

Star\_Trek\_Federation\_in\_Peril

Generated by Doxygen 1.9.5



|                                       |          |
|---------------------------------------|----------|
| <b>1 Data Structure Index</b>         | <b>1</b> |
| 1.1 Data Structures                   | 1        |
| <b>2 File Index</b>                   | <b>3</b> |
| 2.1 File List                         | 3        |
| <b>3 Data Structure Documentation</b> | <b>5</b> |
| 3.1 colordata Struct Reference        | 5        |
| 3.1.1 Detailed Description            | 5        |
| 3.1.2 Field Documentation             | 5        |
| 3.1.2.1 a                             | 5        |
| 3.1.2.2 b                             | 5        |
| 3.1.2.3 g                             | 6        |
| 3.1.2.4 r                             | 6        |
| 3.2 DebugmallocData Struct Reference  | 6        |
| 3.2.1 Field Documentation             | 6        |
| 3.2.1.1 all_alloc_bytes               | 6        |
| 3.2.1.2 all_alloc_count               | 6        |
| 3.2.1.3 alloc_bytes                   | 7        |
| 3.2.1.4 alloc_count                   | 7        |
| 3.2.1.5 head                          | 7        |
| 3.2.1.6 logfile                       | 7        |
| 3.2.1.7 max_block_size                | 7        |
| 3.2.1.8 tail                          | 7        |
| 3.3 DebugmallocEntry Struct Reference | 7        |
| 3.3.1 Field Documentation             | 8        |
| 3.3.1.1 expr                          | 8        |
| 3.3.1.2 file                          | 8        |
| 3.3.1.3 func                          | 8        |
| 3.3.1.4 line                          | 8        |
| 3.3.1.5 next                          | 8        |
| 3.3.1.6 prev                          | 9        |
| 3.3.1.7 real_mem                      | 9        |
| 3.3.1.8 size                          | 9        |
| 3.3.1.9 user_mem                      | 9        |
| 3.4 DinStr Struct Reference           | 9        |
| 3.4.1 Field Documentation             | 9        |
| 3.4.1.1 size                          | 9        |
| 3.4.1.2 str                           | 10       |
| 3.5 enemyship Struct Reference        | 10       |
| 3.5.1 Detailed Description            | 10       |
| 3.5.2 Field Documentation             | 11       |
| 3.5.2.1 centerline_y_coord            | 11       |

|  |    |
|--|----|
| 3.5.2.2 health                           | 11 |
| 3.5.2.3 hitbox_beg_coor                  | 11 |
| 3.5.2.4 hitbox_end_coor                  | 11 |
| 3.5.2.5 movement_dir                     | 11 |
| 3.5.2.6 next_ship                        | 12 |
| 3.5.2.7 prev_ship                        | 12 |
| 3.5.2.8 score_value                      | 12 |
| 3.5.2.9 speed                            | 12 |
| 3.5.2.10 sprite_map_data                 | 12 |
| 3.5.2.11 texture_data                    | 12 |
| 3.5.2.12 x_coor                          | 13 |
| 3.5.2.13 y_coor                          | 13 |
| 3.6 gameassets Struct Reference          | 13 |
| 3.6.1 Detailed Description               | 13 |
| 3.6.2 Field Documentation                | 13 |
| 3.6.2.1 enemy_armada                     | 13 |
| 3.6.2.2 enemy_torpedoes                  | 14 |
| 3.6.2.3 player_ship                      | 14 |
| 3.6.2.4 player_torpedoes                 | 14 |
| 3.6.2.5 star_map                         | 14 |
| 3.7 gameattributes Struct Reference      | 14 |
| 3.7.1 Detailed Description               | 14 |
| 3.7.2 Field Documentation                | 15 |
| 3.7.2.1 enemy_armada_size                | 15 |
| 3.7.2.2 enemy_ships_per_row              | 15 |
| 3.7.2.3 game_score                       | 15 |
| 3.7.2.4 height                           | 15 |
| 3.7.2.5 id                               | 15 |
| 3.7.2.6 isi                              | 15 |
| 3.7.2.7 num_of_rows                      | 15 |
| 3.7.2.8 width                            | 16 |
| 3.8 inputstateinterface Struct Reference | 16 |
| 3.8.1 Detailed Description               | 16 |
| 3.8.2 Field Documentation                | 16 |
| 3.8.2.1 down                             | 16 |
| 3.8.2.2 game_over                        | 16 |
| 3.8.2.3 left                             | 17 |
| 3.8.2.4 n                                | 17 |
| 3.8.2.5 quit                             | 17 |
| 3.8.2.6 restart                          | 17 |
| 3.8.2.7 right                            | 17 |
| 3.8.2.8 torpedo                          | 17 |

|   |    |
|---|----|
| 3.8.2.9 torpedo_ready . . . . .               | 17 |
| 3.8.2.10 up . . . . .                         | 17 |
| 3.8.2.11 y . . . . .                          | 18 |
| 3.9 keymap Struct Reference . . . . .         | 18 |
| 3.9.1 Detailed Description . . . . .          | 18 |
| 3.9.2 Field Documentation . . . . .           | 18 |
| 3.9.2.1 downkey . . . . .                     | 18 |
| 3.9.2.2 leftkey . . . . .                     | 18 |
| 3.9.2.3 rightkey . . . . .                    | 19 |
| 3.9.2.4 torpedokey . . . . .                  | 19 |
| 3.9.2.5 upkey . . . . .                       | 19 |
| 3.10 playership Struct Reference . . . . .    | 19 |
| 3.10.1 Detailed Description . . . . .         | 20 |
| 3.10.2 Field Documentation . . . . .          | 20 |
| 3.10.2.1 centerline_y_coor . . . . .          | 20 |
| 3.10.2.2 health . . . . .                     | 20 |
| 3.10.2.3 hitbox_beg_coor . . . . .            | 20 |
| 3.10.2.4 hitbox_end_coor . . . . .            | 20 |
| 3.10.2.5 score_value . . . . .                | 20 |
| 3.10.2.6 speed . . . . .                      | 21 |
| 3.10.2.7 sprite_map_data . . . . .            | 21 |
| 3.10.2.8 texture_data . . . . .               | 21 |
| 3.10.2.9 x_coor . . . . .                     | 21 |
| 3.10.2.10 y_coor . . . . .                    | 21 |
| 3.11 shipdtt Struct Reference . . . . .       | 21 |
| 3.11.1 Detailed Description . . . . .         | 22 |
| 3.11.2 Field Documentation . . . . .          | 22 |
| 3.11.2.1 health . . . . .                     | 22 |
| 3.11.2.2 score_value . . . . .                | 22 |
| 3.11.2.3 speed . . . . .                      | 22 |
| 3.12 spritemapdata Struct Reference . . . . . | 23 |
| 3.12.1 Detailed Description . . . . .         | 23 |
| 3.12.2 Field Documentation . . . . .          | 23 |
| 3.12.2.1 height . . . . .                     | 23 |
| 3.12.2.2 width . . . . .                      | 23 |
| 3.12.2.3 x_coor . . . . .                     | 23 |
| 3.12.2.4 y_coor . . . . .                     | 23 |
| 3.13 star Struct Reference . . . . .          | 24 |
| 3.13.1 Detailed Description . . . . .         | 24 |
| 3.13.2 Field Documentation . . . . .          | 24 |
| 3.13.2.1 radius . . . . .                     | 24 |
| 3.13.2.2 x_coor . . . . .                     | 24 |

|  |           |
|--|-----------|
| 3.13.2.3 y_coor                          | 25        |
| 3.14 starcolor Struct Reference          | 25        |
| 3.14.1 Detailed Description              | 25        |
| 3.14.2 Field Documentation               | 25        |
| 3.14.2.1 a                               | 25        |
| 3.14.2.2 b                               | 26        |
| 3.14.2.3 g                               | 26        |
| 3.14.2.4 r                               | 26        |
| 3.15 starmap Struct Reference            | 26        |
| 3.15.1 Detailed Description              | 26        |
| 3.15.2 Field Documentation               | 27        |
| 3.15.2.1 color                           | 27        |
| 3.15.2.2 length                          | 27        |
| 3.15.2.3 stars                           | 27        |
| 3.16 texturedata Struct Reference        | 27        |
| 3.16.1 Detailed Description              | 27        |
| 3.16.2 Field Documentation               | 28        |
| 3.16.2.1 height                          | 28        |
| 3.16.2.2 texture_center_x                | 28        |
| 3.16.2.3 texture_center_y                | 28        |
| 3.16.2.4 width                           | 28        |
| 3.17 torpedocolors Struct Reference      | 28        |
| 3.17.1 Detailed Description              | 28        |
| 3.17.2 Field Documentation               | 29        |
| 3.17.2.1 center                          | 29        |
| 3.17.2.2 inner_ring                      | 29        |
| 3.17.2.3 outter_ring                     | 29        |
| 3.18 torpedoshot Struct Reference        | 29        |
| 3.18.1 Detailed Description              | 29        |
| 3.18.2 Field Documentation               | 30        |
| 3.18.2.1 colors                          | 30        |
| 3.18.2.2 damage                          | 30        |
| 3.18.2.3 dir                             | 30        |
| 3.18.2.4 next_torpedo                    | 30        |
| 3.18.2.5 prev_torpedo                    | 30        |
| 3.18.2.6 speed                           | 30        |
| 3.18.2.7 x_coor                          | 30        |
| 3.18.2.8 y_coor                          | 30        |
| <b>4 File Documentation</b>              | <b>31</b> |
| 4.1 data_transfer_types.h File Reference | 31        |
| 4.1.1 Typedef Documentation              | 31        |

|  |    |
|--|----|
| 4.1.1.1 ShipDTT                            | 31 |
| 4.2 data_transfer_types.h                  | 31 |
| 4.3 debugmalloc.h File Reference           | 32 |
| 4.3.1 Macro Definition Documentation       | 32 |
| 4.3.1.1 calloc                             | 32 |
| 4.3.1.2 free                               | 33 |
| 4.3.1.3 malloc                             | 33 |
| 4.3.1.4 realloc                            | 33 |
| 4.3.2 Typedef Documentation                | 33 |
| 4.3.2.1 DebugmallocData                    | 33 |
| 4.3.2.2 DebugmallocEntry                   | 33 |
| 4.3.3 Enumeration Type Documentation       | 33 |
| 4.3.3.1 anonymous enum                     | 33 |
| 4.4 debugmalloc.h                          | 34 |
| 4.5 enemy_hit_management.h File Reference  | 40 |
| 4.5.1 Function Documentation               | 40 |
| 4.5.1.1 explode_enemy_ship_if_dead()       | 40 |
| 4.6 enemy_hit_management.h                 | 41 |
| 4.7 enemy_hit_manangement.c File Reference | 41 |
| 4.7.1 Function Documentation               | 41 |
| 4.7.1.1 explode_enemy_ship_if_dead()       | 41 |
| 4.8 enemy_ship.c File Reference            | 42 |
| 4.8.1 Function Documentation               | 42 |
| 4.8.1.1 find_max_enemy_armada_y_coor()     | 42 |
| 4.8.1.2 free_enemy_armada()                | 43 |
| 4.8.1.3 init_enemy_armada()                | 43 |
| 4.8.1.4 move_enemy_armada()                | 43 |
| 4.8.1.5 pop_enemy_ship()                   | 44 |
| 4.9 enemy_ship.h File Reference            | 44 |
| 4.9.1 Typedef Documentation                | 45 |
| 4.9.1.1 EnemyShip                          | 45 |
| 4.9.2 Function Documentation               | 45 |
| 4.9.2.1 find_max_enemy_armada_y_coor()     | 45 |
| 4.9.2.2 free_enemy_armada()                | 46 |
| 4.9.2.3 init_enemy_armada()                | 46 |
| 4.9.2.4 move_enemy_armada()                | 47 |
| 4.9.2.5 pop_enemy_ship()                   | 47 |
| 4.10 enemy_ship.h                          | 47 |
| 4.11 file_management.c File Reference      | 48 |
| 4.11.1 Function Documentation              | 48 |
| 4.11.1.1 import_ship_dtt()                 | 48 |
| 4.11.1.2 read_texture_data()               | 49 |

|   |    |
|---|----|
| 4.12 file_management.h File Reference . . . . . | 49 |
| 4.12.1 Function Documentation . . . . .         | 50 |
| 4.12.1.1 import_ship_dtt() . . . . .            | 50 |
| 4.12.1.2 read_texture_data() . . . . .          | 50 |
| 4.13 file_management.h . . . . .                | 51 |
| 4.14 fire_management.h File Reference . . . . . | 51 |
| 4.14.1 Function Documentation . . . . .         | 51 |
| 4.14.1.1 fire_enemy_torpedoes() . . . . .       | 51 |
| 4.14.1.2 fire_player_torpedo() . . . . .        | 52 |
| 4.15 fire_management.h . . . . .                | 52 |
| 4.16 fire_managemet.c File Reference . . . . .  | 52 |
| 4.16.1 Function Documentation . . . . .         | 53 |
| 4.16.1.1 fire_enemy_torpedoes() . . . . .       | 53 |
| 4.16.1.2 fire_player_torpedo() . . . . .        | 53 |
| 4.17 game_assets.h File Reference . . . . .     | 54 |
| 4.17.1 Typedef Documentation . . . . .          | 54 |
| 4.17.1.1 GameAssets . . . . .                   | 54 |
| 4.18 game_assets.h . . . . .                    | 54 |
| 4.19 game_attributes.h File Reference . . . . . | 55 |
| 4.19.1 Typedef Documentation . . . . .          | 55 |
| 4.19.1.1 GameAttributes . . . . .               | 55 |
| 4.20 game_attributes.h . . . . .                | 55 |
| 4.21 game_engine.c File Reference . . . . .     | 55 |
| 4.21.1 Function Documentation . . . . .         | 56 |
| 4.21.1.1 free_ship_dtt() . . . . .              | 56 |
| 4.21.1.2 input_timer() . . . . .                | 56 |
| 4.21.1.3 runtime() . . . . .                    | 57 |
| 4.22 game_engine.h File Reference . . . . .     | 57 |
| 4.22.1 Function Documentation . . . . .         | 58 |
| 4.22.1.1 input_timer() . . . . .                | 58 |
| 4.22.1.2 runtime() . . . . .                    | 58 |
| 4.23 game_engine.h . . . . .                    | 58 |
| 4.24 graphics.c File Reference . . . . .        | 59 |
| 4.24.1 Function Documentation . . . . .         | 60 |
| 4.24.1.1 clear_screen() . . . . .               | 60 |
| 4.24.1.2 create_font() . . . . .                | 60 |
| 4.24.1.3 create_textures() . . . . .            | 60 |
| 4.24.1.4 create_window() . . . . .              | 61 |
| 4.24.1.5 destroy_textures() . . . . .           | 61 |
| 4.24.1.6 draw_background() . . . . .            | 61 |
| 4.24.1.7 draw_end_screen() . . . . .            | 62 |
| 4.24.1.8 draw_enemy_ships() . . . . .           | 62 |



