segéd függvények.

Szerző

Funk Gábor

Dátum

2025-04-21

7.112.2. Makródefiníciók dokumentációja

7.112.2.1. WITH_SFML_RENDER

```
#define WITH_SFML_RENDER
```

Ez arra kell, ha nem headless mode kell a programból.

7.112.3. Függvények dokumentációja

7.112.3.1. 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.

7.112.3.2. 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.

7.112.3.3. 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"
```

Függvények

- std::ostream & [operator<<](#) (std::ostream &os, [World](#) &w)
- std::ifstream & [operator>>](#) (std::ifstream &in, [World](#) &w)

7.113.1. Függvények dokumentációja**7.113.1.1. operator<<()**

```
std::ostream& operator<< (
    std::ostream & os,
    World & w )
```

7.113.1.2. operator>>()

```
std::ifstream& operator>> (
    std::ifstream & in,
    World & w )
```

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.hpp"
#include <fstream>
#include <string>
#include <sstream>
#include "Utils.hpp"
#include "TerrainContainer.hpp"
#include "creatures/Living.hpp"
#include "creatures/humans/Human.hpp"
#include "creatures/humans/Woodcutter.hpp"
#include "creatures/humans/Farmer.hpp"
#include "creatures/humans/Stonemason.hpp"
#include "creatures/humans/Fisherman.hpp"
#include "creatures/humans/Builder.hpp"
#include "creatures/humans/King.hpp"
#include "creatures/humans/AnglerMiner.hpp"
#include "creatures/humans/Soldier.hpp"
#include "creatures/Goat.hpp"
#include "creatures/hostiles/Crocodile.hpp"
#include "creatures/hostiles/Bear.hpp"
#include "creatures/hostiles/KillerRobot.hpp"
#include <unordered_map>
#include <queue>
#include "world_object/Structure.hpp"
#include "world_object/ResourceStructure.hpp"
#include "world_object/BerryBush.hpp"
#include "world_object/Stone.hpp"
#include "world_object/Tree.hpp"
#include "world_object/Iron.hpp"
#include "world_object/CityCenter.hpp"
#include "world_object/House.hpp"
#include <vector>
#include "SoundPlayer.hpp"
#include "SaveHelpers.hpp"
#include "../exceptions/FileExceptions.hpp"
#include "../exceptions/WorldExceptions.hpp"
#include "HumanResources.hpp"
```

Osztályok

- class [WorldBase](#)

A világ elemeinek nyilvántartása és a kiszolgáló függvények is itt vannak. Csak az alap függvényeket tartalmazza.

- 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.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.117. src/world_object/BerryBush.d fájlreferencia

7.118. src/world_object/BerryBush.hpp fájlreferencia

A bokor osztály itt van deklarálva.

```
#include "ResourceStructure.hpp"
```

Osztályok

- class [minerals::BerryBush](#)

A bokor osztály leírása. Ételt ad, ha kitermelik.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.118.1. Részletes leírás

A bokor osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.119. src/world_object/CityCenter.cpp fájlreferencia

```
#include "CityCenter.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.120. src/world_object/CityCenter.d fájlreferencia

7.121. src/world_object/CityCenter.hpp fájlreferencia

A városközpont osztály itt van deklarálva.

```
#include "Structure.hpp"  
#include <string>
```

Osztályok

- class [minerals::CityCenter](#)

A városközpont osztály leírása. E köré épülnek a házak.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.121.1. Részletes leírás

A városközpont osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.122. src/world_object/House.cpp fájlreferencia

```
#include "House.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.123. src/world_object/House.d fájlreferencia

7.124. src/world_object/House.hpp fájlreferencia

A Ház osztály itt van deklarálva.

```
#include "Structure.hpp"  
#include <string>  
#include "../Random_Gen.hpp"
```

Osztályok

- class [minerals::House](#)

A ház osztály leírása. Szinttől függően idéz embereket.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.124.1. Részletes leírás

A Ház osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.125. src/world_object/Iron.cpp fájlreferencia

```
#include "Iron.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.126. src/world_object/Iron.d fájlreferencia

7.127. src/world_object/Iron.hpp fájlreferencia

A Vasérc osztály itt van deklarálva.

```
#include "ResourceStructure.hpp"  
#include "../Random_Gen.hpp"
```

Osztályok

- class [minerals::Iron](#)

A vasérc osztály leírása. Vasat ad, amikor kitermelik.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.127.1. Részletes leírás

A Vasérc osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.128. src/world_object/ResourceStructure.cpp fájlreferencia

```
#include "ResourceStructure.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.129. src/world_object/ResourceStructure.d fájlreferencia

7.130. src/world_object/ResourceStructure.hpp fájlreferencia

A kibányászható osztály itt van deklarálva.

```
#include "Structure.hpp"  
#include <string>
```

Osztályok

- class [minerals::ResourceStructure](#)

Az erőforrás struktúra osztály leírása.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.130.1. Részletes leírás

A kibányászható osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.131. src/world_object/Stone.cpp fájlreferencia

```
#include "Stone.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.132. src/world_object/Stone.d fájlreferencia

7.133. src/world_object/Stone.hpp fájlreferencia

A Kő osztály itt van deklarálva.

```
#include "ResourceStructure.hpp"
```

```
#include "../Random_Gen.hpp"
```

Osztályok

- class [minerals::Stone](#)

A kő osztály leírása. követ ad, amikor kitermelik.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.133.1. Részletes leírás

A Kő osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.134. src/world_object/Structure.cpp fájlreferencia

```
#include "Structure.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.135. src/world_object/Structure.d fájlreferencia

7.136. src/world_object/Structure.hpp fájlreferencia

A struktúra osztály itt van deklarálva.

```
#include "../Textureable.hpp"
```

```
#include "../TextureManager.hpp"
```

```
#include "../GameConfig.hpp"
```

```
#include "../Shadowable.hpp"
```

```
#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](#) }

7.136.1. Részletes leírás

A struktúra osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.137. src/world_object/Tree.cpp fájlreferencia

```
#include "Tree.hpp"
```

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.138. src/world_object/Tree.d fájlreferencia

7.139. src/world_object/Tree.hpp fájlreferencia

A fa osztály itt van deklarálva.

```
#include "ResourceStructure.hpp"  
#include "../Random_Gen.hpp"
```

Osztályok

- class [minerals::Tree](#)

A fa osztály leírása. Fát ad, ha kitermelik.

Névterek

- [minerals](#)

Az összes struktúra ebben a névtérben van.

7.139.1. Részletes leírás

A fa osztály itt van deklarálva.

Szerző

Funk Gábor

Dátum

2025-04-21

7.140. src/WorldBase.cpp fájlreferencia

```
#include "World.hpp"
```

7.141. src/WorldBase.d fájlreferencia

7.142. src/YAMLParse.cpp fájlreferencia

```
#include "YAMLParse.hpp"  
#include "GameConfig.hpp"  
#include "Utils.hpp"
```

7.143. src/YAMLParse.d fájlreferencia

7.144. src/YAMLParse.hpp fájlreferencia

A config beolvasó osztály itt található.