|   |    |
|---|----|
| 4.24.1.9 draw_LCARS_bacground()             | 62 |
| 4.24.1.10 draw_player_ship()                | 63 |
| 4.24.1.11 draw_score()                      | 63 |
| 4.24.1.12 draw_torpedo()                    | 63 |
| 4.24.1.13 load_sdl_texture()                | 64 |
| 4.24.1.14 open_font()                       | 64 |
| 4.24.1.15 render_screen()                   | 65 |
| 4.24.1.16 sdl_init()                        | 65 |
| 4.25 graphics.h File Reference              | 65 |
| 4.25.1 Function Documentation               | 66 |
| 4.25.1.1 clear_screen()                     | 66 |
| 4.25.1.2 create_font()                      | 67 |
| 4.25.1.3 create_textures()                  | 67 |
| 4.25.1.4 create_window()                    | 67 |
| 4.25.1.5 destroy_textures()                 | 68 |
| 4.25.1.6 draw_background()                  | 68 |
| 4.25.1.7 draw_end_screen()                  | 68 |
| 4.25.1.8 draw_enemy_ships()                 | 69 |
| 4.25.1.9 draw_LCARS_bacground()             | 69 |
| 4.25.1.10 draw_player_ship()                | 69 |
| 4.25.1.11 draw_score()                      | 70 |
| 4.25.1.12 draw_torpedo()                    | 70 |
| 4.25.1.13 render_screen()                   | 70 |
| 4.26 graphics.h                             | 71 |
| 4.27 hit_management.c File Reference        | 71 |
| 4.27.1 Function Documentation               | 72 |
| 4.27.1.1 manage_enemy_hits()                | 72 |
| 4.27.1.2 manage_player_hits()               | 72 |
| 4.27.1.3 remove_torpedo_if_out_of_bounds()  | 73 |
| 4.28 hit_management.h File Reference        | 73 |
| 4.28.1 Function Documentation               | 73 |
| 4.28.1.1 manage_enemy_hits()                | 74 |
| 4.28.1.2 remove_torpedo_if_out_of_bounds()  | 74 |
| 4.29 hit_management.h                       | 74 |
| 4.30 input_state_interface.h File Reference | 75 |
| 4.30.1 Typedef Documentation                | 75 |
| 4.30.1.1 InputStateInterface                | 75 |
| 4.31 input_state_interface.h                | 75 |
| 4.32 keymap.h File Reference                | 76 |
| 4.32.1 Typedef Documentation                | 76 |
| 4.32.1.1 KeyMap                             | 76 |
| 4.33 keymap.h                               | 76 |

|   |    |
|---|----|
| 4.34 main.c File Reference . . . . .                      | 76 |
| 4.34.1 Function Documentation . . . . .                   | 77 |
| 4.34.1.1 main() . . . . .                                 | 77 |
| 4.35 player_hit_management.c File Reference . . . . .     | 77 |
| 4.35.1 Function Documentation . . . . .                   | 77 |
| 4.35.1.1 explode_player_ship_if_dead() . . . . .          | 77 |
| 4.36 player_hit_management.h File Reference . . . . .     | 78 |
| 4.36.1 Function Documentation . . . . .                   | 78 |
| 4.36.1.1 explode_player_ship_if_dead() . . . . .          | 78 |
| 4.37 player_hit_management.h . . . . .                    | 78 |
| 4.38 player_ship.c File Reference . . . . .               | 79 |
| 4.38.1 Function Documentation . . . . .                   | 79 |
| 4.38.1.1 free_player_ship() . . . . .                     | 79 |
| 4.38.1.2 init_player_ship() . . . . .                     | 79 |
| 4.38.1.3 move_player_ship() . . . . .                     | 80 |
| 4.39 player_ship.h File Reference . . . . .               | 80 |
| 4.39.1 Typedef Documentation . . . . .                    | 81 |
| 4.39.1.1 PlayerShip . . . . .                             | 81 |
| 4.39.2 Function Documentation . . . . .                   | 81 |
| 4.39.2.1 free_player_ship() . . . . .                     | 81 |
| 4.39.2.2 init_player_ship() . . . . .                     | 82 |
| 4.39.2.3 move_player_ship() . . . . .                     | 82 |
| 4.40 player_ship.h . . . . .                              | 83 |
| 4.41 random_number_in_interval.c File Reference . . . . . | 83 |
| 4.41.1 Function Documentation . . . . .                   | 83 |
| 4.41.1.1 random_number_in_range() . . . . .               | 84 |
| 4.42 random_number_in_interval.h File Reference . . . . . | 84 |
| 4.42.1 Function Documentation . . . . .                   | 84 |
| 4.42.1.1 random_number_in_range() . . . . .               | 84 |
| 4.43 random_number_in_interval.h . . . . .                | 85 |
| 4.44 star_map.c File Reference . . . . .                  | 85 |
| 4.44.1 Function Documentation . . . . .                   | 85 |
| 4.44.1.1 advance_starmap_frame() . . . . .                | 85 |
| 4.44.1.2 free_starmap() . . . . .                         | 86 |
| 4.44.1.3 starmap_init() . . . . .                         | 86 |
| 4.45 star_map.h File Reference . . . . .                  | 87 |
| 4.45.1 Typedef Documentation . . . . .                    | 87 |
| 4.45.1.1 Star . . . . .                                   | 87 |
| 4.45.1.2 StarColor . . . . .                              | 88 |
| 4.45.1.3 StarMap . . . . .                                | 88 |
| 4.45.2 Function Documentation . . . . .                   | 88 |
| 4.45.2.1 advance_starmap_frame() . . . . .                | 88 |

|  |     |
|--|-----|
| 4.45.2.2 free_starmap()                      | 89  |
| 4.45.2.3 starmap_init()                      | 89  |
| 4.46 star_map.h                              | 89  |
| 4.47 string_operations.c File Reference      | 90  |
| 4.47.1 Function Documentation                | 90  |
| 4.47.1.1 dinstr_alloc()                      | 90  |
| 4.48 string_operations.h File Reference      | 91  |
| 4.48.1 Typedef Documentation                 | 91  |
| 4.48.1.1 DinStr                              | 91  |
| 4.49 string_operations.h                     | 91  |
| 4.50 texture_data.h File Reference           | 91  |
| 4.50.1 Typedef Documentation                 | 92  |
| 4.50.1.1 SpriteMapData                       | 92  |
| 4.50.1.2 TextureData                         | 92  |
| 4.51 texture_data.h                          | 92  |
| 4.52 torpedo.c File Reference                | 93  |
| 4.52.1 Function Documentation                | 93  |
| 4.52.1.1 add_torpedo_shot()                  | 93  |
| 4.52.1.2 free_torpedoes()                    | 94  |
| 4.52.1.3 move_torpedoes()                    | 94  |
| 4.52.1.4 pop_torpedo_shot()                  | 94  |
| 4.53 torpedo.h File Reference                | 95  |
| 4.53.1 Typedef Documentation                 | 96  |
| 4.53.1.1 ColorData                           | 96  |
| 4.53.1.2 TorpedoColors                       | 96  |
| 4.53.1.3 TorpedoShot                         | 96  |
| 4.53.2 Function Documentation                | 96  |
| 4.53.2.1 add_torpedo_shot()                  | 96  |
| 4.53.2.2 free_torpedoes()                    | 97  |
| 4.53.2.3 move_torpedoes()                    | 97  |
| 4.53.2.4 pop_torpedo_shot()                  | 97  |
| 4.54 torpedo.h                               | 98  |
| 4.55 torpedo_hit_management.c File Reference | 98  |
| 4.55.1 Function Documentation                | 99  |
| 4.55.1.1 explode_torpedo()                   | 99  |
| 4.55.1.2 is_torpedo_out_of_bounds()          | 99  |
| 4.56 torpedo_hit_management.h File Reference | 99  |
| 4.56.1 Function Documentation                | 100 |
| 4.56.1.1 explode_torpedo()                   | 100 |
| 4.56.1.2 is_torpedo_out_of_bounds()          | 100 |
| 4.57 torpedo_hit_management.h                | 101 |
| 4.58 ui_input.c File Reference               | 101 |

|  |            |
|--|------------|
| 4.58.1 Function Documentation . . . . .  | 101        |
| 4.58.1.1 user_input() . . . . .          | 101        |
| 4.59 ui_input.h File Reference . . . . . | 102        |
| 4.59.1 Function Documentation . . . . .  | 102        |
| 4.59.1.1 user_input() . . . . .          | 102        |
| 4.60 ui_input.h . . . . .                | 103        |
| <b>Index</b>                             | <b>105</b> |

# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

|  |    |
|--|----|
| <a href="#">colordata</a>                  |    |
| ColorData . . . . .                        | 5  |
| <a href="#">DebugmallocData</a> . . . . .  | 6  |
| <a href="#">DebugmallocEntry</a> . . . . . | 7  |
| <a href="#">DinStr</a> . . . . .           | 9  |
| <a href="#">enemyship</a>                  |    |
| EnemyShip . . . . .                        | 10 |
| <a href="#">gameassets</a>                 |    |
| GameAssets . . . . .                       | 13 |
| <a href="#">gameattributes</a>             |    |
| GameAttributes . . . . .                   | 14 |
| <a href="#">inputstateinterface</a>        |    |
| InputStateInterface . . . . .              | 16 |
| <a href="#">keymap</a>                     |    |
| KeyMap . . . . .                           | 18 |
| <a href="#">playership</a>                 |    |
| PlayerShip . . . . .                       | 19 |
| <a href="#">shipdtl</a>                    |    |
| ShipDTT . . . . .                          | 21 |
| <a href="#">spritemapdata</a>              |    |
| SpriteMapData . . . . .                    | 23 |
| <a href="#">star</a>                       |    |
| Star . . . . .                             | 24 |
| <a href="#">starcolor</a>                  |    |
| StarColor . . . . .                        | 25 |
| <a href="#">starmap</a>                    |    |
| StarMap . . . . .                          | 26 |
| <a href="#">texturedata</a>                |    |
| TextureData . . . . .                      | 27 |
| <a href="#">torpedocolors</a>              |    |
| TorpedoColors . . . . .                    | 28 |
| <a href="#">torpedoshot</a>                |    |
| TorpedoShot . . . . .                      | 29 |



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

|                                       |     |
|---------------------------------------|-----|
| data_transfer_types.h . . . . .       | 31  |
| debugmalloc.h . . . . .               | 32  |
| enemy_hit_management.h . . . . .      | 40  |
| enemy_hit_mananagement.c . . . . .    | 41  |
| enemy_ship.c . . . . .                | 42  |
| enemy_ship.h . . . . .                | 44  |
| file_management.c . . . . .           | 48  |
| file_management.h . . . . .           | 49  |
| fire_management.h . . . . .           | 51  |
| fire_managemet.c . . . . .            | 52  |
| game_assets.h . . . . .               | 54  |
| game_attributes.h . . . . .           | 55  |
| game_engine.c . . . . .               | 55  |
| game_engine.h . . . . .               | 57  |
| graphics.c . . . . .                  | 59  |
| graphics.h . . . . .                  | 65  |
| hit_management.c . . . . .            | 71  |
| hit_management.h . . . . .            | 73  |
| input_state_interface.h . . . . .     | 75  |
| keymap.h . . . . .                    | 76  |
| main.c . . . . .                      | 76  |
| player_hit_management.c . . . . .     | 77  |
| player_hit_management.h . . . . .     | 78  |
| player_ship.c . . . . .               | 79  |
| player_ship.h . . . . .               | 80  |
| random_number_in_interval.c . . . . . | 83  |
| random_number_in_interval.h . . . . . | 84  |
| star_map.c . . . . .                  | 85  |
| star_map.h . . . . .                  | 87  |
| string_operations.c . . . . .         | 90  |
| string_operations.h . . . . .         | 91  |
| texture_data.h . . . . .              | 91  |
| torpedo.c . . . . .                   | 93  |
| torpedo.h . . . . .                   | 95  |
| torpedo_hit_management.c . . . . .    | 98  |
| torpedo_hit_management.h . . . . .    | 99  |
| ui_input.c . . . . .                  | 101 |
| ui_input.h . . . . .                  | 102 |





## Chapter 3

# Data Structure Documentation

### 3.1 colordata Struct Reference

ColorData.

```
#include <torpedo.h>
```

#### Data Fields

- int [r](#)
- int [g](#)
- int [b](#)
- int [a](#)

#### 3.1.1 Detailed Description

ColorData.

#### 3.1.2 Field Documentation

##### 3.1.2.1 [a](#)

```
int a
```

##### 3.1.2.2 [b](#)

```
int b
```

### 3.1.2.3 g

```
int g
```

### 3.1.2.4 r

```
int r
```

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

- [torpedo.h](#)

## 3.2 DebugmallocData Struct Reference

```
#include <debugmalloc.h>
```

### Data Fields

- char [logfile](#) [256]
- long [max\\_block\\_size](#)
- long [alloc\\_count](#)
- long long [alloc\\_bytes](#)
- long [all\\_alloc\\_count](#)
- long long [all\\_alloc\\_bytes](#)
- [DebugmallocEntry](#) head [[debugmalloc\\_tablesize](#)]
- [DebugmallocEntry](#) tail [[debugmalloc\\_tablesize](#)]

### 3.2.1 Field Documentation

#### 3.2.1.1 all\_alloc\_bytes

```
long long all_alloc_bytes
```

#### 3.2.1.2 all\_alloc\_count

```
long all_alloc_count
```

### 3.2.1.3 alloc\_bytes

```
long long alloc_bytes
```

### 3.2.1.4 alloc\_count

```
long alloc_count
```

### 3.2.1.5 head

```
DebugmallocEntry head[debugmalloc_tablesize]
```

### 3.2.1.6 logfile

```
char logfile[256]
```

### 3.2.1.7 max\_block\_size

```
long max_block_size
```

### 3.2.1.8 tail

```
DebugmallocEntry tail[debugmalloc_tablesize]
```

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

- [debugmalloc.h](#)

## 3.3 DebugmallocEntry Struct Reference

```
#include <debugmalloc.h>
```

## Data Fields

- void \* [real\\_mem](#)
- void \* [user\\_mem](#)
- size\_t [size](#)
- char [file](#) [64]
- unsigned [line](#)
- char [func](#) [32]
- char [expr](#) [128]
- struct [DebugmallocEntry](#) \* [prev](#)
- struct [DebugmallocEntry](#) \* [next](#)

### 3.3.1 Field Documentation

#### 3.3.1.1 [expr](#)

```
char expr[128]
```

#### 3.3.1.2 [file](#)

```
char file[64]
```

#### 3.3.1.3 [func](#)

```
char func[32]
```

#### 3.3.1.4 [line](#)

```
unsigned line
```

#### 3.3.1.5 [next](#)

```
struct DebugmallocEntry * next
```

### 3.3.1.6 prev

```
struct DebugmallocEntry* prev
```

### 3.3.1.7 real\_mem

```
void* real_mem
```

### 3.3.1.8 size

```
size_t size
```

### 3.3.1.9 user\_mem

```
void* user_mem
```

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

- [debugmalloc.h](#)

## 3.4 DinStr Struct Reference

```
#include <string_operations.h>
```

### Data Fields

- int [size](#)
- char \* [str](#)

### 3.4.1 Field Documentation

#### 3.4.1.1 size

```
int size
```

### 3.4.1.2 str

```
char* str
```

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

- [string\\_operations.h](#)

## 3.5 enemyship Struct Reference

EnemyShip.

```
#include <enemy_ship.h>
```

### Data Fields

- int [y\\_coor](#)  
*Az ellenseges hajo y koordinataja.*
- int [x\\_coor](#)  
*Az ellenseges hajo x koordinataja.*
- int [movement\\_dir](#)  
*Az ellenseges hajo mozgasi iranya.*
- int [hitbox\\_beg\\_coor](#)  
*A hitbox bal szele.*
- int [hitbox\\_end\\_coor](#)  
*A hitbox jobb szele.*
- int [centerline\\_y\\_coor](#)  
*A hitbox also hatara (ahonnan mar erzekeli a sprite a talalatot)*
- [TextureData texture\\_data](#)  
*Az ellenseges hajo textura adatai (meretek)*
- [SpriteMapData sprite\\_map\\_data](#)  
*Az ellenseges hajo spritemap meretei (SDL2 igenyli)*
- int [speed](#)  
*Az ellenseges hajo mozgasi sebessege.*
- int [health](#)  
*Az ellenseges hajo HP-ja.*
- int [score\\_value](#)  
*Az ellenseges hajo lelovesevel megszerezhető pont.*
- struct [enemyship](#) \* [next\\_ship](#)  
*Az ellenseges hajokat tartalmazó lancolt lista következő eleme.*
- struct [enemyship](#) \* [prev\\_ship](#)  
*Az ellenseges hajokat tartalmazó lancolt lista előző eleme.*

### 3.5.1 Detailed Description

EnemyShip.

Az ellenseges hajokat tarolo lancolt lista listaeleme

## 3.5.2 Field Documentation

### 3.5.2.1 centerline\_y\_coor

```
int centerline_y_coor
```

A hitbox also hatara (ahonnan mar erzekeli a sprite a talalatot)

### 3.5.2.2 health

```
int health
```

Az ellenseges hajo HP-ja.

### 3.5.2.3 hitbox\_beg\_coor

```
int hitbox_beg_coor
```

A hitbox bal szele.

### 3.5.2.4 hitbox\_end\_coor

```
int hitbox_end_coor
```

A hitbox jobb szele.

### 3.5.2.5 movement\_dir

```
int movement_dir
```

Az ellenseges hajo mozgasi iranya.

### 3.5.2.6 next\_ship

```
struct enemyship* next_ship
```

Az ellenseges hajokat tartalmazó láncolt lista következő eleme.

### 3.5.2.7 prev\_ship

```
struct enemyship* prev_ship
```

Az ellenseges hajokat tartalmazó láncolt lista előző eleme.

### 3.5.2.8 score\_value

```
int score_value
```

Az ellenseges hajó lelovesével megszerezhető pont.

### 3.5.2.9 speed

```
int speed
```

Az ellenseges hajó mozgási sebessége.

### 3.5.2.10 sprite\_map\_data

```
SpriteMapData sprite_map_data
```

Az ellenseges hajó spritemap méretei (SDL2 igényli)

### 3.5.2.11 texture\_data

```
TextureData texture_data
```

Az ellenseges hajó textúra adatai (méretek)



### 3.5.2.12 x\_coor

```
int x_coor
```

Az ellenseges hajo x koordinataja.

### 3.5.2.13 y\_coor

```
int y_coor
```

Az ellenseges hajo y koordinataja.

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

- [enemy\\_ship.h](#)

## 3.6 gameassets Struct Reference

GameAssets.

```
#include <game_assets.h>
```

### Data Fields

- [StarMap](#) \* [star\\_map](#)
- [PlayerShip](#) \* [player\\_ship](#)
- [EnemyShip](#) \* [enemy\\_armada](#)
- [TorpedoShot](#) \* [player\\_torpedoes](#)
- [TorpedoShot](#) \* [enemy\\_torpedoes](#)

### 3.6.1 Detailed Description

GameAssets.

### 3.6.2 Field Documentation

#### 3.6.2.1 enemy\_armada

```
EnemyShip* enemy_armada
```

### 3.6.2.2 enemy\_torpedoes

`TorpedoShot*` enemy\_torpedoes

### 3.6.2.3 player\_ship

`PlayerShip*` player\_ship

### 3.6.2.4 player\_torpedoes

`TorpedoShot*` player\_torpedoes

### 3.6.2.5 star\_map

`StarMap*` star\_map

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

- [game\\_assets.h](#)

## 3.7 gameattributes Struct Reference

GameAttributes.

```
#include <game_attributes.h>
```

### Data Fields

- int [width](#)
- int [height](#)
- int [enemy\\_armada\\_size](#)
- int [num\\_of\\_rows](#)
- int [enemy\\_ships\\_per\\_row](#)
- int [game\\_score](#)
- [InputStateInterface](#) [isi](#)
- [SDL\\_TimerID](#) [id](#)

### 3.7.1 Detailed Description

GameAttributes.

## 3.7.2 Field Documentation

### 3.7.2.1 enemy\_armada\_size

```
int enemy_armada_size
```

### 3.7.2.2 enemy\_ships\_per\_row

```
int enemy_ships_per_row
```

### 3.7.2.3 game\_score

```
int game_score
```

### 3.7.2.4 height

```
int height
```

### 3.7.2.5 id

```
SDL_TimerID id
```

### 3.7.2.6 isi

```
InputStateInterface isi
```

### 3.7.2.7 num\_of\_rows

```
int num_of_rows
```

### 3.7.2.8 width

```
int width
```

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

- [game\\_attributes.h](#)

## 3.8 inputstateinterface Struct Reference

InputStateInterface.

```
#include <input_state_interface.h>
```

### Data Fields

- bool [quit](#)
- bool [restart](#)
- bool [game\\_over](#)
- bool [up](#)
- bool [down](#)
- bool [left](#)
- bool [right](#)
- bool [y](#)
- bool [n](#)
- bool [torpedo](#)
- bool [torpedo\\_ready](#)

### 3.8.1 Detailed Description

InputStateInterface.

### 3.8.2 Field Documentation

#### 3.8.2.1 down

```
bool down
```

#### 3.8.2.2 game\_over

```
bool game_over
```

### 3.8.2.3 left

`bool left`

### 3.8.2.4 n

`bool n`

### 3.8.2.5 quit

`bool quit`

### 3.8.2.6 restart

`bool restart`

### 3.8.2.7 right

`bool right`

### 3.8.2.8 torpedo

`bool torpedo`

### 3.8.2.9 torpedo\_ready

`bool torpedo_ready`

### 3.8.2.10 up

`bool up`

### 3.8.2.11 y

bool y

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

- [input\\_state\\_interface.h](#)

## 3.9 keymap Struct Reference

KeyMap.

```
#include <keymap.h>
```

### Data Fields

- char \* [upkey](#)
- char \* [downkey](#)
- char \* [leftkey](#)
- char \* [rightkey](#)
- char \* [torpedokey](#)

### 3.9.1 Detailed Description

KeyMap.

### 3.9.2 Field Documentation

#### 3.9.2.1 downkey

char\* downkey

#### 3.9.2.2 leftkey

char\* leftkey

### 3.9.2.3 rightkey

```
char* rightkey
```

### 3.9.2.4 torpedokey

```
char* torpedokey
```

### 3.9.2.5 upkey

```
char* upkey
```

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

- [keymap.h](#)

## 3.10 playership Struct Reference

PlayerShip.

```
#include <player_ship.h>
```

### Data Fields

- int [y\\_coor](#)  
*A hajo y koordinataja.*
- int [x\\_coor](#)  
*A hajo x koordinataja.*
- int [hitbox\\_beg\\_coor](#)  
*A hitbox bal szele.*
- int [hitbox\\_end\\_coor](#)  
*A hitbox jobb szele.*
- int [centerline\\_y\\_coor](#)  
*A hitbox also hatara (ahonnan mar erzekeli a sprite a talalatot)*
- [TextureData texture\\_data](#)  
*A hajo textura adatai (meretek)*
- [SpriteMapData sprite\\_map\\_data](#)  
*A hajo spritemap meretei (SDL2 igenyli)*
- int [speed](#)  
*A hajo mozgasi sebessege.*
- int [health](#)  
*A hajo HP-ja.*
- int [score\\_value](#)  
*A hajo elpusztulasanal elveszített pont.*

### 3.10.1 Detailed Description

PlayerShip.

A jatekos hajokat tarolo adatstruktura.

### 3.10.2 Field Documentation

#### 3.10.2.1 centerline\_y\_coor

```
int centerline_y_coor
```

A hitbox also hatara (ahonnan mar erzekeli a sprite a talalatot)

#### 3.10.2.2 health

```
int health
```

A hajo HP-ja.

#### 3.10.2.3 hitbox\_beg\_coor

```
int hitbox_beg_coor
```

A hitbox bal szele.

#### 3.10.2.4 hitbox\_end\_coor

```
int hitbox_end_coor
```

A hitbox jobb szele.

#### 3.10.2.5 score\_value

```
int score_value
```

A hajo elpusztulasanal elveszített pont.



### 3.10.2.6 speed

```
int speed
```

A hajo mozgasi sebessege.

### 3.10.2.7 sprite\_map\_data

```
SpriteMapData sprite_map_data
```

A hajo spritemap meretei (SDL2 igenyli)

### 3.10.2.8 texture\_data

```
TextureData texture_data
```

A hajo textura adatai (meretek)

### 3.10.2.9 x\_coor

```
int x_coor
```

A hajo x koordinataja.

### 3.10.2.10 y\_coor

```
int y_coor
```

A hajo y koordinataja.

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

- [player\\_ship.h](#)

## 3.11 shipdtt Struct Reference

ShipDTT.

```
#include <data_transfer_types.h>
```

## Data Fields

- int [speed](#)  
*A hajo sebessege.*
- int [health](#)  
*A hajo HP-ja.*
- int [score\\_value](#)  
*A hajo pont-erteke.*

### 3.11.1 Detailed Description

ShipDTT.

### 3.11.2 Field Documentation

#### 3.11.2.1 health

```
int health
```

A hajo HP-ja.

#### 3.11.2.2 score\_value

```
int score_value
```

A hajo pont-erteke.

#### 3.11.2.3 speed

```
int speed
```

A hajo sebessege.

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

- [data\\_transfer\\_types.h](#)

## 3.12 spritemapdata Struct Reference

SpriteMapData.

```
#include <texture_data.h>
```

### Data Fields

- int [x\\_coor](#)
- int [y\\_coor](#)
- int [width](#)
- int [height](#)

### 3.12.1 Detailed Description

SpriteMapData.

### 3.12.2 Field Documentation

#### 3.12.2.1 height

```
int height
```

#### 3.12.2.2 width

```
int width
```

#### 3.12.2.3 x\_coor

```
int x_coor
```

#### 3.12.2.4 y\_coor

```
int y_coor
```

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

- [texture\\_data.h](#)

## 3.13 star Struct Reference

Star.

```
#include <star_map.h>
```

### Data Fields

- int `y_coor`  
*A csillagot jelkepző kor y koordinátája.*
- int `x_coor`  
*A csillagot jelkepző kor x koordinátája.*
- int `radius`  
*A csillagot jelkepző kor sugara.*

### 3.13.1 Detailed Description

Star.

Ez az adatstruktúra tárolja a hatter egy csillagot. Értékei a csillag kirajzolásához szükséges koordináták és sugár. Ez az adattartó a függvényhíváskor megadando parameterlisták lerövidítését, illetve az összetartozó adatok egy helyen tartását szolgálja.

### 3.13.2 Field Documentation

#### 3.13.2.1 radius

```
int radius
```

A csillagot jelkepző kor sugara.

#### 3.13.2.2 x\_coor

```
int x_coor
```

A csillagot jelkepző kor x koordinátája.

### 3.13.2.3 y\_coor

```
int y_coor
```

A csillagot jelkepző y koordinátája.

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

- [star\\_map.h](#)

## 3.14 starcolor Struct Reference

StarColor.

```
#include <star_map.h>
```

### Data Fields

- int [r](#)  
*A csillag RGBA piros értéke.*
- int [g](#)  
*A csillag RGBA zöld értéke.*
- int [b](#)  
*A csillag RGBA kék értéke.*
- int [a](#)  
*A csillag RGBA alfa értéke (ez határozza meg a csillag áttetszőségét).*

### 3.14.1 Detailed Description

StarColor.

Ez az adatstruktúra tárolja a hatter csillaganak színet. Értékei a csillag RGBA-ban meghatározott színértékei. Ez az adatterelő a függvényhíváskor megadando parameterlistak leroviditeset szolgálja.

### 3.14.2 Field Documentation

#### 3.14.2.1 a

```
int a
```

A csillag RGBA alfa értéke (ez határozza meg a csillag áttetszőségét).

#### 3.14.2.2 b

```
int b
```

A csillag RGBA kek erteke.

#### 3.14.2.3 g

```
int g
```

A csillag RGBA zold erteke.

#### 3.14.2.4 r

```
int r
```

A csillag RGBA piros erteke.

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

- [star\\_map.h](#)

## 3.15 starmap Struct Reference

StarMap.

```
#include <star_map.h>
```

### Data Fields

- int [length](#)  
*A lista hossza.*
- [Star](#) \* [stars](#)  
*A csillagokat tarolo lista.*
- [StarColor](#) [color](#)  
*A csillagok szinet tarolo struktura.*

### 3.15.1 Detailed Description

StarMap.

Ez az adatstruktura tarolja a hatter osszes csillagat. Ertekei a lista hossza, a csillagokat tartalmazo lista, illetve azok szine. Ez az adattarolo a hatter csillagainak konnyu letehozasat, tarolasat es felszabaditasat szolgalja.

### 3.15.2 Field Documentation

#### 3.15.2.1 color

`StarColor` color

A csillagok szinet tarolo struktura.

#### 3.15.2.2 length

`int` length

A lista hossza.

#### 3.15.2.3 stars

`Star*` stars

A csillagokat tarolo lista.

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

- [star\\_map.h](#)

## 3.16 texturedata Struct Reference

TextureData.

```
#include <texture_data.h>
```

### Data Fields

- `int` [width](#)
- `int` [height](#)
- `int` [texture\\_center\\_x](#)
- `int` [texture\\_center\\_y](#)

### 3.16.1 Detailed Description

TextureData.

### 3.16.2 Field Documentation

#### 3.16.2.1 height

```
int height
```

#### 3.16.2.2 texture\_center\_x

```
int texture_center_x
```

#### 3.16.2.3 texture\_center\_y

```
int texture_center_y
```

#### 3.16.2.4 width

```
int width
```

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

- [texture\\_data.h](#)

## 3.17 torpedocolors Struct Reference

TorpedoColors.

```
#include <torpedo.h>
```

### Data Fields

- [ColorData](#) outter\_ring
- [ColorData](#) inner\_ring
- [ColorData](#) center

#### 3.17.1 Detailed Description

TorpedoColors.



### 3.17.2 Field Documentation

#### 3.17.2.1 center

[ColorData](#) center

#### 3.17.2.2 inner\_ring

[ColorData](#) inner\_ring

#### 3.17.2.3 outter\_ring

[ColorData](#) outter\_ring

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

- [torpedo.h](#)

## 3.18 torpedoshot Struct Reference

TorpedoShot.

```
#include <torpedo.h>
```

### Data Fields

- int [x\\_coor](#)
- int [y\\_coor](#)
- int [damage](#)
- int [speed](#)
- int [dir](#)
- [TorpedoColors](#) colors
- struct [torpedoshot](#) \* [next\\_torpedo](#)
- struct [torpedoshot](#) \* [prev\\_torpedo](#)

### 3.18.1 Detailed Description

TorpedoShot.