```
#include <iostream>  
#include <string>  
#include <unordered_map>  
#include <algorithm>  
#include <fstream>
```

Osztályok

- class [YAMLParse](#)

Egy YAML (Yet Another Markup Language) fájl beolvasó osztály.

7.144.1. Részletes leírás

A config beolvasó osztály itt található.

Szerző

Funk Gábor

Dátum

2025-05-09

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`, 61
~Fisherman
 `creature::Fisherman`, 63
~GameManager
 `GameManager`, 71
~Goat
 `creature::Goat`, 74
~HostileInterface
 `creature::HostileInterface`, 77
~Human
 `creature::Human`, 85
~KillerRobot
 `creature::KillerRobot`, 100
~King
 `creature::King`, 103
~Living
 `creature::Living`, 106
~MusicPlayer
 `MusicPlayer`, 118
~ResourceStructure
 `minerals::ResourceStructure`, 136
~Shadowable
 `Shadowable`, 143
~Soldier
 `creature::Soldier`, 149
~Sound
 `sf::Sound`, 150
~SoundSource
 `sf::SoundSource`, 153
~Sprite
 `sf::Sprite`, 154
~Stonemason
 `creature::Stonemason`, 160
~Structure
 `minerals::Structure`, 162
~TerrainContainer
 `TerrainContainer< T >`, 168
~Test
 `gtest_lite::Test`, 174
~Texture
 `sf::Texture`, 177
~Textureable
 `Textureable`, 179
~Woodcutter
 `creature::Woodcutter`, 194
~World
 `World`, 196
~WorldBase
 `WorldBase`, 201
~ostreamRedir
 `gtest_lite::ostreamRedir`, 121

a
 `sf::Color`, 44
ablocks
 `gtest_lite::Test`, 175
ADD_FAILURE
 `gtest_lite.h`, 231
add_resources
 `HumanResources`, 88
Additive
 `sf`, 21
almostEQ
 `gtest_lite`, 16
Alpha
 `sf`, 21
AnglerMiner
 `creature::AnglerMiner`, 27
ANIMAL
 `creature`, 14
animation_speed
 `creature::LivingTexture`, 112
apply_age
 `creature::EntityBase`, 51
asSeconds
 `sf::ClockTime`, 43
ASSERT_
 `gtest_lite.h`, 231
ASSERT_EQ
 `gtest_lite.h`, 232
ASSERT_NO_THROW
 `gtest_lite.h`, 232
ASSERTTHROW
 `gtest_lite.h`, 232

astatus
 gtest_lite::Test, 174
 attack_speed
 creature::HostileInterface, 79
 attack_texture_path
 creature::LivingTexture, 112
 ATTACKING
 creature, 15

 b
 sf::Color, 45
 Bear
 creature::Bear, 29
 begin
 gtest_lite::Test, 174
 BerryBush
 minerals::BerryBush, 32
 bitsPerPixel
 sf::VideoMode, 192
 Black
 sf::Color, 45
 BlendAdd
 sf, 21, 22
 BlendMode
 sf, 21
 blendMode
 sf::RenderStates, 132
 Blue
 sf::Color, 45
 build_city_center_at
 WorldBase, 201
 Builder
 creature::Builder, 34
 Button
 ui::Button, 36

 camp_needs_spawn
 WorldBase, 206
 check_aggroed
 creature::HostileInterface, 77
 creature::Living, 106
 CITY_CENTER
 minerals, 20
 CityCenter
 minerals::CityCenter, 39
 CityCenterException, 41
 CityCenterException, 41
 clear
 sf::RenderWindow, 133
 TerrainContainer< T >, 168
 TextureManager, 181
 World, 196
 clear_at
 TerrainContainer< T >, 168
 ClockTime
 sf::ClockTime, 42, 43
 close
 sf::RenderWindow, 133
 Closed
 sf::Event, 59
 Color
 sf::Color, 44
 combine
 sf::Transform, 185
 contains
 sf::FloatRect, 65
 convertible
 _Is_Types< F, T >, 26
 create
 RoleOption, 137
 sf::RenderWindow, 133
 CREATE_Has_
 gtest_lite.h, 232
 CREATE_Has_fn_
 gtest_lite.h, 233
 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, 54
 - die, 51
 - facing, 54
 - gender, 55
 - get_gender, 52
 - get_state, 52
 - get_type, 52
 - health, 55
 - hit_timer, 55
 - inner_timer, 55
 - max_age, 55
 - posx, 55
 - posy, 56
 - run_speed_modifier, 56
 - save_name, 56
 - set_attack_texture, 52
 - set_death_texture, 53
 - set_health, 53
 - set_idle_texture, 53
 - set_run_texture, 53
 - set_state, 54
 - set_walk_texture, 54
 - speed, 56
 - state, 56
 - texture_data, 56
- creature::Farmer, 60
 - ~Farmer, 61
 - Farmer, 61
 - update_logic, 61
- creature::Fisherman, 62
 - ~Fisherman, 63
 - Fisherman, 63
 - fishing, 64
 - try_fishing, 63
 - update_logic, 64
- creature::Goat, 73
 - ~Goat, 74
 - die, 75
 - draw_logic, 75
 - get_type, 75
 - Goat, 74
 - update_logic, 75
- creature::HostileInterface, 76
 - ~HostileInterface, 77
 - attack_speed, 79
 - check_aggroed, 77
 - damage, 79
 - goal, 80
 - hostile_run, 78
 - hostile_walk, 78
 - retarget, 78
 - select_target, 78
 - set_hostile_config, 79
 - target, 80
 - try_attack, 79
- creature::Human, 83
 - ~Human, 85
 - die, 85
 - draw_logic, 85
 - get_profession_string, 85
 - get_type, 86
 - goal, 87
 - Human, 84
 - initialize, 86
 - needs_promotion, 87
 - needs_to_be_royal, 87
 - profession, 87
 - select_texture, 86
 - update_logic, 87
- creature::KillerRobot, 99
 - ~KillerRobot, 100
 - die, 100
 - draw_logic, 100
 - get_type, 101
 - KillerRobot, 100
 - select_target, 101
 - update_logic, 101
- creature::King, 102
 - ~King, 103
 - King, 103
 - update_logic, 103
- creature::Living, 104
 - ~Living, 106
 - check_aggroed, 106
 - damage, 106
 - damaged_by, 111
 - draw, 106
 - draw_logic, 107
 - get_width, 107
 - init_spritesheet_data, 107
 - look_left, 108
 - look_right, 108
 - MAX_CREATURE_SIZE, 111
 - needs_drawn, 108
 - retarget, 108
 - set_state, 109
 - setPosition, 109
 - setTexture, 109
 - setTheShadow, 109
 - shadow_logic, 110
 - update_logic, 110
 - update_spritesheet, 110
- creature::LivingTexture, 111
 - animation_speed, 112
 - attack_texture_path, 112
 - current_animation_time, 112
 - death_texture, 112
 - frame_count, 113
 - idle_texture_path, 113
 - run_texture_path, 113
 - walk_texture_path, 113
- creature::Soldier, 148
 - ~Soldier, 149

- Soldier, 148
- update_logic, 149
- creature::Stonemason, 158
 - ~Stonemason, 160
 - mining_iron, 161
 - Stonemason, 159
 - try_mine, 160
 - update_logic, 160
- creature::Woodcutter, 193
 - ~Woodcutter, 194
 - update_logic, 194
 - Woodcutter, 193
- CreatureFactory
 - SaveHelper, 138
- Crocodile
 - creature::Crocodile, 47
- current_animation_time
 - creature::LivingTexture, 112
- current_city_center
 - WorldBase, 206
- damage
 - creature::HostileInterface, 79
 - creature::Living, 106
- damaged_by
 - creature::Living, 111
- day_length
 - GameConfig, 70
- DEATH
 - creature, 15
- death_texture
 - creature::LivingTexture, 112
- death_timer
 - creature::EntityBase, 54
- deleteFile
 - SaveManager, 140
- die
 - creature::Bear, 29
 - creature::Crocodile, 47
 - creature::EntityBase, 51
 - creature::Goat, 75
 - creature::Human, 85
 - creature::KillerRobot, 100
- display
 - sf::RenderWindow, 134
- distance_to
 - Utils.cpp, 251
 - Utils.hpp, 252
- DOING_ITS_WORK
 - creature, 15
- Down
 - sf::Keyboard, 98
- draw
 - creature::Living, 106
 - minerals::Structure, 162
 - PostProcessor, 122
 - Profession, 125
 - sf::RenderWindow, 134
 - sf::Sprite, 154
 - TerrainContainer< T >, 169
 - Textureable, 179
 - tiles::Tile, 183
 - ui::Button, 37
 - World, 196
- draw_buttons
 - GameManager, 72
- draw_logic
 - creature::Bear, 29
 - creature::Crocodile, 47
 - creature::Goat, 75
 - creature::Human, 85
 - creature::KillerRobot, 100
 - creature::Living, 107
 - minerals::Structure, 163
- drawShadow
 - Shadowable, 143
- END
 - gtest_lite.h, 233
- end
 - gtest_lite::Test, 174
- ENDM
 - gtest_lite.h, 233
- ENDMsg
 - gtest_lite.h, 233
- ENGLISH
 - GameConfig.hpp, 240
- entities
 - WorldBase, 206
- ENTITY_GENDER
 - creature, 14
- ENTITY_TYPE
 - creature, 14
- EntityPlacer, 57
 - EntityPlacer, 57
 - reset_mouse, 58
 - select_entity, 58
 - setup_factory, 58
 - spacePreviouslyPressed, 58
 - toggle_placing, 58
 - try_place_entity, 58
- eq
 - gtest_lite, 16
- eqstr
 - gtest_lite, 16
- eqstrcase
 - gtest_lite, 17
- EType
 - sf::Event, 59
- Event
 - sf::Event, 59
- expect
 - gtest_lite::Test, 174
- EXPECT_
 - gtest_lite, 17
- EXPECT_ANY_THROW
 - gtest_lite.h, 233
- EXPECT_DOUBLE_EQ

- gtest_lite.h, [233](#)
- EXPECT_ENVCASEEQ
 - gtest_lite.h, [234](#)
- EXPECT_ENVEQ
 - gtest_lite.h, [234](#)
- EXPECT_EQ
 - gtest_lite.h, [234](#)
- EXPECT_FALSE
 - gtest_lite.h, [234](#)
- EXPECT_FLOAT_EQ
 - gtest_lite.h, [234](#)
- EXPECT_GE
 - gtest_lite.h, [234](#)
- EXPECT_GT
 - gtest_lite.h, [234](#)
- EXPECT_LE
 - gtest_lite.h, [235](#)
- EXPECT_LT
 - gtest_lite.h, [235](#)
- EXPECT_NE
 - gtest_lite.h, [235](#)
- EXPECT_NO_THROW
 - gtest_lite.h, [235](#)
- EXPECT_STRCASEEQ
 - gtest_lite.h, [235](#)
- EXPECT_STRCASENE
 - gtest_lite.h, [235](#)
- EXPECT_STREQ
 - gtest_lite.h, [236](#)
- EXPECT_STRNE
 - gtest_lite.h, [236](#)
- EXPECT_THROW
 - gtest_lite.h, [236](#)
- EXPECT_THROW_THROW
 - gtest_lite.h, [236](#)
- EXPECT_TRUE
 - gtest_lite.h, [236](#)
- EXPECTSTR
 - gtest_lite, [17](#)
- EXPECTTHROW
 - gtest_lite.h, [236](#)
- f
 - _Is_Types< F, T >, [25](#), [26](#)
- FACING
 - creature, [15](#)
- facing
 - creature::EntityBase, [54](#)
- FAIL
 - gtest_lite.h, [237](#)
- fail
 - gtest_lite::Test, [175](#)
- failed
 - gtest_lite::Test, [175](#)
- Farmer
 - creature::Farmer, [61](#)
- FEMALE
 - creature, [14](#)
- file_exists_at_path
 - sf, [22](#)
- Fisherman
 - creature::Fisherman, [63](#)
- fishing
 - creature::Fisherman, [64](#)
- FloatRect
 - sf::FloatRect, [65](#)
- FOOD
 - minerals, [20](#)
- frame_count
 - creature::LivingTexture, [113](#)
- g
 - sf::Color, [45](#)
- game_loop
 - GameManager, [72](#)
- GameConfig, [66](#)
 - day_length, [70](#)
 - GameConfig, [67](#)
 - get_config_level, [68](#)
 - get_hostiles_count, [68](#)
 - get_instance, [68](#)
 - get_lang, [68](#)
 - get_max_spawn_tries, [68](#)
 - get_resource_scarcity, [68](#)
 - get_screen_height, [69](#)
 - get_screen_width, [69](#)
 - get_sfml_lang, [69](#)
 - get_target_fps, [69](#)
 - get_world_size, [69](#)
 - operator=, [69](#)
 - read_from_config_file, [70](#)
 - set_config_level, [70](#)
 - set_world_size, [70](#)
- GameConfig.cpp
 - trim, [239](#)
- GameConfig.hpp
 - ENGLISH, [240](#)
 - Language, [240](#)
 - MAGYAR, [240](#)
 - NONE, [240](#)
- GameManager, [71](#)
 - ~GameManager, [71](#)
 - draw_buttons, [72](#)
 - game_loop, [72](#)
 - GameManager, [71](#)
 - get_elapsed_time, [72](#)
 - handle_unit_placement, [72](#)
 - is_valid, [72](#)
 - run, [72](#)
 - setup_buttons, [73](#)
 - simulate_tick, [73](#)
 - update_buttons, [73](#)
- ge
 - gtest_lite, [18](#)
- gender
 - creature::EntityBase, [55](#)
- generate_world
 - TerrainContainer< T >, [169](#)

get_border_height
 World, 197
 get_border_width
 World, 197
 get_config_level
 GameConfig, 68
 get_count_from
 HumanResources, 89
 get_current_city_center
 WorldBase, 201
 get_elapsed_time
 GameManager, 72
 get_excluded_entities
 WorldBase, 202
 get_gender
 creature::EntityBase, 52
 get_harvested
 minerals::ResourceStructure, 136
 get_height
 TerrainContainer< T >, 169
 get_height_offset
 Shadowable, 144
 get_hostiles_count
 GameConfig, 68
 get_instance
 GameConfig, 68
 RandomGenerator, 128
 get_lang
 GameConfig, 68
 get_max_spawn_tries
 GameConfig, 68
 get_position_nearby_town
 WorldBase, 202
 get_profession_string
 creature::Human, 85
 get_random_house_pos
 WorldBase, 202
 get_random_int
 RandomGenerator, 128
 get_random_suitable_position
 WorldBase, 202
 get_resource_scarcity
 GameConfig, 68
 get_resources
 WorldBase, 203
 get_screen_height
 GameConfig, 69
 get_screen_width
 GameConfig, 69
 get_seed
 TerrainContainer< T >, 169
 get_settlement_age
 minerals::CityCenter, 40
 get_sfml_lang
 GameConfig, 69
 get_shadow_strength
 Shadowable, 144