### 3.18.2 Field Documentation

#### 3.18.2.1 colors

`TorpedoColors` colors

#### 3.18.2.2 damage

`int` damage

#### 3.18.2.3 dir

`int` dir

#### 3.18.2.4 next\_torpedo

`struct torpedo`shot\* next\_torpedo

#### 3.18.2.5 prev\_torpedo

`struct torpedo`shot\* prev\_torpedo

#### 3.18.2.6 speed

`int` speed

#### 3.18.2.7 x\_coor

`int` x\_coor

#### 3.18.2.8 y\_coor

`int` y\_coor

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

- [torpedo.h](#)

## Chapter 4

# File Documentation

### 4.1 data\_transfer\_types.h File Reference

#### Data Structures

- struct `shipdtt`  
*ShipDTT.*

#### Typedefs

- typedef struct `shipdtt` `ShipDTT`  
*ShipDTT.*

#### 4.1.1 Typedef Documentation

##### 4.1.1.1 ShipDTT

```
typedef struct shipdtt ShipDTT
```

*ShipDTT.*

### 4.2 data\_transfer\_types.h

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

```
1
5 #ifndef DATA_TRANSFER_TYPES_H_INCLUDED
6 #define DATA_TRANSFER_TYPES_H_INCLUDED
7
8
13 typedef struct shipdtt{
14     int speed;
15     int health;
16     int score_value;
17 }ShipDTT;
18
19 #endif // DATA_TRANSFER_TYPES_H_INCLUDED
```

## 4.3 debugmalloc.h File Reference

```
#include <stdbool.h>
#include <stddef.h>
#include <stdlib.h>
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include <stdarg.h>
#include <unistd.h>
```

### Data Structures

- struct [DebugmallocEntry](#)
- struct [DebugmallocData](#)

### Macros

- #define [malloc](#)(S) debugmalloc\_malloc\_full((S), "malloc", #S, \_\_FILE\_\_, \_\_LINE\_\_, false)
- #define [calloc](#)(N, S) debugmalloc\_malloc\_full((N)\*(S), "calloc", #N " ", " #S, \_\_FILE\_\_, \_\_LINE\_\_, true)
- #define [realloc](#)(P, S) debugmalloc\_realloc\_full((P), (S), "realloc", #S, \_\_FILE\_\_, \_\_LINE\_\_)
- #define [free](#)(P) debugmalloc\_free\_full((P), "free", \_\_FILE\_\_, \_\_LINE\_\_)

### Typedefs

- typedef struct [DebugmallocEntry](#) [DebugmallocEntry](#)
- typedef struct [DebugmallocData](#) [DebugmallocData](#)

### Enumerations

- enum { [debugmalloc\\_canary\\_size](#) = 64 , [debugmalloc\\_canary\\_char](#) = 'K' , [debugmalloc\\_tablesize](#) = 256 , [debugmalloc\\_max\\_block\\_size\\_default](#) = 1048576 }

## 4.3.1 Macro Definition Documentation

### 4.3.1.1 calloc

```
#define calloc(  
    N,  
    S ) debugmalloc_malloc_full((N)*(S), "calloc", #N " ", " #S, __FILE__, __LINE__↵  
, true)
```

#### 4.3.1.2 free

```
#define free(  
    P ) debugmalloc_free_full((P), "free", __FILE__, __LINE__)
```

#### 4.3.1.3 malloc

```
#define malloc(  
    S ) debugmalloc_malloc_full((S), "malloc", #S, __FILE__, __LINE__, false)
```

#### 4.3.1.4 realloc

```
#define realloc(  
    P,  
    S ) debugmalloc_realloc_full((P), (S), "realloc", #S, __FILE__, __LINE__)
```

### 4.3.2 Typedef Documentation

#### 4.3.2.1 DebugmallocData

```
typedef struct DebugmallocData DebugmallocData
```

#### 4.3.2.2 DebugmallocEntry

```
typedef struct DebugmallocEntry DebugmallocEntry
```

### 4.3.3 Enumeration Type Documentation

#### 4.3.3.1 anonymous enum

```
anonymous enum
```

## Enumerator

|                                    |  |
|------------------------------------|--|
| debugmalloc_canary_size            |  |
| debugmalloc_canary_char            |  |
| debugmalloc_tablesize              |  |
| debugmalloc_max_block_size_default |  |

## 4.4 debugmalloc.h

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

```

1 #ifndef DEBUGMALLOC_H
2 #define DEBUGMALLOC_H
3
4 #include <stdbool.h>
5 #include <stddef.h>
6 #include <stdlib.h>
7 #include <stdio.h>
8 #include <ctype.h>
9 #include <string.h>
10 #include <stdarg.h>
11
12
13 enum {
14     /* size of canary in bytes. should be multiple of largest alignment
15      * required by any data type (usually 8 or 16) */
16     debugmalloc_canary_size = 64,
17
18     /* canary byte */
19     debugmalloc_canary_char = 'K',
20
21     /* hash table size for allocated entries */
22     debugmalloc_tablesize = 256,
23
24     /* max block size for allocation, can be modified with debugmalloc_max_block_size() */
25     debugmalloc_max_block_size_default = 1048576
26 };
27
28
29 /* make getpid and putenv "crossplatform". deprecated on windows but they work just fine,
30  * however not declared. */
31 #ifdef _WIN32
32     /* windows */
33     #include <process.h>
34     #ifdef _MSC_VER
35         /* visual studio, getenv/getpid deprecated warning */
36         #pragma warning(disable: 4996)
37     #else
38         /* other windows. the declaration is unfortunately hidden
39          * in mingw header files by ifdefs. */
40         int putenv(const char *);
41     #endif
42 #else
43     /* posix */
44     #include <unistd.h>
45 #endif
46
47
48 /* linked list entry for allocated blocks */
49 typedef struct DebugmallocEntry {
50     void *real_mem;    /* the address of the real allocation */
51     void *user_mem;    /* address shown to the user */
52     size_t size;       /* size of block requested by user */
53
54     char file[64];     /* malloc called in this file */
55     unsigned line;     /* malloc called at this line in file */
56     char func[32];     /* allocation function called (malloc, calloc, realloc) */
57     char expr[128];    /* expression calculating the size of allocation */
58
59     struct DebugmallocEntry *prev, *next; /* for doubly linked list */
60 } DebugmallocEntry;
61
62
63 /* debugmalloc singleton, storing all state */
64 typedef struct DebugmallocData {
65     char logfile[256]; /* log file name or empty string */
66     long max_block_size; /* max size of a single block allocated */

```

```

67     long alloc_count;      /* currently allocated; decreased with free */
68     long long alloc_bytes;
69     long all_alloc_count; /* all allocations, never decreased */
70     long long all_alloc_bytes;
71     DebugmallocEntry head[debugmalloc_tablesize], tail[debugmalloc_tablesize]; /* head and tail elements
of allocation lists */
72 } DebugmallocData;
73
74
75 /* this forward declaration is required by the singleton manager function */
76 static DebugmallocData * debugmalloc_create(void);
77
78
79 /* creates singleton instance. as this function is static included to different
80 * translation units, multiple instances of the static variables are created.
81 * to make sure it is really a singleton, these instances must know each other
82 * somehow. an environment variable is used for that purpose, ie. the address
83 * of the singleton allocated is stored by the operating system.
84 * this implementation is not thread-safe. */
85 static DebugmallocData * debugmalloc_singleton(void) {
86     static char envstr[100];
87     static void *instance = NULL;
88
89     /* if we do not know the address of the singleton:
90     * - maybe we are the one to create it (env variable also does not exist)
91     * - or it is already created, and stored in the env variable. */
92     if (instance == NULL) {
93         char envvarname[100] = "";
94         sprintf(envvarname, "%s%d", "debugmallocsingleton", (int) getpid());
95         char *envptr = getenv(envvarname);
96         if (envptr == NULL) {
97             /* no env variable: create singleton. */
98             instance = debugmalloc_create();
99             sprintf(envstr, "%s=%p", envvarname, instance);
100             putenv(envstr);
101         } else {
102             /* another copy of this function already created it. */
103             int ok = sscanf(envptr, "%p", &instance);
104             if (ok != 1) {
105                 fprintf(stderr, "debugmalloc: nem lehet ertelmezni: %s!\n", envptr);
106                 abort();
107             }
108         }
109     }
110
111     return (DebugmallocData *) instance;
112 }
113
114
115 /* better version of strncpy, always terminates string with \0. */
116 static void debugmalloc_strncpy(char *dest, char const *src, size_t destsize) {
117     strncpy(dest, src, destsize);
118     dest[destsize - 1] = '\0';
119 }
120
121
122 /* set the name of the log file for debugmalloc. empty filename
123 * means logging to stderr. */
124 static void debugmalloc_log_file(char const *logfilename) {
125     if (logfilename == NULL)
126         logfilename = "";
127     DebugmallocData *instance = debugmalloc_singleton();
128     debugmalloc_strncpy(instance->logfile, logfilename, sizeof(instance->logfile));
129 }
130
131
132 /* set the maximum size of one block. useful for debugging purposes. */
133 static void debugmalloc_max_block_size(long max_block_size) {
134     DebugmallocData *instance = debugmalloc_singleton();
135     instance->max_block_size = max_block_size;
136 }
137
138
139
140 /* printf to the log file, or stderr. */
141 static void debugmalloc_log(char const *format, ...) {
142     DebugmallocData *instance = debugmalloc_singleton();
143     FILE *f = stderr;
144     if (instance->logfile[0] != '\0') {
145         f = fopen(instance->logfile, "at");
146         if (f == NULL) {
147             f = stderr;
148             fprintf(stderr, "debugmalloc: nem tudom megnyitni a %s fajlt irasra!\n", instance->logfile);
149             debugmalloc_strncpy(instance->logfile, "", sizeof(instance->logfile));
150         }
151     }
152     vfprintf(f, format, va_arg(0, void *));
153     fflush(f);
154 }

```

```

153     va_list ap;
154     va_start(ap, format);
155     vfprintf(f, format, ap);
156     va_end(ap);
157
158     if (f != stderr)
159         fclose(f);
160 }
161
162
163 /* initialize a memory block allocated for the user. the start and the end
164  * of the block is initialized with the canary characters. if 'zero' is
165  * true, the user memory area is zero-initialized, otherwise it is also
166  * filled with the canary character to simulate garbage in memory. */
167 static void debugmalloc_memory_init(DebugmallocEntry *elem, bool zero) {
168     unsigned char *real_mem = (unsigned char *) elem->real_mem;
169     unsigned char *user_mem = (unsigned char *) elem->user_mem;
170     unsigned char *canary1 = real_mem;
171     unsigned char *canary2 = real_mem + debugmalloc_canary_size + elem->size;
172     memset(canary1, debugmalloc_canary_char, debugmalloc_canary_size);
173     memset(canary2, debugmalloc_canary_char, debugmalloc_canary_size);
174     memset(user_mem, zero ? 0 : debugmalloc_canary_char, elem->size);
175 }
176
177 /* check canary, return true if ok, false if corrupted. */
178 static bool debugmalloc_canary_ok(DebugmallocEntry const *elem) {
179     unsigned char *real_mem = (unsigned char *) elem->real_mem;
180     unsigned char *canary1 = real_mem;
181     unsigned char *canary2 = real_mem + debugmalloc_canary_size + elem->size;
182     for (size_t i = 0; i < debugmalloc_canary_size; ++i) {
183         if (canary1[i] != debugmalloc_canary_char)
184             return false;
185         if (canary2[i] != debugmalloc_canary_char)
186             return false;
187     }
188     return true;
189 }
190
191
192 /* dump memory contents to log file. */
193 static void debugmalloc_dump_memory(char const *mem, size_t size) {
194     for (unsigned y = 0; y < (size + 15) / 16; y++) {
195         char line[80];
196         int pos = 0;
197         pos += sprintf(line + pos, "      %04x ", y * 16);
198         for (unsigned x = 0; x < 16; x++) {
199             if (y * 16 + x < size)
200                 pos += sprintf(line + pos, "%02x ", mem[y * 16 + x]);
201             else
202                 pos += sprintf(line + pos, "   ");
203         }
204         pos += sprintf(line + pos, " ");
205         for (unsigned x = 0; x < 16; x++) {
206             if (y * 16 + x < size) {
207                 unsigned char c = mem[y * 16 + x];
208                 pos += sprintf(line + pos, "%c", isprint(c) ? c : '.');
209             }
210             else {
211                 pos += sprintf(line + pos, " ");
212             }
213         }
214         debugmalloc_log("%s\n", line);
215     }
216 }
217
218
219 /* dump data of allocated memory block.
220  * if the canary is corrupted, it is also written to the log. */
221 static void debugmalloc_dump_elem(DebugmallocEntry const *elem) {
222     bool canary_ok = debugmalloc_canary_ok(elem);
223
224     debugmalloc_log("  %p, %u bajt, kanari: %s\n"
225                    "  %s:%u, %s(%s)\n",
226                    elem->user_mem, (unsigned) elem->size, canary_ok ? "ok" : "**SERULT**",
227                    elem->file, elem->line,
228                    elem->func, elem->expr);
229
230     if (!canary_ok) {
231         debugmalloc_log("      ELOTTE kanari: \n");
232         debugmalloc_dump_memory((char const *) elem->real_mem, debugmalloc_canary_size);
233     }
234
235     debugmalloc_dump_memory((char const *) elem->user_mem, elem->size > 64 ? 64 : elem->size);
236
237     if (!canary_ok) {
238         debugmalloc_log("      UTANA kanari: \n");
239         debugmalloc_dump_memory((char const *) elem->real_mem + debugmalloc_canary_size + elem->size,

```



```

        debugmalloc_canary_size);
    }
}
242
243
244 /* dump data of all memory blocks allocated. */
245 static void debugmalloc_dump(void) {
246     DebugmallocData *instance = debugmalloc_singleton();
247     debugmalloc_log("** DEBUGMALLOC DUMP *****\n");
248     int cnt = 0;
249     for (size_t i = 0; i < debugmalloc_tablesize; i++) {
250         DebugmallocEntry *head = &instance->head[i];
251         for (DebugmallocEntry *iter = head->next; iter->next != NULL; iter = iter->next) {
252             ++cnt;
253             debugmalloc_log("** %d/%d. rekord:\n", cnt, instance->alloc_count);
254             debugmalloc_dump_elem(iter);
255         }
256     }
257     debugmalloc_log("** DEBUGMALLOC DUMP VEGE *****\n");
258 }
259
260
261 /* called at program exit to dump data if there is a leak,
262  * ie. allocated block remained. */
263 static void debugmalloc_atexit_dump(void) {
264     DebugmallocData *instance = debugmalloc_singleton();
265
266     if (instance->alloc_count > 0) {
267         debugmalloc_log("\n"
268             "*****\n"
269             "* MEMORIASZIVARGAS VAN A PROGRAMBAN!!!\n"
270             "*****\n"
271             "\n");
272         debugmalloc_dump();
273     } else {
274         debugmalloc_log("*****\n"
275             "* Debugmalloc: nincs memoriaszivargas a programban.\n"
276             "* Osszes foglalas: %d blokk, %d bajt.\n"
277             "*****\n",
278             instance->all_alloc_count, instance->all_alloc_bytes);
279     }
280 }
281
282
283 /* hash function for bucket hash. */
284 static size_t debugmalloc_hash(void *address) {
285     /* the last few bits are ignored, as they are usually zero for
286      * alignment purposes. all tested architectures used 16 byte allocation. */
287     size_t cut = (size_t)address >> 4;
288     return cut % debugmalloc_tablesize;
289 }
290
291
292 /* insert element to hash table. */
293 static void debugmalloc_insert(DebugmallocEntry *entry) {
294     DebugmallocData *instance = debugmalloc_singleton();
295     size_t idx = debugmalloc_hash(entry->user_mem);
296     DebugmallocEntry *head = &instance->head[idx];
297     entry->prev = head;
298     entry->next = head->next;
299     head->next->prev = entry;
300     head->next = entry;
301     instance->alloc_count += 1;
302     instance->alloc_bytes += entry->size;
303     instance->all_alloc_count += 1;
304     instance->all_alloc_bytes += entry->size;
305 }
306
307
308 /* remove element from hash table */
309 static void debugmalloc_remove(DebugmallocEntry *entry) {
310     DebugmallocData *instance = debugmalloc_singleton();
311     entry->next->prev = entry->prev;
312     entry->prev->next = entry->next;
313     instance->alloc_count -= 1;
314     instance->alloc_bytes -= entry->size;
315 }
316
317
318 /* find element in hash table, given with the memory address that the user sees.
319  * @return the linked list entry, or null if not found. */
320 static DebugmallocEntry *debugmalloc_find(void *mem) {
321     DebugmallocData *instance = debugmalloc_singleton();
322     size_t idx = debugmalloc_hash(mem);
323     DebugmallocEntry *head = &instance->head[idx];
324     for (DebugmallocEntry *iter = head->next; iter->next != NULL; iter = iter->next)
325         if (iter->user_mem == mem)