 get_skew_offset
 Shadowable, 144
 get_state
 creature::EntityBase, 52
 get_structure_type
 WorldBase, 203
 get_target_fps
 GameConfig, 69
 get_type
 creature::Bear, 30
 creature::Crocodile, 48
 creature::EntityBase, 52
 creature::Goat, 75
 creature::Human, 86
 creature::KillerRobot, 101
 minerals::BerryBush, 32
 minerals::CityCenter, 40
 minerals::House, 81
 minerals::Iron, 97
 minerals::Stone, 158
 minerals::Structure, 163
 minerals::Tree, 187
 tiles::Tile, 183
 get_value_of_key
 YAMLParse, 208
 get_width
 creature::Living, 107
 TerrainContainer< T >, 170
 get_world_size
 GameConfig, 69
 getCreatureFactory
 SaveHelper, 139
 getDesktopMode
 sf::VideoMode, 192
 getElapsedTime
 sf::Clock, 42
 getGlobalBounds
 sf::Sprite, 154, 155
 getHumanFactory
 SaveHelper, 139
 getInstance
 TextureManager, 181
 getLocalBounds
 sf::Sprite, 155
 getPosition
 sf::Mouse, 114
 sf::Sprite, 155
 getResourceFactory
 SaveHelper, 139
 getSize
 sf::Texture, 177
 getStatus
 sf::Music, 115
 getTest
 gtest_lite::Test, 175
 getTexture
 sf::Sprite, 155
 TextureManager, 181
 getTileAt

- WorldBase, [203](#)
- goal
 - creature::HostileInterface, [80](#)
 - creature::Human, [87](#)
- Goat
 - creature::Goat, [74](#)
- GRASS
 - tiles, [23](#)
- Green
 - sf::Color, [45](#)
- gt
 - gtest_lite, [18](#)
- GTEND
 - gtest_lite.h, [237](#)
- 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.h
 - ADD_FAILURE, [231](#)
 - ASSERT_, [231](#)
 - ASSERT_EQ, [232](#)
 - ASSERT_NO_THROW, [232](#)
 - ASSERTTHROW, [232](#)
 - CREATE_Has_, [232](#)
 - CREATE_Has_fn_, [233](#)
 - END, [233](#)
 - ENDM, [233](#)
 - ENDMsg, [233](#)
 - EXPECT_ANY_THROW, [233](#)
 - EXPECT_DOUBLE_EQ, [233](#)
 - EXPECT_ENVCASEEQ, [234](#)
 - EXPECT_ENVEQ, [234](#)
 - EXPECT_EQ, [234](#)
 - EXPECT_FALSE, [234](#)
 - EXPECT_FLOAT_EQ, [234](#)
 - EXPECT_GE, [234](#)
 - EXPECT_GT, [234](#)
 - EXPECT_LE, [235](#)
 - EXPECT_LT, [235](#)
 - EXPECT_NE, [235](#)
 - EXPECT_NO_THROW, [235](#)
 - EXPECT_STRCASEEQ, [235](#)
 - EXPECT_STRCASENE, [235](#)
 - EXPECT_STREQ, [236](#)
 - EXPECT_STRNE, [236](#)
 - EXPECT_THROW, [236](#)
 - EXPECT_THROW_THROW, [236](#)
 - EXPECT_TRUE, [236](#)
 - EXPECTTHROW, [236](#)
 - FAIL, [237](#)
 - GTEND, [237](#)
 - GTINIT, [237](#)
 - hasMember, [237](#)
 - SUCCEED, [237](#)
 - TEST, [237](#)
- gtest_lite::ostreamRedir, [120](#)
 - ~ostreamRedir, [121](#)
 - ostreamRedir, [120](#)
- gtest_lite::Test, [173](#)
 - ~Test, [174](#)
 - ablocks, [175](#)
 - astatus, [174](#)
 - begin, [174](#)
 - end, [174](#)
 - expect, [174](#)
 - fail, [175](#)
 - failed, [175](#)
 - getTest, [175](#)
 - name, [175](#)
 - null, [176](#)
 - os, [176](#)
 - status, [176](#)
 - sum, [176](#)
 - tmp, [176](#)
- GTINIT
 - gtest_lite.h, [237](#)
- handle_unit_placement
 - GameManager, [72](#)
- harvest
 - minerals::BerryBush, [32](#)
 - minerals::Iron, [97](#)
 - minerals::ResourceStructure, [136](#)
 - minerals::Stone, [158](#)
 - minerals::Tree, [187](#)
- harvested
 - minerals::ResourceStructure, [137](#)
- hasMember
 - gtest_lite.h, [237](#)
- health
 - creature::EntityBase, [55](#)
- height
 - sf::Bound, [33](#)
 - sf::FloatRect, [65](#)
 - sf::IntRect, [94](#)
 - sf::VideoMode, [192](#)
- height_offset
 - Shadowable, [146](#)
- hit_timer
 - creature::EntityBase, [55](#)
- hostile_run
 - creature::HostileInterface, [78](#)
- hostile_walk
 - creature::HostileInterface, [78](#)
- House
 - minerals::House, [81](#)
- houses

- WorldBase, 206
- HOUSING
 - minerals, 20
- HUMAN
 - creature, 14
- Human
 - creature::Human, 84
- HumanFactory
 - SaveHelper, 138
- HumanResources, 88
 - add_resources, 88
 - get_count_from, 89
 - is_there_enough_resource, 89
 - remove_resources, 89
 - set_resources, 90
- humans
 - WorldBase, 206
- IDLE
 - creature, 15
- idle_texture_path
 - creature::LivingTexture, 113
- ImportInvalidEntityException, 90
 - ImportInvalidEntityException, 91
- ImportInvalidHousingLevelException, 91
 - ImportInvalidHousingLevelException, 91
- ImportInvalidHumanProfessionException, 92
 - ImportInvalidHumanProfessionException, 92
- ImportInvalidResourceException, 93
 - ImportInvalidResourceException, 93
- increment
 - sf::ClockTime, 43
- init
 - tiles::Tile, 183
- init_spritesheet_data
 - creature::Living, 107
- initialize
 - creature::Human, 86
- inner_timer
 - creature::EntityBase, 55
 - minerals::ResourceStructure, 137
- IntRect
 - sf::IntRect, 94
- Invalid
 - sf::Event, 59
- InvalidBorderSizeException, 95
 - InvalidBorderSizeException, 95
- IRON
 - minerals, 20
- Iron
 - minerals::Iron, 96
- iron_req
 - minerals::House, 82
- is_on_screen
 - TerrainContainer< T >, 170
- is_there_enough_resource
 - HumanResources, 89
- is_there_room_for_housing
 - minerals::CityCenter, 40
- is_valid
 - GameManager, 72
- is_valid_coordinate
 - TerrainContainer< T >, 170
- isButtonPressed
 - sf::Mouse, 114
- isKeyPressed
 - sf::Keyboard, 98
- isOpen
 - sf::RenderWindow, 134
- isValid
 - sf::VideoMode, 192
- Key
 - sf::Keyboard, 98
- KillerRobot
 - creature::KillerRobot, 100
- King
 - creature::King, 103
- Language
 - GameConfig.hpp, 240
- le
 - gtest_lite, 18
- LEFT
 - creature, 15
- Left
 - sf::Keyboard, 98
 - sf::Mouse, 114
- left
 - sf::FloatRect, 66
 - sf::IntRect, 94
- level
 - minerals::House, 82
- LIVINGSTATE
 - creature, 15
- load_music
 - MusicPlayer, 118
- load_profession
 - Profession, 126
- load_sound
 - SoundPlayer, 151
- loadFile
 - SaveManager, 140
- loadFromFile
 - sf::SoundBuffer, 151
 - sf::Texture, 177
- loadTexture
 - TextureManager, 181
- log_text
 - Utils.cpp, 251
 - Utils.hpp, 253
- look_left
 - creature::Living, 108
- look_right
 - creature::Living, 108
- It
 - gtest_lite, 18

MAGYAR
 GameConfig.hpp, 240
 main
 main.cpp, 241
 main.cpp
 main, 241
 MALE
 creature, 14
 matrix
 sf::Transform, 186
 max_age
 creature::EntityBase, 55
 MAX_CREATURE_SIZE
 creature::Living, 111
 MAX_OBJECT_SIZE
 minerals::Structure, 165
 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
 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, 80
 get_type, 81
 House, 81
 iron_req, 82
 level, 82
 stone_req, 82
 update_logic, 81
 wood_req, 82
 minerals::Iron, 96
 get_type, 97
 harvest, 97
 Iron, 96
 update_logic, 97
 minerals::ResourceStructure, 135
 ~ResourceStructure, 136
 get_harvested, 136
 harvest, 136
 harvested, 137
 inner_timer, 137
 ResourceStructure, 136
 minerals::Stone, 157
 get_type, 158
 harvest, 158
 Stone, 157
 update_logic, 158
 minerals::Structure, 161
 ~Structure, 162
 draw, 162
 draw_logic, 163
 get_type, 163
 MAX_OBJECT_SIZE, 165
 needs_drawn, 163
 posx, 165
 posy, 165
 setPosition, 163
 setTexture, 164
 Structure, 162
 update_logic, 164
 minerals::Tree, 186
 get_type, 187
 harvest, 187
 Tree, 187
 update_logic, 188
 mining_iron
 creature::Stonemason, 161
 MOUNTAIN
 tiles, 23
 Mousedowntype
 sf::Mouse, 114
 Multiply
 sf, 21
 Music
 sf::Music, 115
 MusicLoadException, 117
 MusicLoadException, 117
 MusicPlayer, 117
 ~MusicPlayer, 118
 load_music, 118
 MusicPlayer, 118
 set_volume, 119
 toggle_music, 119
 name
 gtest_lite::Test, 175
 ne
 gtest_lite, 18
 needs_drawn
 creature::Living, 108
 minerals::Structure, 163
 needs_promotion
 creature::Human, 87
 needs_to_be_royal
 creature::Human, 87
 nestr
 gtest_lite, 19

NoEvent
 sf::Event, 59
 NONE
 GameConfig.hpp, 240
 None
 sf, 21
 null
 gtest_lite::Test, 176
 Num0
 sf::Keyboard, 98
 Num1
 sf::Keyboard, 98
 Num2
 sf::Keyboard, 98
 Num3
 sf::Keyboard, 98
 Num4
 sf::Keyboard, 98
 Num5
 sf::Keyboard, 98
 Num6
 sf::Keyboard, 98
 Num7
 sf::Keyboard, 98
 Num8
 sf::Keyboard, 98
 Num9
 sf::Keyboard, 98

 ObjectRegistry, 119
 register_type, 120
 spawn, 120
 onClick
 ui::Button, 37
 openFromFile
 sf::Music, 116
 operator<<
 World, 199
 World.cpp, 253
 operator>>
 World, 199
 World.cpp, 253
 operator=
 GameConfig, 69
 RandomGenerator, 128
 sf::Texture, 178
 operator[]
 TerrainContainer< T >, 171
 os
 gtest_lite::Test, 176
 ostreamRedir
 gtest_lite::ostreamRedir, 120

 parse_file
 YAMLPParser, 208
 Paused
 sf::SoundSource, 153
 play
 sf::Music, 116
 sf::Sound, 150
 play_sound
 SoundPlayer, 152
 Playing
 sf::SoundSource, 153
 pollEvent
 sf::RenderWindow, 134
 populate_world
 World, 197
 position
 sf::RectangleShape, 130
 PostProcessor, 121
 draw, 122
 PostProcessor, 122
 setColorOverlay, 122
 setRenderSize, 122
 setTextureFor, 123
 toggle_chromatic_aberration, 123
 toggle_noise, 123
 toggle_vignette, 124
 posix
 creature::EntityBase, 55
 minerals::Structure, 165
 posy
 creature::EntityBase, 56
 minerals::Structure, 165
 Profession, 124
 draw, 125
 load_profession, 126
 Profession, 125
 setPosition, 126
 setTexture, 126
 to_string, 127
 profession
 creature::Human, 87

 r
 sf::Color, 45
 RandomGenerator, 127
 get_instance, 128
 get_random_int, 128
 operator=, 128
 RandomGenerator, 128
 read_from_config_file
 GameConfig, 70
 ReadSaveFileFail, 129
 ReadSaveFileFail, 129
 Red
 sf::Color, 45
 regenerate
 World, 197
 register_new_house
 minerals::CityCenter, 40
 register_type
 ObjectRegistry, 120
 remove_resources
 HumanResources, 89
 remove_structure_at
 WorldBase, 204

- RenderStates
 - sf::RenderStates, 131
- RenderWindow
 - sf::RenderWindow, 133
- requirements
 - RoleOption, 138
- reset
 - sf::ClockTime, 43
- reset_mouse
 - EntityPlacer, 58
- resize
 - TerrainContainer< T >, 171
- ResourceFactory
 - SaveHelper, 139
- ResourceStructure
 - minerals::ResourceStructure, 136
- restart
 - sf::Clock, 42
- retarget
 - creature::HostileInterface, 78
 - creature::Living, 108
- RIGHT
 - creature, 15
- Right
 - sf::Keyboard, 98
 - sf::Mouse, 114
- ROBOTIC
 - creature, 14
- RoleOption, 137
 - create, 137
 - requirements, 138
- RUN
 - creature, 15
- run
 - GameManager, 72
- run_speed_modifier
 - creature::EntityBase, 56
- run_texture_path
 - creature::LivingTexture, 113
- save_name
 - creature::EntityBase, 56
- saveFile
 - SaveManager, 142
- SaveHelper, 138
 - CreatureFactory, 138
 - getCreatureFactory, 139
 - getHumanFactory, 139
 - getResourceFactory, 139
 - HumanFactory, 138
 - ResourceFactory, 139
- SaveManager, 139
 - deleteFile, 140
 - loadFile, 140
 - saveFile, 142
 - SaveManager, 140
- select_entity
 - EntityPlacer, 58
- select_target
 - creature::Bear, 30
 - creature::Crocodile, 48
 - creature::HostileInterface, 78
 - creature::KillerRobot, 101
- select_texture
 - creature::Human, 86
- set_attack_texture
 - creature::EntityBase, 52
- set_border_height
 - World, 197
- set_border_width
 - World, 198
- set_config_level
 - GameConfig, 70
- set_death_texture
 - creature::EntityBase, 53
- set_health
 - creature::EntityBase, 53
- set_height_offset
 - Shadowable, 144
- set_hostile_config
 - creature::HostileInterface, 79
- set_idle_texture
 - creature::EntityBase, 53
- set_resources
 - HumanResources, 90
- set_run_texture
 - creature::EntityBase, 53
- set_seed
 - TerrainContainer< T >, 172
- set_shadow_strength
 - Shadowable, 145
- set_skew_offset
 - Shadowable, 145
- set_state
 - creature::EntityBase, 54
 - creature::Living, 109
- set_volume