```

```

326         return iter;
327     return NULL;
328 }
329
330
331 /* allocate memory. this function is called via the macro. */
332 static void *debugmalloc_malloc_full(size_t size, char const *func, char const *expr, char const *file,
    unsigned line, bool zero) {
333     /* imitate standard malloc: return null if size is zero */
334     if (size == 0)
335         return NULL;
336
337     /* check max size */
338     DebugmallocData *instance = debugmalloc_singleton();
339     if (size > instance->max_block_size) {
340         debugmalloc_log("debugmalloc: %s @ %s:%u: a blokk merete tul nagy, %u bajt;
debugmalloc_max_block_size() fuggvennyel novelheto.\n", func, file, line, (unsigned) size);
341         abort();
342     }
343
344     /* allocate more memory, make room for canary */
345     void *real_mem = malloc(size + 2 * debugmalloc_canary_size);
346     if (real_mem == NULL) {
347         debugmalloc_log("debugmalloc: %s @ %s:%u: nem sikerult %u meretu memoriat foglalni!\n", func,
file, line, (unsigned) size);
348         return NULL;
349     }
350
351     /* allocate memory for linked list element */
352     DebugmallocEntry *newentry = (DebugmallocEntry *) malloc(sizeof(DebugmallocEntry));
353     if (newentry == NULL) {
354         free(real_mem);
355         debugmalloc_log("debugmalloc: %s @ %s:%u: le tudtam foglalni %u memoriat, de utana a sajatnak
nem, sry\n", func, file, line, (unsigned) size);
356         abort();
357     }
358
359     /* metadata of allocation: caller function, code line etc. */
360     debugmalloc_strlcpy(newentry->func, func, sizeof(newentry->func));
361     debugmalloc_strlcpy(newentry->expr, expr, sizeof(newentry->expr));
362     debugmalloc_strlcpy(newentry->file, file, sizeof(newentry->file));
363     newentry->line = line;
364
365     /* address of allocated memory chunk */
366     newentry->real_mem = real_mem;
367     newentry->user_mem = (unsigned char *) real_mem + debugmalloc_canary_size;
368     newentry->size = size;
369     debugmalloc_memory_init(newentry, zero);
370
371     /* store in list and return pointer to user area */
372     debugmalloc_insert(newentry);
373     return newentry->user_mem;
374 }
375
376
377 /* free memory and remove list item. before deleting, the chunk is filled with
378  * the canary byte to make sure that the user will see garbage if the memory
379  * is accessed after freeing. */
380 static void debugmalloc_free_inner(DebugmallocEntry *deleted) {
381     debugmalloc_remove(deleted);
382
383     /* fill with garbage, then remove from linked list */
384     memset(deleted->real_mem, debugmalloc_canary_char, deleted->size + 2 * debugmalloc_canary_size);
385     free(deleted->real_mem);
386     free(deleted);
387 }
388
389
390 /* free memory - called via the macro.
391  * as all allocations are tracked in the list, this function can terminate the program
392  * if a block is freed twice or the free function is called with an invalid address. */
393 static void debugmalloc_free_full(void *mem, char const *func, char const *file, unsigned line) {
394     /* imitate standard free function: if ptr is null, no operation is performed */
395     if (mem == NULL)
396         return;
397
398     /* find allocation, abort if not found */
399     DebugmallocEntry *deleted = debugmalloc_find(mem);
400     if (deleted == NULL) {
401         debugmalloc_log("debugmalloc: %s @ %s:%u: olyan teruletet probalsz felszabaditani, ami nincs
lefoglalva!\n", func, file, line);
402         abort();
403     }
404
405     /* check canary and then free memory */
406     if (!debugmalloc_canary_ok(deleted)) {
407         debugmalloc_log("debugmalloc: %s @ %s:%u: a %p memoriateruletet tulindexelted!\n", func, file,

```

```

        line, mem);
408     debugmalloc_dump_elem(deleted);
409 }
410     debugmalloc_free_inner(deleted);
411 }
412
413
414 /* realloc-like function. */
415 static void *debugmalloc_realloc_full(void *oldmem, size_t newsize, char const *func, char const *expr,
        char const *file, unsigned line) {
416     /* imitate standard realloc: equivalent to free if size is null. */
417     if (newsize == 0) {
418         debugmalloc_free_full(oldmem, func, file, line);
419         return NULL;
420     }
421     /* imitate standard realloc: equivalent to malloc if first param is NULL */
422     if (oldmem == NULL)
423         return debugmalloc_malloc_full(newsize, func, expr, file, line, 0);
424
425     /* find old allocation. abort if not found. */
426     DebugmallocEntry *oldentry = debugmalloc_find(oldmem);
427     if (oldentry == NULL) {
428         debugmalloc_log("debugmalloc: %s @ %s:%u: olyan területet próbalsz atmeretezni, ami nincs
        lefoglalva!\n", func, file, line);
429         abort();
430     }
431
432     /* create new allocation, copy & free old data */
433     void *newmem = debugmalloc_malloc_full(newsize, func, expr, file, line, false);
434     if (newmem == NULL) {
435         debugmalloc_log("debugmalloc: %s @ %s:%u: nem sikerult uj memoriat foglalni az
        atmeretezeshez!\n", func, file, line);
436         /* imitate standard realloc: original block is untouched, but return NULL */
437         return NULL;
438     }
439     size_t smaller = oldentry->size < newsize ? oldentry->size : newsize;
440     memcpy(newmem, oldmem, smaller);
441     debugmalloc_free_inner(oldentry);
442
443     return newmem;
444 }
445
446
447 /* initialize debugmalloc singleton. returns the newly allocated instance */
448 static DebugmallocData * debugmalloc_create(void) {
449     /* config check */
450     if (debugmalloc_canary_size % 16 != 0) {
451         debugmalloc_log("debugmalloc: a kanari merete legyen 16-tal oszthato\n");
452         abort();
453     }
454     if (debugmalloc_canary_char == 0) {
455         debugmalloc_log("debugmalloc: a kanari legyen 0-tol kulonbozo\n");
456         abort();
457     }
458     /* avoid compiler warning if these functions are not used */
459     (void) debugmalloc_realloc_full;
460     (void) debugmalloc_log_file;
461     (void) debugmalloc_max_block_size;
462
463     /* create and initialize instance */
464     DebugmallocData *instance = (DebugmallocData *) malloc(sizeof(DebugmallocData));
465     if (instance == NULL) {
466         debugmalloc_log("debugmalloc: nem sikerult elinditani a memoriakezelest\n");
467         abort();
468     }
469     debugmalloc_strncpy(instance->logfile, "", sizeof(instance->logfile));
470     instance->max_block_size = debugmalloc_max_block_size_default;
471     instance->alloc_count = 0;
472     instance->alloc_bytes = 0;
473     instance->all_alloc_count = 0;
474     instance->all_alloc_bytes = 0;
475     for (size_t i = 0; i < debugmalloc_tablesize; i++) {
476         instance->head[i].prev = NULL;
477         instance->head[i].next = &instance->tail[i];
478         instance->tail[i].next = NULL;
479         instance->tail[i].prev = &instance->head[i];
480     }
481
482     atexit(debugmalloc_atexit_dump);
483     return instance;
484 }
485
486
487 /* These macro-like functions forward all allocation/free
488  * calls to debugmalloc. Usage is the same, malloc(size)
489  * gives the address of a new memory block, free(ptr)
490  * deallocates etc.

```

```

491 *
492 * If you use this file, make sure that you include this
493 * in *ALL* translation units (*.c) of your source. The
494 * builtin free() function cannot deallocate a memory block
495 * that was allocated via debugmalloc, yet the name of
496 * the function is the same! */
497
498 #define malloc(S) debugmalloc_malloc_full((S), "malloc", #S, __FILE__, __LINE__, false)
499 #define calloc(N,S) debugmalloc_malloc_full((N)*(S), "calloc", #N, " " #S, __FILE__, __LINE__, true)
500 #define realloc(P,S) debugmalloc_realloc_full((P), (S), "realloc", #S, __FILE__, __LINE__)
501 #define free(P) debugmalloc_free_full((P), "free", __FILE__, __LINE__)
502
503 #endif

```

## 4.5 enemy\_hit\_management.h File Reference

```

#include "game_attributes.h"
#include "game_assets.h"
#include "torpedo.h"
#include "debugmalloc.h"

```

### Functions

- void [explode\\_enemy\\_ship\\_if\\_dead](#) ([EnemyShip](#) \*\**enemy\_ship*, [EnemyShip](#) \*\**temp\_ship*, [GameAttributes](#) \**game\_attributes*)  
*explode\_enemy\_ship\_if\_dead*

### 4.5.1 Function Documentation

#### 4.5.1.1 explode\_enemy\_ship\_if\_dead()

```

void explode_enemy_ship_if_dead (
    EnemyShip ** enemy_ship,
    EnemyShip ** temp_ship,
    GameAttributes * game_attributes )

```

*explode\_enemy\_ship\_if\_dead*

Felszabadítja a felrobbant ellenseges hajót ha annak HP-ja 0 (vagy kevesebb), és kezeli az ahhoz tartozó pontereket.

#### Parameters

|         |                          |  |
|---------|--------------------------|--|
| in, out | ** <i>enemy_ship</i>     | Az ellenseges hajotkat tartalmazó lancolt lista aktualis elemenek pointerere.      |
| in, out | ** <i>temp_ship</i>      | Az ellenseges hajot tartalmazó lancolt lista head pointerenek ideiglenes taroloja. |
| in, out | * <i>game_attributes</i> | A jatek attributumainak taroloja.  |

## Returns

void

## 4.6 enemy\_hit\_management.h

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

```
1 #ifndef ENEMY_HIT_MANAGEMENT_H_INCLUDED
2 #define ENEMY_HIT_MANAGEMENT_H_INCLUDED
3
4 #include "game_attributes.h"
5 #include "game_assets.h"
6 #include "torpedo.h"
7
8 #include "debugmalloc.h"
9
10 void explode_enemy_ship_if_dead(EnemyShip **enemy_ship, EnemyShip **temp_ship, GameAttributes
    *game_attributes);
11
12 #endif // ENEMY_HIT_MANAGEMENT_H_INCLUDED
```

## 4.7 enemy\_hit\_mananagement.c File Reference

```
#include "enemy_hit_management.h"
```

### Functions

- void [explode\\_enemy\\_ship\\_if\\_dead](#) (EnemyShip \*\*enemy\_ship, EnemyShip \*\*temp\_ship, GameAttributes \*game\_attributes)  
*explode\_enemy\_ship\_if\_dead*

### 4.7.1 Function Documentation

#### 4.7.1.1 explode\_enemy\_ship\_if\_dead()

```
void explode_enemy_ship_if_dead (
    EnemyShip ** enemy_ship,
    EnemyShip ** temp_ship,
    GameAttributes * game_attributes )
```

*explode\_enemy\_ship\_if\_dead*

Felszabadítja a felrobbant ellenseges hajót ha annak HP-ja 0 (vagy kevesebb), és kezeli az ahhoz tartozó pont-ereket.

#### Parameters

|         |                          |  |
|---------|--------------------------|--|
| in, out | ** <i>enemy_ship</i>     | Az ellenseges hajókat tartalmazó lancolt lista aktuális elemének pointerre.        |
| in, out | ** <i>temp_ship</i>      | Az ellenseges hajót tartalmazó lancolt lista head pointerének ideiglenes tarolója. |
| in, out | * <i>game_attributes</i> | A játék attribútumainak tarolója.  |

## Returns

void

## 4.8 enemy\_ship.c File Reference

```
#include "enemy_ship.h"
```

### Functions

- [EnemyShip](#) \* [init\\_enemy\\_armada](#) ([TextureData](#) texture\_data, [SpriteMapData](#) sprite\_map\_data, [ShipDTT](#) \*\*ship\_dtt, [GameAttributes](#) \*game\_attributes)  
*init\_enemy\_armada*
- void [move\\_enemy\\_armada](#) ([EnemyShip](#) \*enemy\_armada, [GameAttributes](#) \*game\_attributes)  
*move\_enemy\_armada*
- int [find\\_max\\_enemy\\_armada\\_y\\_coor](#) ([EnemyShip](#) \*enemy\_armada)  
*find\_max\_enemy\_armada\_y\_coor*
- void [pop\\_enemy\\_ship](#) ([EnemyShip](#) \*\*enemy\_ship)  
*pop\_enemy\_ship*
- void [free\\_enemy\\_armada](#) ([EnemyShip](#) \*enemy\_armada)  
*free\_enemy\_armada*

### 4.8.1 Function Documentation

#### 4.8.1.1 find\_max\_enemy\_armada\_y\_coor()

```
int find_max_enemy_armada_y_coor (
    EnemyShip * enemy_armada )
```

find\_max\_enemy\_armada\_y\_coor

Megkeresi a kepernyo also szelehez legkozelebb eso ellenseges hajo koordinatajat.

## Parameters

|         |                       |   |
|---------|-----------------------|---|
| in, out | * <i>enemy_armada</i> | Az ellenseges hajokat tartalmazo lancolt lista. |
|---------|-----------------------|---|

## Returns

[out] max\_y\_coor A kepernyo also szelehez legkozelebb eso ellenseges hajo koordinataja.

#### 4.8.1.2 free\_enemy\_armada()

```
void free_enemy_armada (
    EnemyShip * enemy_armada )
```

free\_enemy\_armada

Az ellenseges hadsereg által elfoglalt memoriaterület felszabadításaért felel.

##### Parameters

|     |              |  |
|-----|--------------|--|
| [ ] | enemy_armada | Az ellenseges hajotkat tartalmazó lancolt lista head pointere. |
|-----|--------------|--|

##### Returns

void

#### 4.8.1.3 init\_enemy\_armada()

```
EnemyShip * init_enemy_armada (
    TextureData texture_data,
    SpriteMapData sprite_map_data,
    ShipDTT ** ship_dtt,
    GameAttributes * game_attributes )
```

init\_enemy\_armada

Az ellenseges hadsereget inicializálja a bemeneti paraméterek alapján.

VIGYAZAT: a hadsereg és az azt alkotó hajók felszabadításaért a hívó felel!

##### Parameters

|    |                 |  |
|----|-----------------|--|
| in | texture_data    | Az ellenseges hajó textúra-koordináta adatait tartalmazó tároló.                     |
| in | sprite_map_data | Az ellenseges hajó spritemap koordináta adatait tartalmazó tároló (az SDL2 igényli). |
| in | **ship_dtt      | Az ellenseges hajó attribútumait tartalmazó fájlbeolvasásból származó adatstruktúra. |
| in | game_attributes | Az összes játékattribútumot tartalmazó adatstruktúra.                                |

##### Returns

[out] enemy\_armada az ellenseges hajók lancolt listájával tér vissza.

#### 4.8.1.4 move\_enemy\_armada()

```
void move_enemy_armada (
    EnemyShip * enemy_armada,
    GameAttributes * game_attributes )
```

move\_enemy\_armada

Az ellenseges hadsereg mozgásat vezérli.

#### Parameters

|         |                        |  |
|---------|------------------------|--|
| in, out | <i>*enemy_armada</i>   | Az ellenseges hajokat tartalmazo lancolt lista.      |
| in      | <i>game_attributes</i> | Az osszes jatekattributumot tartlmazo adatstruktura. |

#### Returns

void

#### 4.8.1.5 pop\_enemy\_ship()

```
void pop_enemy_ship (
    EnemyShip ** enemy_ship )
```

pop\_enemy\_ship

Egy hajo torleset vegzi.

#### Parameters

|         |                   |                                     |
|---------|-------------------|-------------------------------------|
| in, out | <i>enemy_ship</i> | Az ellenseges hadsereg adott hajoja |
|---------|-------------------|-------------------------------------|

#### Returns

void

## 4.9 enemy\_ship.h File Reference

```
#include "game_attributes.h"
#include "data_transfer_types.h"
#include "random_number_in_interval.h"
#include "texture_data.h"
#include "torpedo.h"
#include <stdlib.h>
#include <stdbool.h>
#include <math.h>
#include "debugmalloc.h"
```

### Data Structures

- struct [enemyship](#)  
*EnemyShip*.