 - MusicPlayer, 119
- set_walk_texture
 - creature::EntityBase, 54
- set_world_size
 - GameConfig, 70
- setBlendMode
 - sf::RenderStates, 131
- setBuffer
 - sf::Sound, 150
- setCallback
 - ui::Button, 37
- setColor
 - sf::Sprite, 155
- setColorOverlay
 - PostProcessor, 122
- setFillColor
 - sf::RectangleShape, 130
- setFramerateLimit
 - sf::RenderWindow, 134
- setLoop

- sf::Music, 116
- setOrigin
 - sf::Sprite, 155
- setPosition
 - creature::Living, 109
 - minerals::Structure, 163
 - Profession, 126
 - sf::RectangleShape, 130
 - sf::Sprite, 155
 - Textureable, 179
 - tiles::Tile, 184
 - ui::Button, 37
- setRenderSize
 - PostProcessor, 122
- setRotation
 - sf::Sprite, 156
- setScale
 - sf::Sprite, 156
- setShadow
 - Shadowable, 145
- setShadowDayNightCycle
 - Shadowable, 145
- setShadowPosition
 - Shadowable, 146
- setShadowTexture
 - Shadowable, 146
- setSize
 - sf::RectangleShape, 130
- setTexture
 - creature::Living, 109
 - minerals::Structure, 164
 - Profession, 126
 - sf::Sprite, 156
 - Textureable, 180
 - tiles::Tile, 184
 - ui::Button, 38
- setTextureFor
 - PostProcessor, 123
- setTextureRect
 - sf::Sprite, 156
- setTheShadow
 - creature::Living, 109
- setTransform
 - sf::RenderStates, 131
- setup_buttons
 - GameManager, 73
- setup_factory
 - EntityPlacer, 58
- setVolume
 - sf::Music, 116
- 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, 59
 - Closed, 59
 - EType, 59
 - Event, 59
 - Invalid, 59
 - NoEvent, 59
 - type, 60
- sf::FloatRect, 64
 - contains, 65
 - FloatRect, 65
 - height, 65
 - left, 66
 - top, 66
 - width, 66
- sf::IntRect, 93
 - height, 94
 - IntRect, 94
 - left, 94
 - top, 94
 - width, 94
- sf::Keyboard, 97
 - Down, 98
 - isKeyPressed, 98
 - Key, 98
 - Left, 98
 - Num0, 98
 - Num1, 98
 - Num2, 98
 - Num3, 98
 - Num4, 98
 - Num5, 98
 - Num6, 98
 - Num7, 98
 - Num8, 98

- Num9, 98
- Right, 98
- simulate_key_press, 99
- simulate_key_release, 99
- Space, 98
- Up, 98
- sf::Mouse, 113
 - getPosition, 114
 - isButtonPressed, 114
 - Left, 114
 - Mousedowntype, 114
 - Right, 114
 - simulate_key_press, 114
 - simulate_key_release, 115
- sf::Music, 115
 - getStatus, 115
 - Music, 115
 - openFromFile, 116
 - play, 116
 - setLoop, 116
 - setVolume, 116
 - stop, 116
- sf::RectangleShape, 130
 - position, 130
 - setFillColor, 130
 - setPosition, 130
 - setSize, 130
- sf::RenderStates, 131
 - blendMode, 132
 - RenderStates, 131
 - setBlendMode, 131
 - setTransform, 131
 - transform, 132
- sf::RenderWindow, 132
 - clear, 133
 - close, 133
 - create, 133
 - display, 134
 - draw, 134
 - isOpen, 134
 - pollEvent, 134
 - RenderWindow, 133
 - setFramerateLimit, 134
- sf::Sound, 149
 - ~Sound, 150
 - play, 150
 - setBuffer, 150
 - stop, 150
- sf::SoundBuffer, 150
 - loadFromFile, 151
- sf::SoundSource, 152
 - ~SoundSource, 153
 - Paused, 153
 - Playing, 153
 - SoundSource, 153
 - SoundSourceType, 153
 - Stopped, 153
 - type, 153
- sf::Sprite, 154
 - ~Sprite, 154
 - draw, 154
 - getGlobalBounds, 154, 155
 - getLocalBounds, 155
 - getPosition, 155
 - getTexture, 155
 - setColor, 155
 - setOrigin, 155
 - setPosition, 155
 - setRotation, 156
 - setScale, 156
 - setTexture, 156
 - setTextureRect, 156
 - Sprite, 154
- sf::Texture, 177
 - ~Texture, 177
 - getSize, 177
 - loadFromFile, 177
 - operator=, 178
 - Texture, 177
- sf::Transform, 184
 - combine, 185
 - matrix, 186
 - Transform, 185
 - transformPoint, 185
 - translate, 186
- 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, 110
- Shadowable, 142
 - ~Shadowable, 143
 - drawShadow, 143
 - get_height_offset, 144
 - get_shadow_strength, 144
 - get_skew_offset, 144
 - height_offset, 146
 - set_height_offset, 144
 - set_shadow_strength, 145
 - set_skew_offset, 145
 - setShadow, 145
 - setShadowDayNightCycle, 145
 - setShadowPosition, 146
 - setShadowTexture, 146

simulate_key_press
 sf::Keyboard, 99
 sf::Mouse, 114
 simulate_key_release
 sf::Keyboard, 99
 sf::Mouse, 115
 simulate_tick
 GameManager, 73
 SimulationException, 147
 SimulationException, 147
 Soldier
 creature::Soldier, 148
 sound_player
 WorldBase, 207
 SoundPlayer, 151
 load_sound, 151
 play_sound, 152
 stop_sound, 152
 SoundSource
 sf::SoundSource, 153
 SoundSourceType
 sf::SoundSource, 153
 Space
 sf::Keyboard, 98
 spacePreviouslyPressed
 EntityPlacer, 58
 spawn
 ObjectRegistry, 120
 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, 56
 Sprite
 sf::Sprite, 154
 src/creatures/EntityBase.cpp, 209
 src/creatures/EntityBase.d, 209
 src/creatures/EntityBase.hpp, 209
 src/creatures/EntityUtils.hpp, 210
 src/creatures/Goat.cpp, 210
 src/creatures/Goat.d, 210
 src/creatures/Goat.hpp, 210
 src/creatures/HostileInterface.cpp, 211
 src/creatures/HostileInterface.d, 211
 src/creatures/HostileInterface.hpp, 211
 src/creatures/hostiles/Bear.cpp, 212
 src/creatures/hostiles/Bear.d, 212
 src/creatures/hostiles/Bear.hpp, 212
 src/creatures/hostiles/Crocodile.cpp, 213
 src/creatures/hostiles/Crocodile.d, 213
 src/creatures/hostiles/Crocodile.hpp, 213
 src/creatures/hostiles/KillerRobot.cpp, 214
 src/creatures/hostiles/KillerRobot.d, 214
 src/creatures/hostiles/KillerRobot.hpp, 214
 src/creatures/humans/AnglerMiner.cpp, 215
 src/creatures/humans/AnglerMiner.d, 215
 src/creatures/humans/AnglerMiner.hpp, 215
 src/creatures/humans/Builder.cpp, 216
 src/creatures/humans/Builder.d, 216
 src/creatures/humans/Builder.hpp, 216
 src/creatures/humans/Farmer.cpp, 217
 src/creatures/humans/Farmer.d, 217
 src/creatures/humans/Farmer.hpp, 217
 src/creatures/humans/Fisherman.cpp, 218
 src/creatures/humans/Fisherman.d, 218
 src/creatures/humans/Fisherman.hpp, 218
 src/creatures/humans/Human.cpp, 219
 src/creatures/humans/Human.d, 219
 src/creatures/humans/Human.hpp, 219
 src/creatures/humans/King.cpp, 220
 src/creatures/humans/King.d, 220
 src/creatures/humans/King.hpp, 220
 src/creatures/humans/Soldier.cpp, 221
 src/creatures/humans/Soldier.d, 221
 src/creatures/humans/Soldier.hpp, 221
 src/creatures/humans/Stonemason.cpp, 222
 src/creatures/humans/Stonemason.d, 222
 src/creatures/humans/Stonemason.hpp, 222
 src/creatures/humans/Woodcutter.cpp, 223
 src/creatures/humans/Woodcutter.d, 223
 src/creatures/humans/Woodcutter.hpp, 223
 src/creatures/Living.cpp, 224
 src/creatures/Living.d, 224
 src/creatures/Living.hpp, 224
 src/EntityPlacer.cpp, 225
 src/EntityPlacer.d, 225
 src/EntityPlacer.hpp, 225
 src/exceptions/FileExceptions.hpp, 226
 src/exceptions/MusicLoadException.hpp, 227
 src/exceptions/SimulationException.hpp, 227
 src/exceptions/WorldExceptions.hpp, 228
 src/external/gtest_lite.h, 228
 src/external/memtrace.cpp, 238
 src/external/memtrace.h, 238
 src/fake_sfml/fake_sfml.cpp, 238
 src/fake_sfml/fake_sfml.d, 238
 src/fake_sfml/fake_sfml.hpp, 238
 src/GameConfig.cpp, 239
 src/GameConfig.d, 239
 src/GameConfig.hpp, 239
 src/GameManager.cpp, 240
 src/GameManager.d, 240
 src/GameManager.hpp, 240
 src/HumanResources.cpp, 241
 src/HumanResources.d, 241
 src/HumanResources.hpp, 241
 src/main.cpp, 241
 src/main.d, 242
 src/MusicPlayer.cpp, 242

[src/MusicPlayer.d, 242](#)
[src/MusicPlayer.hpp, 242](#)
[src/PostProcessor.cpp, 242](#)
[src/PostProcessor.d, 242](#)
[src/PostProcessor.hpp, 242](#)
[src/Profession.cpp, 243](#)
[src/Profession.d, 243](#)
[src/Profession.hpp, 243](#)
[src/Random_Gen.cpp, 243](#)
[src/Random_Gen.d, 243](#)
[src/Random_Gen.hpp, 243](#)
[src/SaveHelpers.cpp, 244](#)
[src/SaveHelpers.d, 244](#)
[src/SaveHelpers.hpp, 244](#)
[src/SaveManager.cpp, 245](#)
[src/SaveManager.d, 245](#)
[src/SaveManager.hpp, 245](#)
[src/Shadowable.cpp, 246](#)
[src/Shadowable.d, 246](#)
[src/Shadowable.hpp, 246](#)
[src/SoundPlayer.cpp, 246](#)
[src/SoundPlayer.d, 247](#)
[src/SoundPlayer.hpp, 247](#)
[src/terrain_tiles/Tile.cpp, 247](#)
[src/terrain_tiles/Tile.d, 247](#)
[src/terrain_tiles/Tile.hpp, 247](#)
[src/TerrainContainer.hpp, 248](#)
[src/TerrainContainer_def.hpp, 248](#)
[src/Textureable.hpp, 248](#)
[src/TextureManager.cpp, 249](#)
[src/TextureManager.d, 249](#)
[src/TextureManager.hpp, 249](#)
[src/ui/button.cpp, 249](#)
[src/ui/button.d, 250](#)
[src/ui/button.hpp, 250](#)
[src/Utils.cpp, 250](#)
[src/Utils.d, 251](#)
[src/Utils.hpp, 251](#)
[src/World.cpp, 253](#)
[src/World.d, 254](#)
[src/World.hpp, 254](#)
[src/world_object/BerryBush.cpp, 255](#)
[src/world_object/BerryBush.d, 255](#)
[src/world_object/BerryBush.hpp, 255](#)
[src/world_object/CityCenter.cpp, 255](#)
[src/world_object/CityCenter.d, 256](#)
[src/world_object/CityCenter.hpp, 256](#)
[src/world_object/House.cpp, 256](#)
[src/world_object/House.d, 256](#)
[src/world_object/House.hpp, 256](#)
[src/world_object/Iron.cpp, 257](#)
[src/world_object/Iron.d, 257](#)
[src/world_object/Iron.hpp, 257](#)
[src/world_object/ResourceStructure.cpp, 258](#)
[src/world_object/ResourceStructure.d, 258](#)
[src/world_object/ResourceStructure.hpp, 258](#)
[src/world_object/Stone.cpp, 258](#)
[src/world_object/Stone.d, 259](#)
[src/world_object/Stone.hpp, 259](#)
[src/world_object/Structure.cpp, 259](#)
[src/world_object/Structure.d, 259](#)
[src/world_object/Structure.hpp, 259](#)
[src/world_object/Tree.cpp, 260](#)
[src/world_object/Tree.d, 260](#)
[src/world_object/Tree.hpp, 260](#)
[src/WorldBase.cpp, 261](#)
[src/WorldBase.d, 261](#)
[src/YAMLParse.cpp, 261](#)
[src/YAMLParse.d, 261](#)
[src/YAMLParse.hpp, 261](#)
 state
 [creature::EntityBase, 56](#)
 status
 [gtest_lite::Test, 176](#)
 STONE
 [minerals, 20](#)
 Stone
 [minerals::Stone, 157](#)
 stone_req
 [minerals::House, 82](#)
 Stonemason
 [creature::Stonemason, 159](#)
 stop
 [sf::Music, 116](#)
 [sf::Sound, 150](#)
 stop_sound
 [SoundPlayer, 152](#)
 Stopped
 [sf::SoundSource, 153](#)
 Structure
 [minerals::Structure, 162](#)
 StructureException, [165](#)
 [StructureException, 166](#)
 structures
 [WorldBase, 207](#)
 SUCCEED
 [gtest_lite.h, 237](#)
 sum
 [gtest_lite::Test, 176](#)
 swap_at
 [TerrainContainer< T >, 172](#)
 target
 [creature::HostileInterface, 80](#)
 terrain
 [WorldBase, 207](#)
 TerrainContainer
 [TerrainContainer< T >, 167](#)
 TerrainContainer< T >, [166](#)
 [~TerrainContainer, 168](#)
 [clear, 168](#)
 [clear_at, 168](#)
 [draw, 169](#)
 [generate_world, 169](#)
 [get_height, 169](#)
 [get_seed, 169](#)
 [get_width, 170](#)

- is_on_screen, 170
- is_valid_coordinate, 170
- operator[], 171
- resize, 171
- set_seed, 172
- swap_at, 172
- TerrainContainer, 167
- TILE_SIZE, 172
- TEST
 - gtest_lite.h, 237
- Texture
 - sf::Texture, 177
- texture_data
 - creature::EntityBase, 56
- Textureable, 178
 - ~Textureable, 179
 - draw, 179
 - setPosition, 179
 - setTexture, 180
- TextureManager, 180
 - clear, 181
 - getInstance, 181
 - getTexture, 181
 - loadTexture, 181
- TILE_SIZE
 - TerrainContainer< T >, 172
- tiles, 22
 - GRASS, 23
 - MOUNTAIN, 23
 - TILETYPE, 22
 - WATER, 23
- tiles::Tile, 182
 - draw, 183
 - get_type, 183
 - init, 183
 - setPosition, 184
 - setTexture, 184
- TILETYPE
 - tiles, 22
- tmp
 - gtest_lite::Test, 176
- to_string
 - Profession, 127
 - sf, 22