## Typedefs

- typedef struct [enemyship](#) [EnemyShip](#)  
*EnemyShip.*

## Functions

- [EnemyShip](#) \* [init\\_enemy\\_armada](#) ([TextureData](#) texture\_data, [SpriteMapData](#) sprite\_map\_data, [ShipDTT](#) \*\*ship\_dtt, [GameAttributes](#) \*game\_attributes)  
*init\_enemy\_armada*
- void [move\\_enemy\\_armada](#) ([EnemyShip](#) \*enemy\_armada, [GameAttributes](#) \*game\_attributes)  
*move\_enemy\_armada*
- int [find\\_max\\_enemy\\_armada\\_y\\_coor](#) ([EnemyShip](#) \*enemy\_armada)  
*find\_max\_enemy\_armada\_y\_coor*
- void [pop\\_enemy\\_ship](#) ([EnemyShip](#) \*\*enemy\_ship)  
*pop\_enemy\_ship*
- void [free\\_enemy\\_armada](#) ([EnemyShip](#) \*enemy\_armada)  
*free\_enemy\_armada*

### 4.9.1 Typedef Documentation

#### 4.9.1.1 EnemyShip

```
typedef struct enemyship EnemyShip
```

*EnemyShip.*

Az ellenseges hajokat tarolo lancolt lista listaeleme

### 4.9.2 Function Documentation

#### 4.9.2.1 find\_max\_enemy\_armada\_y\_coor()

```
int find\_max\_enemy\_armada\_y\_coor (  
    EnemyShip * enemy_armada )
```

*find\_max\_enemy\_armada\_y\_coor*

Megkeresi a kepernyo also szelehez legkozelebb eso ellenseges hajo koordinatajat.

#### Parameters

|                |                      |   |
|----------------|----------------------|---|
| <i>in, out</i> | <i>*enemy_armada</i> | Az ellenseges hajokat tartalmazo lancolt lista. |
|----------------|----------------------|---|

**Returns**

[out] max\_y\_coor A kepernyo also szelehez legkozelebb eso ellenseges hajo koordinataja.

**4.9.2.2 free\_enemy\_armada()**

```
void free_enemy_armada (
    EnemyShip * enemy_armada )
```

free\_enemy\_armada

Az ellenseges hadsereg által elfoglalt memoriaterület felszabadításaert felel.

**Parameters**

|     |  |
|-----|--|
| [ ] | enemy_armada Az ellenseges hajokat tartalmazo lancolt lista head pointere. |
|-----|--|

**Returns**

void

**4.9.2.3 init\_enemy\_armada()**

```
EnemyShip * init_enemy_armada (
    TextureData texture_data,
    SpriteMapData sprite_map_data,
    ShipDTT ** ship_dtt,
    GameAttributes * game_attributes )
```

init\_enemy\_armada

Az ellenseges hadsereget inicializalja a bemeneti parameterek alapján.

VIGYAZAT: a hadsereg es az azt alkoto hajok felszabaditasaert a hivo felel!

**Parameters**

|    |                 |  |
|----|-----------------|--|
| in | texture_data    | Az ellenseges hajo textura-koordinata adatait tartalmazo tarolo.                     |
| in | sprite_map_data | Az ellenseges hajo spritemap koordinata adatait tartalmazo tarolo (az SDL2 igenyli). |
| in | **ship_dtt      | Az ellenseges hajo attributumait tartalmazo fajlbeolvasasbol szarmazo adatstruktura. |
| in | game_attributes | Az osszes jatekattributumot tartlmazo adatstruktura.                                 |

**Returns**

[out] enemy\_armada az ellenseges hajok lancolt listajaval ter vissza.

#### 4.9.2.4 move\_enemy\_armada()

```
void move_enemy_armada (
    EnemyShip * enemy_armada,
    GameAttributes * game_attributes )
```

move\_enemy\_armada

Az ellenseges hadsereg mozgását vezérli.

##### Parameters

|         |                        |   |
|---------|------------------------|---|
| in, out | * <i>enemy_armada</i>  | Az ellenseges hajokat tartalmazó lancolt lista.       |
| in      | <i>game_attributes</i> | Az összes játékattribútumot tartalmazó adatstruktúra. |

##### Returns

void

#### 4.9.2.5 pop\_enemy\_ship()

```
void pop_enemy_ship (
    EnemyShip ** enemy_ship )
```

pop\_enemy\_ship

Egy hajó torleset vegzi.

##### Parameters

|         |                   |                                     |
|---------|-------------------|-------------------------------------|
| in, out | <i>enemy_ship</i> | Az ellenseges hadsereg adott hajója |
|---------|-------------------|-------------------------------------|

##### Returns

void

## 4.10 enemy\_ship.h

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

```
1
2
3 #ifndef ENEMY_SHIP_H_INCLUDED
4 #define ENEMY_SHIP_H_INCLUDED
5
6 #include "game_attributes.h"
7 #include "data_transfer_types.h"
8 #include "random_number_in_interval.h"
9 #include "texture_data.h"
10 #include "torpedo.h"
11
12 #include <stdlib.h>
13 #include <stdbool.h>
```

```

16 #include <math.h>
17
18 #include "debugmalloc.h"
19
24 typedef struct enemyship{
25     int y_coor;
26     int x_coor;
27     int movement_dir;
28     int hitbox_beg_coor;
29     int hitbox_end_coor;
30     int centerline_y_coor;
31     TextureData texture_data;
32     SpriteMapData sprite_map_data;
33     int speed;
34     int health;
35     int score_value;
36     struct enemyship *next_ship;
37     struct enemyship *prev_ship;
38 }EnemyShip;
39
40 EnemyShip *init_enemy_armada(TextureData texture_data,
41                               SpriteMapData sprite_map_data,
42                               ShipDTT **ship_dtt,
43                               GameAttributes *game_attributes);
44
45 void move_enemy_armada(EnemyShip *enemy_armada, GameAttributes *game_attributes);
46
47 int find_max_enemy_armada_y_coor(EnemyShip *enemy_armada);
48
49 void pop_enemy_ship(EnemyShip **enemy_ship);
50
51 void free_enemy_armada(EnemyShip *enemy_armada);
52
53 #endif // ENEMY_SHIP_H_INCLUDED

```

## 4.11 file\_management.c File Reference

```
#include "file_management.h"
```

### Functions

- `ShipDTT ** import_ship_dtt` (char \*filename, int \*num\_of\_rows)  
*import\_ship\_dtt*
- void `read_texture_data` (char \*filename, `TextureData` \*texture\_data, `SpriteMapData` \*sprite\_map\_data)  
*read\_texture\_data*

### 4.11.1 Function Documentation

#### 4.11.1.1 import\_ship\_dtt()

```

ShipDTT ** import_ship_dtt (
    char * filename,
    int * num_of_rows )

```

*import\_ship\_dtt*

Beolvassa az urhajok alapvető attribútumainak listait egy adott forrásfájlból, majd azokból egy ShipDTT tagokból álló dinamikus tömb pointerével tér vissza.

## Parameters

|     |                    |  |
|-----|--------------------|--|
| in  | <i>filename</i>    | A megnyitandó file neve.                                   |
| out | <i>num_of_rows</i> | Az urhajok listájában található sorok számával tér vissza. |

## Returns

enemy\_armada ShipDTT-ket tartalmazó lista pointere.

## 4.11.1.2 read\_texture\_data()

```
void read_texture_data (
    char * filename,
    TextureData * texture_data,
    SpriteMapData * sprite_map_data )
```

read\_texture\_data

Beolvassa az urhajok texturáihoz szükséges adatokat. majd azokból egy ShipDTT tagokból álló dinamikus tömb pointerevel tér vissza.

## Parameters

|     |                      |   |
|-----|----------------------|---|
| in  | <i>filename</i>      | A megnyitandó file neve.                                |
| out | <i>texture_data</i>  | Az urhajok megjelenítéséhez szükséges textúra adatok.   |
| out | <i>SpriteMapData</i> | Az urhajok texturáinak beolvasásához szükséges méretek. |

## Returns

void

## 4.12 file\_management.h File Reference

```
#include "data_transfer_types.h"
#include "texture_data.h"
#include "keymap.h"
#include <stdio.h>
#include "debugmalloc.h"
```

## Functions

- ShipDTT \*\* import\_ship\_dtt (char \*filename, int \*num\_of\_rows)  
import\_ship\_dtt
- void read\_texture\_data (char \*filename, TextureData \*texture\_data, SpriteMapData \*sprite\_map\_data)  
read\_texture\_data

## 4.12.1 Function Documentation

### 4.12.1.1 import\_ship\_dtt()

```
ShipDTT ** import_ship_dtt (
    char * filename,
    int * num_of_rows )
```

import\_ship\_dtt

Beolvassa az urhajok alapvető attribútumainak listait egy adott forrásfájlból, majd azokból egy ShipDTT tagokból álló dinamikus tömb pointerével tér vissza.

#### Parameters

|     |                    |  |
|-----|--------------------|--|
| in  | <i>filename</i>    | A megnyitandó fájl neve.                                   |
| out | <i>num_of_rows</i> | Az urhajok listájában található sorok számával tér vissza. |

#### Returns

enemy\_armada ShipDTT-ket tartalmazó lista pointerével.

### 4.12.1.2 read\_texture\_data()

```
void read_texture_data (
    char * filename,
    TextureData * texture_data,
    SpriteMapData * sprite_map_data )
```

read\_texture\_data

Beolvassa az urhajok textúráihoz szükséges adatokat. majd azokból egy ShipDTT tagokból álló dinamikus tömb pointerével tér vissza.

#### Parameters

|     |                      |   |
|-----|----------------------|---|
| in  | <i>filename</i>      | A megnyitandó fájl neve.                                |
| out | <i>texture_data</i>  | Az urhajok megjelenítéséhez szükséges textúra adatok.   |
| out | <i>SpriteMapData</i> | Az urhajok textúráinak beolvasásához szükséges méretek. |

## Returns

void

## 4.13 file\_management.h

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

```

1 #ifndef FILE_MANAGEMENT_H_INCLUDED
2 #define FILE_MANAGEMENT_H_INCLUDED
3
4 #include "data_transfer_types.h"
5 #include "texture_data.h"
6 #include "keymap.h"
7
8 #include <stdio.h>
9
10 #include "debugmalloc.h"
11
12 ShipDTT **import_ship_dtt(char *filename, int *num_of_rows);
13
14 void read_texture_data(char *filename,
15                       TextureData *texture_data,
16                       SpriteMapData *sprite_map_data);
17
18 #endif // FILE_MANAGEMENT_H_INCLUDED

```

## 4.14 fire\_management.h File Reference

```

#include "game_attributes.h"
#include "game_assets.h"
#include "enemy_ship.h"
#include "debugmalloc.h"

```

### Functions

- void [fire\\_player\\_torpedo](#) ([GameAssets](#) \*game\_assets, [GameAttributes](#) \*game\_attributes)  
*fire\_player\_torpedo*
- void [fire\\_enemy\\_torpedoes](#) (int armada\_size, [GameAssets](#) \*game\_assets)  
*fire\_enemy\_torpedoes*

### 4.14.1 Function Documentation

#### 4.14.1.1 fire\_enemy\_torpedoes()

```

void fire_enemy_torpedoes (
    int armada_size,
    GameAssets * game_assets )

```

[fire\\_enemy\\_torpedoes](#)

Az ellenseges torpedok kiloveseert felel.

**Parameters**

|     |                     |  |
|-----|---------------------|--|
| in  | <i>armada_size</i>  | Az ellenseges hadseregmerete.          |
| out | <i>*game_assets</i> | A jatekhoz szukseges assetek taroloja. |

**Returns**

void

**4.14.1.2 fire\_player\_torpedo()**

```
void fire_player_torpedo (
    GameAssets * game_assets,
    GameAttributes * game_attributes )
```

fire\_player\_torpedo

A jatekos torpedok kiloveseert felel.

**Parameters**

|     |                         |  |
|-----|-------------------------|--|
| out | <i>*game_assets</i>     | A jatekhoz szukseges assetek taroloja. |
| out | <i>*game_attributes</i> | A jatek attributumainak taroloja.      |

**Returns**

void

**4.15 fire\_management.h**[Go to the documentation of this file.](#)

```
1 #ifndef FIRE_MANAGEMENT_H_INCLUDED
2 #define FIRE_MANAGEMENT_H_INCLUDED
3
4 #include "game_attributes.h"
5 #include "game_assets.h"
6 #include "enemy_ship.h"
7
8 #include "debugmalloc.h"
9
10 void fire_player_torpedo(GameAssets *game_assets, GameAttributes *game_attributes);
11
12 void fire_enemy_torpedoes(int armada_size, GameAssets *game_assets);
13
14 #endif // FIRE_MANAGEMENT_H_INCLUDED
```

**4.16 fire\_managemet.c File Reference**

```
#include "fire_management.h"
```



## Functions

- void `fire_player_torpedo` (`GameAssets` \*game\_assets, `GameAttributes` \*game\_attributes)  
*fire\_player\_torpedo*
- void `fire_enemy_torpedoes` (int armada\_size, `GameAssets` \*game\_assets)  
*fire\_enemy\_torpedoes*

### 4.16.1 Function Documentation

#### 4.16.1.1 fire\_enemy\_torpedoes()

```
void fire_enemy_torpedoes (  
    int armada_size,  
    GameAssets * game_assets )
```

*fire\_enemy\_torpedoes*

Az ellenseges torpedok kiloveseert felel.

##### Parameters

|     |                     |  |
|-----|---------------------|--|
| in  | <i>armada_size</i>  | Az ellenseges hadseregmerete.          |
| out | <i>*game_assets</i> | A jatekhoz szukseges assetek taroloja. |

##### Returns

void

#### 4.16.1.2 fire\_player\_torpedo()

```
void fire_player_torpedo (  
    GameAssets * game_assets,  
    GameAttributes * game_attributes )
```

*fire\_player\_torpedo*

A jatekos torpedok kiloveseert felel.

##### Parameters

|     |                         |  |
|-----|-------------------------|--|
| out | <i>*game_assets</i>     | A jatekhoz szukseges assetek taroloja. |
| out | <i>*game_attributes</i> | A jatek attributumainak taroloja.      |

## Returns

void

## 4.17 game\_assets.h File Reference

```
#include "star_map.h"
#include "player_ship.h"
#include "enemy_ship.h"
#include "torpedo.h"
```

## Data Structures

- struct [gameassets](#)  
*GameAssets.*

## Typedefs

- typedef struct [gameassets](#) [GameAssets](#)  
*GameAssets.*

### 4.17.1 Typedef Documentation

#### 4.17.1.1 GameAssets

```
typedef struct gameassets GameAssets
```

*GameAssets.*

## 4.18 game\_assets.h

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

```
1 #ifndef GAME_ASSETS_H_INCLUDED
2 #define GAME_ASSETS_H_INCLUDED
3
4 #include "star_map.h"
5 #include "player_ship.h"
6 #include "enemy_ship.h"
7 #include "torpedo.h"
8
13 typedef struct gameassets{
14     StarMap *star_map;
15     PlayerShip *player_ship;
16     EnemyShip *enemy_armada;
17     TorpedoShot *player_torpedoes;
18     TorpedoShot *enemy_torpedoes;
19 }GameAssets;
20
21 #endif // GAME_ASSETS_H_INCLUDED
```

## 4.19 game\_attributes.h File Reference

```
#include "input_state_interface.h"
#include <SDL.h>
```

### Data Structures

- struct [gameattributes](#)  
*GameAttributes.*

### Typedefs

- typedef struct [gameattributes](#) [GameAttributes](#)  
*GameAttributes.*

#### 4.19.1 Typedef Documentation

##### 4.19.1.1 GameAttributes

```
typedef struct gameattributes GameAttributes
```

*GameAttributes.*

## 4.20 game\_attributes.h

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

```
1 #ifndef GAME_ATTRIBUTES_H_INCLUDED
2 #define GAME_ATTRIBUTES_H_INCLUDED
3
4 #include "input_state_interface.h"
5 #include <SDL.h>
6
11 typedef struct gameattributes{
12     int width;
13     int height;
14     int enemy_armada_size;
15     int num_of_rows;
16     int enemy_ships_per_row;
17     int game_score;
18     InputStateInterface isi;
19     SDL\_TimerID id;
20 }GameAttributes;
21
22 #endif // GAME_ATTRIBUTES_H_INCLUDED
```

## 4.21 game\_engine.c File Reference

```
#include "game_engine.h"
```

## Functions

- void `free_ship_dtt` (`ShipDTT **ship_dtt`, `int num_of_ships`)  
*free\_ship\_dtt*
- void `runtime` ()  
*runtime*
- `UInt32 input_timer` (`UInt32 ms`, `void *param`)  
*input\_timer*

### 4.21.1 Function Documentation

#### 4.21.1.1 `free_ship_dtt()`

```
void free_ship_dtt (
    ShipDTT ** ship_dtt,
    int num_of_ships )
```

`free_ship_dtt`

Hasznalat utan felszabaditja a ShipDTT ideiglenes taroloikat.

##### Parameters

|    |                           |  |
|----|---------------------------|--|
| in | <code>**ship_dtt</code>   | A hajok inicializalasahoz szukseges fajlbol beolvasott adatok ideiglenes taroloja. |
| in | <code>num_of_ships</code> | A torlendo hajok szama.  |

##### Returns

`void`

#### 4.21.1.2 `input_timer()`

```
UInt32 input_timer (
    UInt32 ms,
    void * param )
```

`input_timer`

Az inputok beolvasasanak idoziteseert felel.

##### Parameters

|    |                    |  |
|----|--------------------|--|
| in | <code>ms</code>    |  |
| in | <code>param</code> |  |

**Returns**

Uint32

**4.21.1.3 runtime()**

```
void runtime ( )
```

runtime

Aggregálja az összes játék működéséhez szükséges logikai függvényt. Amikor a játékos kilep a játékból, felszabadít mindent és kilep az SDL2ből.

**Returns**

void

**4.22 game\_engine.h File Reference**

```
#include "input_state_interface.h"
#include "keymap.h"
#include "graphics.h"
#include "ui_input.h"
#include "game_assets.h"
#include "game_attributes.h"
#include "star_map.h"
#include "player_ship.h"
#include "enemy_ship.h"
#include "fire_management.h"
#include "hit_management.h"
#include "data_transfer_types.h"
#include "file_management.h"
#include "random_number_in_interval.h"
#include "texture_data.h"
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <SDL.h>
#include "debugmalloc.h"
```

**Functions**

- void [runtime](#) ()  
*runtime*
- Uint32 [input\\_timer](#) (Uint32 ms, void \*param)  
*input\_timer*

## 4.22.1 Function Documentation

### 4.22.1.1 input\_timer()

```
Uint32 input_timer (
    Uint32 ms,
    void * param )
```

input\_timer

Az inputok beolvasasanak idoziteseert felel.

#### Parameters

|    |              |  |
|----|--------------|--|
| in | <i>ms</i>    |  |
| in | <i>param</i> |  |

#### Returns

Uint32

### 4.22.1.2 runtime()

```
void runtime ( )
```

runtime

Aggregálja az összes játék mukodesehez szukseges logikai fuggvenyt. Amikor a jatekos kilep a jatekbol, felszabadit mindent es kilep az SDL2bol.

#### Returns

void

## 4.23 game\_engine.h

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

```
1 #ifndef GAME_ENGINE_H_INCLUDED
2 #define GAME_ENGINE_H_INCLUDED
3
4 #include "input_state_interface.h"
5 #include "keymap.h"
6 #include "graphics.h"
7 #include "ui_input.h"
8 #include "game_assets.h"
9 #include "game_attributes.h"
10 #include "star_map.h"
11 #include "player_ship.h"
12 #include "enemy_ship.h"
```

```

13 #include "fire_management.h"
14 #include "hit_management.h"
15 #include "data_transfer_types.h"
16 #include "file_management.h"
17 #include "random_number_in_interval.h"
18 #include "texture_data.h"
19
20 #include <stdio.h>
21 #include <stdlib.h>
22 #include <stdbool.h>
23 #include <SDL.h>
24
25 #include "debugmalloc.h"
26
27 void runtime();
28
29 Uint32 input_timer(Uint32 ms, void *param);
30
31 #endif // GAME_ENGINE_H_INCLUDED

```

## 4.24 graphics.c File Reference

```
#include "graphics.h"
```

### Functions

- `SDL_Texture *` [load\\_sdl\\_texture](#) (`char *img_name`)  
*load\_sdl\_texture*
- `TTF_Font *` [open\\_font](#) (`int size`)  
*open\_font*
- `void` [sdl\\_init](#) (`char const *title`, `int width`, `int height`, `SDL_Window **pwindow`, `SDL_Renderer **prenderer`)  
*sdl\_init*
- `void` [create\\_font](#) ()  
*create\_font*
- `void` [create\\_textures](#) (`char *fed`, `char *enemy`)  
*create\_textures*
- `void` [create\\_window](#) (`int width`, `int height`)  
*create\_window*
- `void` [draw\\_background](#) (`StarMap *sm`)  
*draw\_background*
- `void` [draw\\_player\\_ship](#) (`PlayerShip *ps`)  
*draw\_player\_ship*
- `void` [draw\\_enemy\\_ships](#) (`EnemyShip *enemy_armada`)  
*draw\_enemy\_ships*
- `void` [draw\\_torpedo](#) (`TorpedoShot *torpedoes`)  
*draw\_torpedo*
- `void` [draw\\_end\\_screen](#) ()  
*draw\_end\_screen*
- `void` [draw\\_score](#) (`int game_score`)  
*draw\_score*
- `void` [draw\\_LCARS\\_bacground](#) ()  
*draw\_LCARS\_bacground*
- `void` [clear\\_screen](#) ()  
*clear\_screen*
- `void` [render\\_screen](#) ()  
*render\_screen*
- `void` [destroy\\_textures](#) ()  
*destroy\_textures*

## 4.24.1 Function Documentation

### 4.24.1.1 `clear_screen()`

```
void clear_screen ( )
```

`clear_screen`

Torol mindent a jatekablakbol.

#### Returns

`void`

### 4.24.1.2 `create_font()`

```
void create_font ( )
```

`create_font`

Beolvassa a szukseges meretu fontokat.

#### Returns

`void`

### 4.24.1.3 `create_textures()`

```
void create_textures (
    char * fed,
    char * enemy )
```

`create_textures`

felepiti a texturakat

#### Parameters

|    |              |   |
|----|--------------|---|
| in | <i>fed</i>   | A jatekos texturajat tartalmazó fájl neve.  |
| in | <i>enemy</i> | A ellenség texturáját tartalmazó fájl neve. |



**Returns**

void

**4.24.1.4 create\_window()**

```
void create_window (
    int width,
    int height )
```

create\_window

Legeneralja a jatekablakot.

**Parameters**

|    |               |  |
|----|---------------|--|
| in | <i>width</i>  |  |
| in | <i>height</i> |  |

**Returns**

void

**4.24.1.5 destroy\_textures()**

```
void destroy_textures ( )
```

destroy\_textures

Torli a texturakat.

**Returns**

void

**4.24.1.6 draw\_background()**

```
void draw_background (
    StarMap * sm )
```

draw\_background

kirajzolja a hatteret

**Parameters**

|    |            |                           |
|----|------------|---------------------------|
| in | <i>*sm</i> | A csillagterkep pointere. |
|----|------------|---------------------------|

**Returns**

void

**4.24.1.7 draw\_end\_screen()**

```
void draw_end_screen ( )
```

draw\_end\_screen

Kirajzolja a jatek vege kepernyot.

**Returns**

void

**4.24.1.8 draw\_enemy\_ships()**

```
void draw_enemy_ships (
    EnemyShip * enemy_armada )
```

draw\_enemy\_ships

Kirajzolja az ellenseges hajokat.

**Parameters**

|    |                |   |
|----|----------------|---|
| in | <i>*armada</i> | Az ellenseges hajok listajanak head pointere. |
|----|----------------|---|

**Returns**

void

**4.24.1.9 draw\_LCARS\_bacground()**

```
void draw_LCARS_bacground ( )
```

draw\_LCARS\_bacground

Kirajzolja a jatekos pontjainak hatteret.

**Returns**

void

**4.24.1.10 draw\_player\_ship()**

```
void draw_player_ship (
    PlayerShip * ps )
```

draw\_player\_ship

Kirajzolja a jatekos hajot.

**Parameters**

|    |     |                              |
|----|-----|------------------------------|
| in | *ps | A jatekos hajojanak pointer. |
|----|-----|------------------------------|

**Returns**

void

**4.24.1.11 draw\_score()**

```
void draw_score (
    int game_score )
```

draw\_score

Kirajzolja a jatekos pontjait kepernyot.

**Parameters**

|    |            |                    |
|----|------------|--------------------|
| in | game_score | a jatekos pontjai. |
|----|------------|--------------------|

**Returns**

void

**4.24.1.12 draw\_torpedo()**

```
void draw_torpedo (
    TorpedoShot * torpedoes )
```

draw\_torpedo

Kirajzolja a kilott torpedot.

**Parameters**

|    |                   |                                      |
|----|-------------------|--------------------------------------|
| in | <i>*torpedoes</i> | A torpedok listajanak head pointere. |
|----|-------------------|--------------------------------------|

**Returns**

void

**4.24.1.13 load\_sdl\_texture()**

```
SDL_Texture * load_sdl_texture (
    char * img_name )
```

load\_sdl\_texture

Betolti az SDL2 által hasznalt texturakat.

**Parameters**

|    |                 |                            |
|----|-----------------|----------------------------|
| in | <i>img_name</i> | A betoltendo kepfajl neve. |
|----|-----------------|----------------------------|

**Returns**

fontokat A betoltott textura.

**4.24.1.14 open\_font()**

```
TTF_Font * open_font (
    int size )
```

open\_font

Betolti az SDL2 által hasznalt fontokat.

**Parameters**

|    |             |                           |
|----|-------------|---------------------------|
| in | <i>size</i> | A betoltendo font merete. |
|----|-------------|---------------------------|

**Returns**

font A betoltott font.

**4.24.1.15 render\_screen()**

```
void render_screen ( )
```

render\_screen

Rendereli a jatekablakot.

**Returns**

void

**4.24.1.16 sdl\_init()**

```
void sdl_init (
    char const * title,
    int width,
    int height,
    SDL_Window ** pwindow,
    SDL_Renderer ** prenderer )
```

sdl\_init

Inicializálja az SDL2-t.

**Parameters**

|    |                  |  |
|----|------------------|--|
| in | <i>title</i>     |  |
| in | <i>width</i>     |  |
| in | <i>height</i>    |  |
| in | <i>pwindow</i>   |  |
| in | <i>prenderer</i> |  |

**Returns**

void

**4.25 graphics.h File Reference**

```
#include "star_map.h"
#include "player_ship.h"
#include "enemy_ship.h"
#include "input_state_interface.h"
#include "torpedo.h"
#include <SDL.h>
#include <SDL_image.h>
#include <SDL_ttf.h>
#include <SDL2_gfxPrimitives.h>
```

```
#include <math.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>
#include "debugmalloc.h"
```

## Functions

- void [create\\_font](#) ()  
*create\_font*
- void [create\\_textures](#) (char \*fed, char \*enemy)  
*create\_textures*
- void [create\\_window](#) (int width, int height)  
*create\_window*
- void [draw\\_background](#) (StarMap \*sm)  
*draw\_background*
- void [draw\\_player\\_ship](#) (PlayerShip \*ps)  
*draw\_player\_ship*
- void [draw\\_enemy\\_ships](#) (EnemyShip \*enemy\_armada)  
*draw\_enemy\_ships*
- void [draw\\_torpedo](#) (TorpedoShot \*torpedoes)  
*draw\_torpedo*
- void [draw\\_end\\_screen](#) ()  
*draw\_end\_screen*
- void [draw\\_score](#) (int game\_score)  
*draw\_score*
- void [draw\\_LCARS\\_bacground](#) ()  
*draw\_LCARS\_bacground*
- void [clear\\_screen](#) ()  
*clear\_screen*
- void [render\\_screen](#) ()  
*render\_screen*
- void [destroy\\_textures](#) ()  
*destroy\_textures*

### 4.25.1 Function Documentation

#### 4.25.1.1 clear\_screen()

```
void clear_screen ( )
```

`clear_screen`

Torol mindent a jatekablakbol.

#### Returns

void

#### 4.25.1.2 create\_font()

```
void create_font ( )
```

create\_font

Beolvassa a szukseges meretu fontokat.

##### Returns

void

#### 4.25.1.3 create\_textures()

```
void create_textures (
    char * fed,
    char * enemy )
```

create\_textures

felepiti a texturakat

##### Parameters

|    |              |   |
|----|--------------|---|
| in | <i>fed</i>   | A jatekos texturajat tartalmazó fájl neve.  |
| in | <i>enemy</i> | A ellenség texturáját tartalmazó fájl neve. |

##### Returns

void

#### 4.25.1.4 create\_window()

```
void create_window (
    int width,
    int height )
```

create\_window

Legeneralja a jatekablakot.

##### Parameters

|    |               |  |
|----|---------------|--|
| in | <i>width</i>  |  |
| in | <i>height</i> |  |

**Returns**

void

**4.25.1.5 destroy\_textures()**

```
void destroy_textures ( )
```

destroy\_textures

Torli a texturakat.

**Returns**

void

**4.25.1.6 draw\_background()**

```
void draw_background (
    StarMap * sm )
```

draw\_background

kirajzolja a hatteret

**Parameters**

|    |             |                            |
|----|-------------|----------------------------|
| in | * <i>sm</i> | A csillagterkep pointerre. |
|----|-------------|----------------------------|

**Returns**

void

**4.25.1.7 draw\_end\_screen()**

```
void draw_end_screen ( )
```

draw\_end\_screen

Kirajzolja a jatek vege kepernyot.

**Returns**

void



#### 4.25.1.8 draw\_enemy\_ships()

```
void draw_enemy_ships (
    EnemyShip * enemy_armada )
```

draw\_enemy\_ships

Kirajzolja az ellenseges hajokat.

##### Parameters

|    |         |   |
|----|---------|---|
| in | *armada | Az ellenseges hajok listajanak head pointerere. |
|----|---------|---|

##### Returns

void

#### 4.25.1.9 draw\_LCARS\_bacground()

```
void draw_LCARS_bacground ( )
```

draw\_LCARS\_bacground

Kirajzolja a jatekos pontjainak hatteret.

##### Returns

void

#### 4.25.1.10 draw\_player\_ship()

```
void draw_player_ship (
    PlayerShip * ps )
```

draw\_player\_ship

Kirajzolja a jatekos hajot.

##### Parameters

|    |     |                                 |
|----|-----|---------------------------------|
| in | *ps | A jatekos hajojanak pointerere. |
|----|-----|---------------------------------|

##### Returns

void

#### 4.25.1.11 draw\_score()

```
void draw_score (
    int game_score )
```

draw\_score

Kirajzolja a jatekos pontjait kepernyot.

##### Parameters

|    |                   |                    |
|----|-------------------|--------------------|
| in | <i>game_score</i> | a jatekos pontjai. |
|----|-------------------|--------------------|

##### Returns

void

#### 4.25.1.12 draw\_torpedo()

```
void draw_torpedo (
    TorpedoShot * torpedoes )
```

draw\_torpedo

Kirajzolja a kilott torpedot.

##### Parameters

|    |                   |                                      |
|----|-------------------|--------------------------------------|
| in | <i>*torpedoes</i> | A torpedok listajanak head pointere. |
|----|-------------------|--------------------------------------|

##### Returns

void

#### 4.25.1.13 render\_screen()

```
void render_screen ( )
```

render\_screen

Rendereli a jatekablakot.