- toggle_chromatic_aberration
 - PostProcessor, 123
- toggle_music
 - MusicPlayer, 119
- toggle_noise
 - PostProcessor, 123
- toggle_placing
 - EntityPlacer, 58
- toggle_vignette
 - PostProcessor, 124
- top
 - sf::FloatRect, 66
 - sf::IntRect, 94
- Transform
 - sf::Transform, 185
- transform
 - sf::RenderStates, 132
- transformPoint
 - sf::Transform, 185
- translate
 - sf::Transform, 186
- Transparent
 - sf::Color, 45
- Tree
 - minerals::Tree, 187
- trim
 - GameConfig.cpp, 239
- try_attack
 - creature::HostileInterface, 79
- try_develop_random_role
 - World, 198
- try_fishing
 - creature::Fisherman, 63
- try_hover_animation
 - ui::Button, 38
- try_mine
 - creature::Stonemason, 160
- try_place_entity
 - EntityPlacer, 58
- type
 - sf::Event, 60
 - sf::SoundSource, 153
- ui, 23
 - ui::Button, 35
 - Button, 36
 - draw, 37
 - onClick, 37
 - setCallback, 37
 - setPosition, 37
 - setTexture, 38
 - try_hover_animation, 38
- Up
 - sf::Keyboard, 98
- update_buttons
 - GameManager, 73
- update_logic
 - creature::AnglerMiner, 27
 - creature::Bear, 30
 - creature::Builder, 35
 - creature::Crocodile, 48
 - creature::Farmer, 61
 - creature::Fisherman, 64
 - creature::Goat, 75
 - creature::Human, 87
 - creature::KillerRobot, 101
 - creature::King, 103
 - creature::Living, 110
 - creature::Soldier, 149
 - creature::Stonemason, 160
 - creature::Woodcutter, 194
 - minerals::BerryBush, 32
 - minerals::CityCenter, 40

- minerals::House, [81](#)
- minerals::Iron, [97](#)
- minerals::Stone, [158](#)
- minerals::Structure, [164](#)
- minerals::Tree, [188](#)
- update_spritesheet
 - creature::Living, [110](#)
- update_world
 - World, [199](#)
- upgrade_house_at
 - WorldBase, [205](#)
- Utils.cpp
 - distance_to, [251](#)
 - log_text, [251](#)
 - warn_text, [251](#)
- Utils.hpp
 - distance_to, [252](#)
 - log_text, [253](#)
 - warn_text, [253](#)
 - WITH_SFML_RENDER, [252](#)
- Vector2f
 - sf::Vector2f, [189](#)
- Vector2i
 - sf::Vector2i, [190](#)
- VideoMode
 - sf::VideoMode, [191](#)
- WALK
 - creature, [15](#)
- walk_texture_path
 - creature::LivingTexture, [113](#)
- warn_text
 - Utils.cpp, [251](#)
 - Utils.hpp, [253](#)
- WATER
 - tiles, [23](#)
- White
 - sf::Color, [46](#)
- width
 - sf::Bound, [33](#)
 - sf::FloatRect, [66](#)
 - sf::IntRect, [94](#)
 - sf::VideoMode, [192](#)
- WITH_SFML_RENDER
 - Utils.hpp, [252](#)
- WOOD
 - minerals, [20](#)
- wood_req
 - minerals::House, [82](#)
- Woodcutter
 - creature::Woodcutter, [193](#)
- World, [194](#)
 - ~World, [196](#)
 - clear, [196](#)
 - draw, [196](#)
 - get_border_height, [197](#)
 - get_border_width, [197](#)
 - operator<<, [199](#)
 - operator>>, [199](#)
 - 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](#)
- World.cpp
 - operator<<, [253](#)
 - operator>>, [253](#)
- WorldBase, [199](#)
 - ~WorldBase, [201](#)
 - build_city_center_at, [201](#)
 - camp_needs_spawn, [206](#)
 - current_city_center, [206](#)
 - entities, [206](#)
 - get_current_city_center, [201](#)
 - get_excluded_entities, [202](#)
 - 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, [207](#)
 - spawn_entity, [204](#)
 - spawn_structure, [204](#)
 - spawn_structure_at, [205](#)
 - structures, [207](#)
 - terrain, [207](#)
 - upgrade_house_at, [205](#)
- x
 - sf::Vector2f, [189](#)
 - sf::Vector2i, [191](#)
- y
 - sf::Vector2f, [190](#)
 - sf::Vector2i, [191](#)
- YAMLParse, [207](#)
 - get_value_of_key, [208](#)
 - parse_file, [208](#)
 - YAMLParse, [208](#)

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

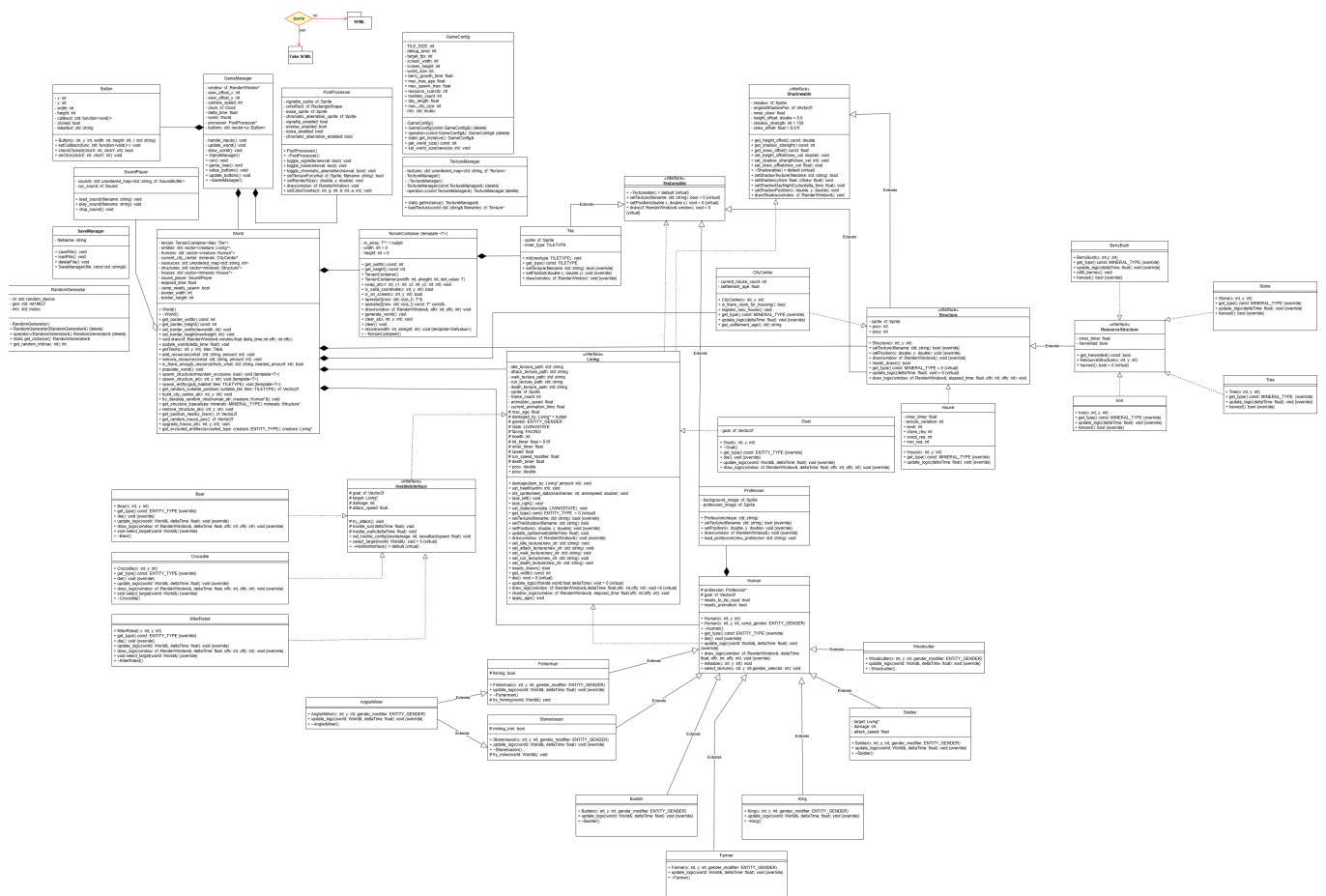
YSDDH7

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.



Entitás class

Ember class

Város class

Ház class

A játékmenedzser osztály fontosabb metódusai:

- Run(): Elindítja a világ szimulálását.
- Game_loop(): Ez a függvény felelős a képek kirajzolása közötti idő determinálásáért (delta time kiszámítása), az input kezeléséért és a világ szimulálásáért.
- Handle_inputs(): Ebben a függvényben történik a gombok kattintásának érzékelése és a jobb-bal-fel-le nyilakkal a kamera mozgatása. A kamera nem tud kimozogni a világ határain kívül, (például nem tud x:-1 y:-1 helyen lenni, mert a

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& 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&): 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.

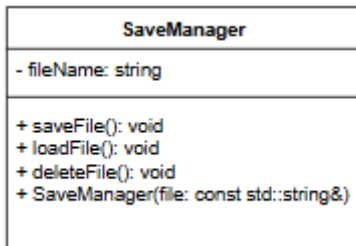
-draw(): Kirajzolja a sprite-okat vagy sprite-ot megvalósítástól függően.

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:

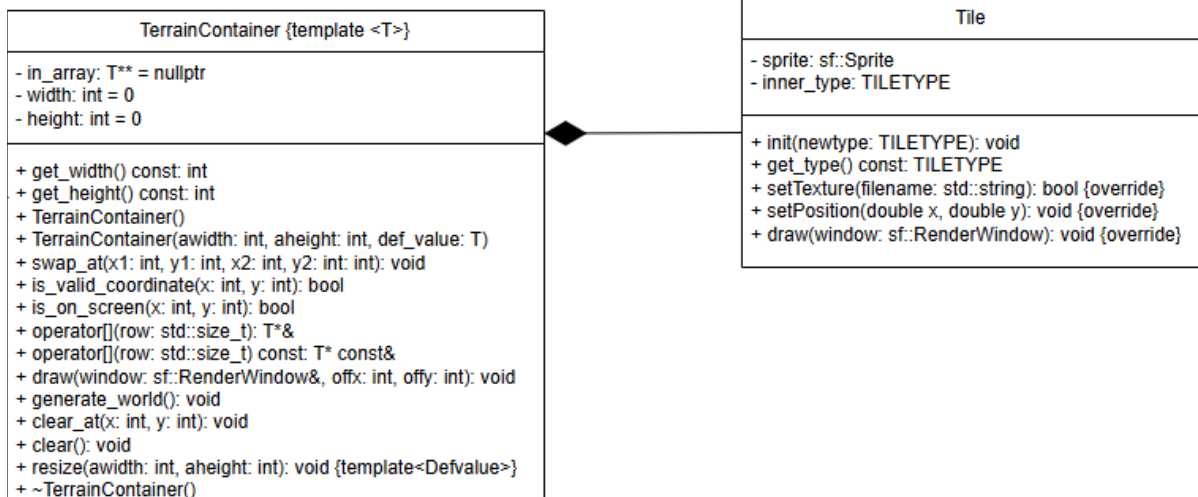
A világ egy 2 dimenziós, speciális dinamikus tömb lesz, ami képes tárolni terepkockákat.

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.

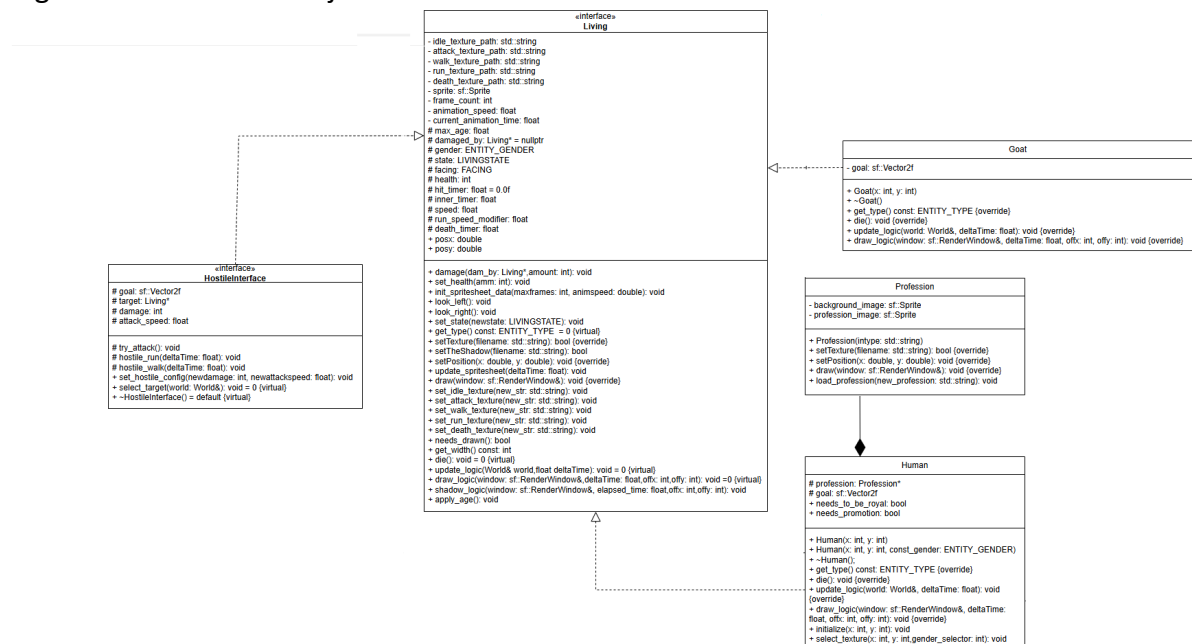
-Mező: Itt lelhető fa és étel, ezen kívül itt medvék fognak idéződni.

-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

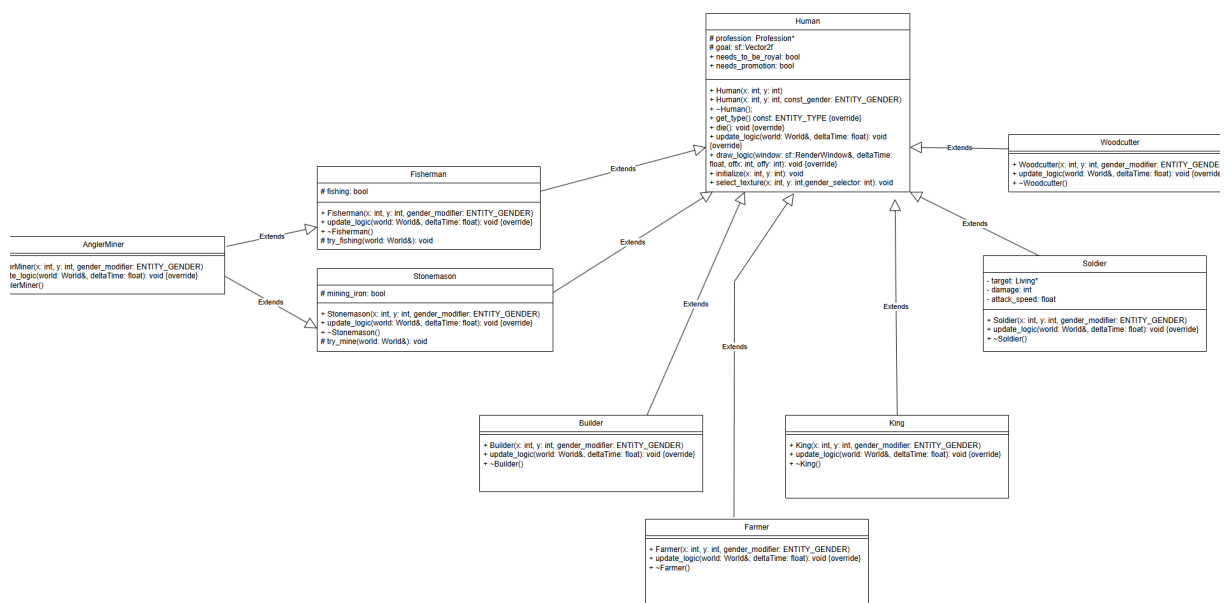
living interface megkívánja, hogy a belőle öröklődő osztályok megvalósítsák ezeket a tagfüggvényeket:

- 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)

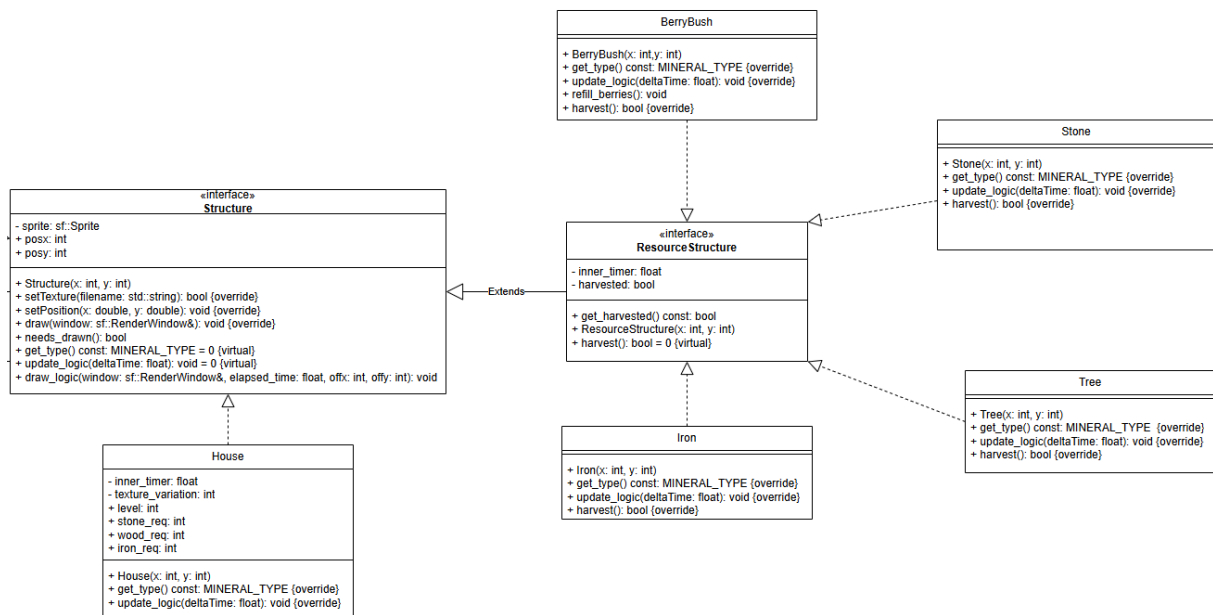
Kecske: Ártalmatlan állat.

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.

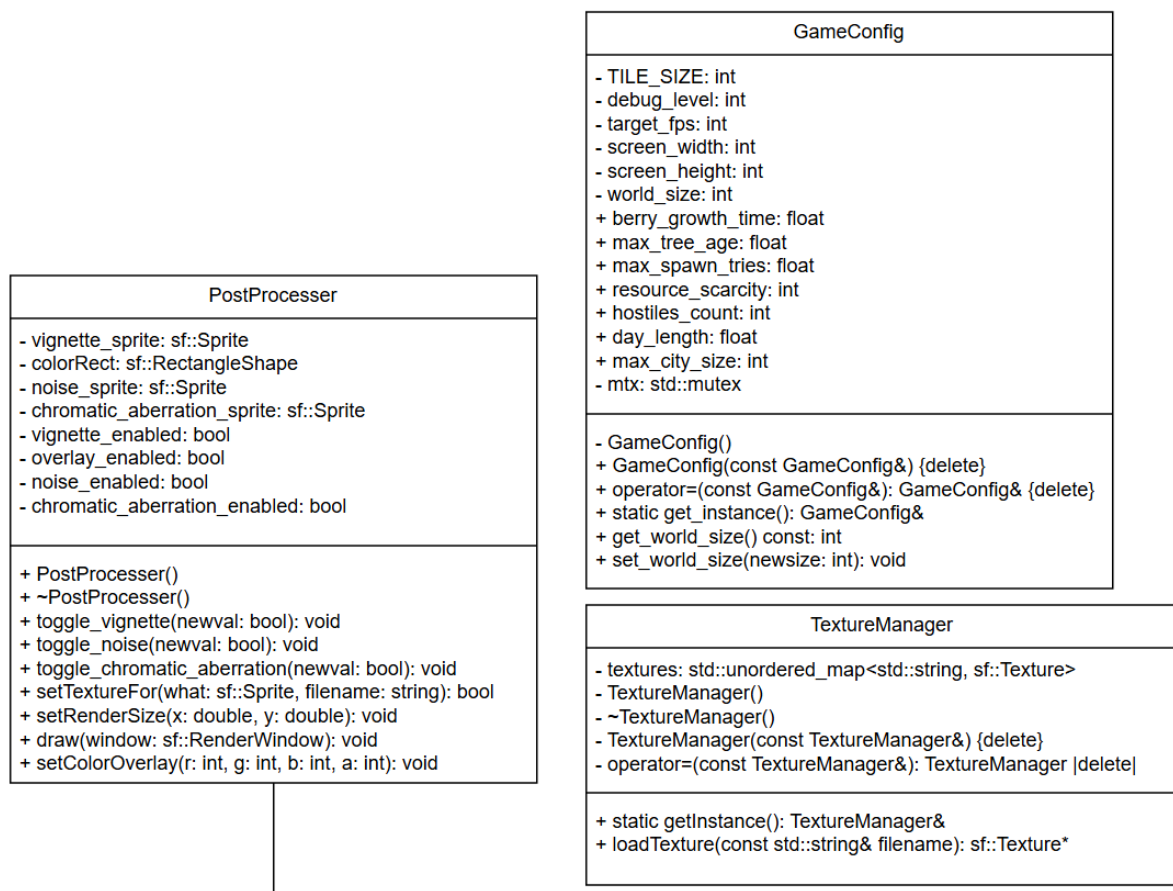
Postprocessor:

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:



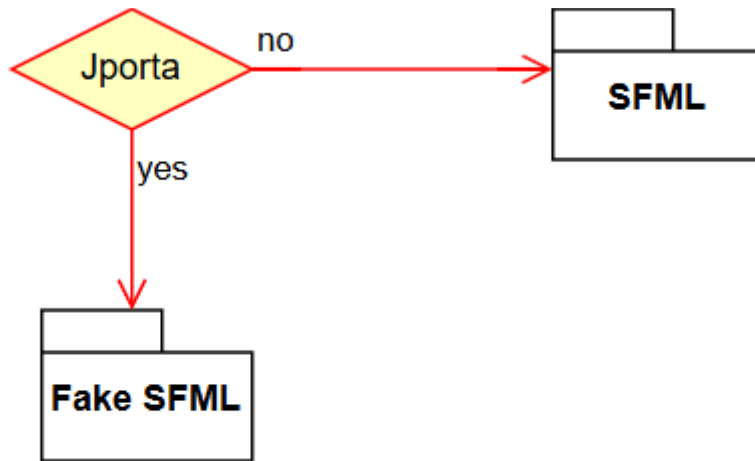
Kikötések:

- 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 funkcióit 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.