##### Returns

void

## 4.26 graphics.h

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

```
1 #ifndef GRAPHICS_H_INCLUDED
2 #define GRAPHICS_H_INCLUDED
3
4 #include "star_map.h"
5 #include "player_ship.h"
6 #include "enemy_ship.h"
7 #include "input_state_interface.h"
8 #include "torpedo.h"
9
10 #include <SDL.h>
11 #include <SDL_image.h>
12 #include <SDL_ttf.h>
13 #include <SDL2_gfxPrimitives.h>
14 #include <math.h>
15 #include <stdlib.h>
16 #include <stdbool.h>
17 #include <string.h>
18
19 #include "debugmalloc.h"
20
21 void create_font();
22
23 void create_textures(char* fed, char* enemy);
24
25 void create_window(int width, int height);
26
27 void draw_background(StarMap *sm);
28
29 void draw_player_ship(PlayerShip *ps);
30
31 void draw_enemy_ships(EnemyShip *enemy_armada);
32
33 void draw_torpedo(TorpedoShot *torpedoes);
34
35 void draw_end_screen();
36
37 void draw_score(int game_score);
38
39 void draw_LCARS_background();
40
41 void clear_screen();
42
43 void render_screen();
44
45 void destroy_textures();
46
47 #endif // GRAPHICS_H_INCLUDED
```

## 4.27 hit\_management.c File Reference

```
#include "hit_management.h"
```

### Functions

- void [remove\\_torpedo\\_if\\_out\\_of\\_bounds](#) ([TorpedoShot](#) \*\*torpedo, [GameAttributes](#) \*game\_attributes)  
-----Externally callable hit managers-----
- void [manage\\_enemy\\_hits](#) ([EnemyShip](#) \*\*enemy\_armada, [TorpedoShot](#) \*\*player\_torpedoes, [GameAssets](#) \*\*game\_assets, [GameAttributes](#) \*game\_attributes)  
*manage\_enemy\_hits*
- void [manage\\_player\\_hits](#) ([PlayerShip](#) \*\*player\_ship, [TorpedoShot](#) \*\*enemy\_torpedoes, [GameAssets](#) \*\*game\_assets, [GameAttributes](#) \*game\_attributes)  
*manage\_player\_hits*

## 4.27.1 Function Documentation

### 4.27.1.1 manage\_enemy\_hits()

```
void manage_enemy_hits (
    EnemyShip ** enemy_armada,
    TorpedoShot ** player_torpedoes,
    GameAssets ** game_assets,
    GameAttributes * game_attributes )
```

manage\_enemy\_hits

Menedzseli a jatekos által bevitt torpedó találatokat.

#### Parameters

|         |                         |  |
|---------|-------------------------|--|
| in, out | <i>enemy_armada</i>     | Az összes ellenseges hajót tartalmazó lancolt lista head pointerre.        |
| in, out | <i>player_torpedoes</i> | A jatekos által kilőtt torpedókat tartalmazó lancolt lista head pointerre. |

#### Returns

void

### 4.27.1.2 manage\_player\_hits()

```
void manage_player_hits (
    PlayerShip ** player_ship,
    TorpedoShot ** enemy_torpedoes,
    GameAssets ** game_assets,
    GameAttributes * game_attributes )
```

manage\_player\_hits

Menedzseli a jatekos által bevitt torpedó találatokat.

#### Parameters

|         |                         |  |
|---------|-------------------------|--|
| in, out | <i>enemy_armada</i>     | Az összes ellenseges hajót tartalmazó lancolt lista head pointerre.        |
| in, out | <i>player_torpedoes</i> | A jatekos által kilőtt torpedókat tartalmazó lancolt lista head pointerre. |

#### Returns

void

### 4.27.1.3 remove\_torpedo\_if\_out\_of\_bounds()

```
void remove_torpedo_if_out_of_bounds (
    TorpedoShot ** torpedo,
    GameAttributes * game_attributes )
```

-----Externally callable hit managers-----

remove\_torpedo\_if\_out\_of\_bounds

Kitorli a torpedot, ha az elhagyta a jatekteret.

#### Parameters

|         |                         |   |
|---------|-------------------------|---|
| in, out | <i>player_torpedoes</i> | A jatekos által kilott torpedokat tartalmazo lancolt lista head pointere. |
| in      | <i>game_attributes</i>  | A jatek attributumait tartalmazo adatszerkezet pointere.                  |

#### Returns

void

## 4.28 hit\_management.h File Reference

```
#include "game_attributes.h"
#include "game_assets.h"
#include "torpedo.h"
#include "enemy_ship.h"
#include "torpedo_hit_management.h"
#include "enemy_hit_management.h"
#include "player_hit_management.h"
#include <stdbool.h>
#include "debugmalloc.h"
```

### Functions

- void [remove\\_torpedo\\_if\\_out\\_of\\_bounds](#) ([TorpedoShot](#) \*\*torpedo, [GameAttributes](#) \*game\_attributes)  
-----Externally callable hit managers-----
- void [manage\\_enemy\\_hits](#) ([EnemyShip](#) \*\*enemy\_armada, [TorpedoShot](#) \*\*player\_torpedo, [GameAssets](#) \*\*game\_assets, [GameAttributes](#) \*game\_attributes)  
*manage\_enemy\_hits*

### 4.28.1 Function Documentation

#### 4.28.1.1 manage\_enemy\_hits()

```
void manage_enemy_hits (
    EnemyShip ** enemy_armada,
    TorpedoShot ** player_torpedoes,
    GameAssets ** game_assets,
    GameAttributes * game_attributes )
```

manage\_enemy\_hits

Menedzseli a jatekos által bevitt torpedó találatokat.

##### Parameters

|         |                         |   |
|---------|-------------------------|---|
| in, out | <i>enemy_armada</i>     | Az összes ellenseges hajót tartalmazó lancolt lista head pointere.        |
| in, out | <i>player_torpedoes</i> | A jatekos által kilőtt torpedókat tartalmazó lancolt lista head pointere. |

##### Returns

void

#### 4.28.1.2 remove\_torpedo\_if\_out\_of\_bounds()

```
void remove_torpedo_if_out_of_bounds (
    TorpedoShot ** torpedo,
    GameAttributes * game_attributes )
```

-----Externally callable hit managers-----

remove\_torpedo\_if\_out\_of\_bounds

Kitorli a torpedót, ha az elhagyta a jatekteret.

##### Parameters

|         |                         |   |
|---------|-------------------------|---|
| in, out | <i>player_torpedoes</i> | A jatekos által kilőtt torpedókat tartalmazó lancolt lista head pointere. |
| in      | <i>game_attributes</i>  | A jatek attributumait tartalmazó adatszerkezet pointere.                  |

##### Returns

void

## 4.29 hit\_management.h

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

```
1 #ifndef HIT_MANAGEMENT_H_INCLUDED
2 #define HIT_MANAGEMENT_H_INCLUDED
3
```

```

4 #include "game_attributes.h"
5 #include "game_assets.h"
6 #include "torpedo.h"
7 #include "enemy_ship.h"
8 #include "torpedo_hit_management.h"
9 #include "enemy_hit_management.h"
10 #include "player_hit_management.h"
11
12 #include <stdbool.h>
13
14 #include "debugmalloc.h"
15
16 void remove_torpedo_if_out_of_bounds(TorpedoShot **torpedo, GameAttributes *game_attributes);
17
18 void manage_enemy_hits(EnemyShip **enemy_armada, TorpedoShot **player_torpedo, GameAssets **game_assets,
19                        GameAttributes *game_attributes);
20 #endif // HIT_MANAGEMENT_H_INCLUDED

```

## 4.30 input\_state\_interface.h File Reference

```
#include <stdbool.h>
```

### Data Structures

- struct [inputstateinterface](#)  
*InputStateInterface.*

### Typedefs

- typedef struct [inputstateinterface](#) [InputStateInterface](#)  
*InputStateInterface.*

#### 4.30.1 Typedef Documentation

##### 4.30.1.1 InputStateInterface

```
typedef struct inputstateinterface InputStateInterface
```

*InputStateInterface.*

## 4.31 input\_state\_interface.h

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

```

1 #ifndef INPUT_STATE_INTERFACE_H_INCLUDED
2 #define INPUT_STATE_INTERFACE_H_INCLUDED
3
4 #include <stdbool.h>
5
10 typedef struct inputstateinterface{
11     bool quit;
12     bool restart;
13     bool game_over;
14     bool up;
15     bool down;
16     bool left;
17     bool right;
18     bool y;
19     bool n;
20     bool torpedo;
21     bool torpedo_ready;
22 } InputStateInterface;
23
24 #endif // INPUT_STATE_INTERFACE_H_INCLUDED

```

## 4.32 keymap.h File Reference

### Data Structures

- struct [keymap](#)  
*KeyMap.*

### Typedefs

- typedef struct [keymap](#) [KeyMap](#)  
*KeyMap.*

#### 4.32.1 Typedef Documentation

##### 4.32.1.1 KeyMap

```
typedef struct keymap KeyMap
```

*KeyMap.*

## 4.33 keymap.h

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

```
1 #ifndef KEYMAP_H_INCLUDED
2 #define KEYMAP_H_INCLUDED
3
4 typedef struct keymap{
5     char *upkey;
6     char *downkey;
7     char *leftkey;
8     char *rightkey;
9     char *torpedokey;
10 } KeyMap;
11
12 #endif // KEYMAP_H_INCLUDED
```

## 4.34 main.c File Reference

```
#include <stdlib.h>
#include "game_engine.h"
#include "debugmalloc.h"
```

### Functions

- int [main](#) (int argc, char \*argv[ ])



### 4.34.1 Function Documentation

#### 4.34.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

## 4.35 player\_hit\_management.c File Reference

```
#include "player_hit_management.h"
```

### Functions

- void [explode\\_player\\_ship\\_if\\_dead](#) ([PlayerShip](#) \*\*player\_ship, [GameAssets](#) \*\*game\_assets, [GameAttributes](#) \*game\_attributes)  
*explode\_player\_ship\_if\_dead*

### 4.35.1 Function Documentation

#### 4.35.1.1 explode\_player\_ship\_if\_dead()

```
void explode_player_ship_if_dead (
    PlayerShip ** player_ship,
    GameAssets ** game_assets,
    GameAttributes * game_attributes )
```

[explode\\_player\\_ship\\_if\\_dead](#)

Felszabadítja a játékos hajóját, ha annak HP-ja 0 (vagy kevesebb), és kezeli az ahhoz tartozó pointereket.

#### Parameters

|         |                          |   |
|---------|--------------------------|---|
| in, out | ** <i>player_ship</i>    | A játékos hajóját tartalmazó adatstruktúra pointer. |
| in, out | ** <i>game_assets</i>    | A játékhoz szükséges assetek tárolója.              |
| in, out | * <i>game_attributes</i> | A játék attribútumainak tárolója.                   |

#### Returns

void

## 4.36 player\_hit\_management.h File Reference

```
#include "game_assets.h"
#include "player_ship.h"
#include "debugmalloc.h"
```

### Functions

- void [explode\\_player\\_ship\\_if\\_dead](#) ([PlayerShip](#) \*\*player\_ship, [GameAssets](#) \*\*game\_assets, [GameAttributes](#) \*game\_attributes)  
*explode\_player\_ship\_if\_dead*

### 4.36.1 Function Documentation

#### 4.36.1.1 explode\_player\_ship\_if\_dead()

```
void explode_player_ship_if_dead (
    PlayerShip ** player_ship,
    GameAssets ** game_assets,
    GameAttributes * game_attributes )
```

[explode\\_player\\_ship\\_if\\_dead](#)

Felszabadítja a játékos hajókat, ha annak HP-ja 0 (vagy kevesebb), és kezeli az ahhoz tartozó pointereket.

#### Parameters

|         |                          |   |
|---------|--------------------------|---|
| in, out | ** <i>player_ship</i>    | A játékos hajókat tartalmazó adatstruktúra pointerre. |
| in, out | ** <i>game_assets</i>    | A játékhoz szükséges assetek tárolója.                |
| in, out | * <i>game_attributes</i> | A játék attribútumainak tárolója.                     |

#### Returns

void

## 4.37 player\_hit\_management.h

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

```
1 #ifndef PLAYER_HIT_MANAGEMENT_H_INCLUDED
2 #define PLAYER_HIT_MANAGEMENT_H_INCLUDED
3
4 #include "game_assets.h"
5 #include "player_ship.h"
6
7 #include "debugmalloc.h"
8
9 void explode_player_ship_if_dead(PlayerShip **player_ship,
10                                GameAssets **game_assets,
11                                GameAttributes *game_attributes);
12
13 #endif // PLAYER_HIT_MANAGEMENT_H_INCLUDED
```

## 4.38 player\_ship.c File Reference

```
#include "player_ship.h"
```

### Functions

- `PlayerShip * init_player_ship (GameAttributes *game_attributes, ShipDTT *ship_dtt, TextureData texture_data, SpriteMapData sprite_map_data)`  
*init\_player\_ship*
- `void move_player_ship (PlayerShip *ps, InputStateInterface *isi, int width, int height)`  
*move\_player\_ship*
- `void free_player_ship (PlayerShip *ps)`  
*move\_player\_ship*

### 4.38.1 Function Documentation

#### 4.38.1.1 free\_player\_ship()

```
void free_player_ship (
    PlayerShip * ps )
```

*move\_player\_ship*

Felszabadítja a játékos hajókat.

#### Parameters

|    |     |                                |
|----|-----|--------------------------------|
| in | *ps | A játékos hajójának pointerre. |
|----|-----|--------------------------------|

#### Returns

void

#### 4.38.1.2 init\_player\_ship()

```
PlayerShip * init_player_ship (
    GameAttributes * game_attributes,
    ShipDTT * ship_dtt,
    TextureData texture_data,
    SpriteMapData sprite_map_data )
```

*init\_player\_ship*

Inicializálja a játékos hajókat.

**Parameters**

|    |                         |  |
|----|-------------------------|--|
| in | <i>*game_attributes</i> | A jatek attributumainak taroloja.            |
| in | <i>*ship_dtt</i>        | A hajo adatainak ideiglenes taroloja.        |
| in | <i>texture_data</i>     | A textura megjelenitesehez szukseges adatok. |
| in | <i>sprite_map_data</i>  | A spritemap beolvasasahoz szukseges adatok.  |

**Returns**

\*ps A jatekos hajojanak pointere.

**4.38.1.3 move\_player\_ship()**

```
void move_player_ship (
    PlayerShip * ps,
    InputStateInterface * isi,
    int width,
    int height )
```

move\_player\_ship

A jatekos hajojanak mozgasaert felelos szamitasokat vegzi.

**Parameters**

|         |               |                                      |
|---------|---------------|--------------------------------------|
| in, out | <i>*ps</i>    | A jatekos hajojanak pointere.        |
| in      | <i>*isi</i>   | A jatek belso allapotainak taroloja. |
| in      | <i>width</i>  | Jatekablak szelessege.               |
| in      | <i>height</i> | Jatekablak magassaga.                |

**Returns**

void

**4.39 player\_ship.h File Reference**

```
#include "game_attributes.h"
#include "input_state_interface.h"
#include "data_transfer_types.h"
#include "texture_data.h"
#include "player_ship.h"
#include <stdbool.h>
#include <stdio.h>
#include "debugmalloc.h"
```

## Data Structures

- struct [playership](#)  
*PlayerShip.*

## Typedefs

- typedef struct [playership](#) [PlayerShip](#)  
*PlayerShip.*

## Functions

- [PlayerShip](#) \* [init\\_player\\_ship](#) ([GameAttributes](#) \*game\_attributes, [ShipDTT](#) \*ship\_dtt, [TextureData](#) texture\_data, [SpriteMapData](#) sprite\_map\_data)  
*init\_player\_ship*
- void [move\\_player\\_ship](#) ([PlayerShip](#) \*ps, [InputStateInterface](#) \*isi, int width, int height)  
*move\_player\_ship*
- void [free\\_player\\_ship](#) ([PlayerShip](#) \*ps)  
*move\_player\_ship*

### 4.39.1 Typedef Documentation

#### 4.39.1.1 PlayerShip

```
typedef struct playership PlayerShip
```

[PlayerShip](#).

A jatekos hajokat tarolo adatstruktura.

### 4.39.2 Function Documentation

#### 4.39.2.1 free\_player\_ship()

```
void free_player_ship (  
    PlayerShip * ps )
```

*move\_player\_ship*

Felszabadítja a jatekos hajokat.

## Parameters

|    |            |                               |
|----|------------|-------------------------------|
| in | <i>*ps</i> | A jatekos hajojanak pointere. |
|----|------------|-------------------------------|

## Returns

void

## 4.39.2.2 init\_player\_ship()

```
PlayerShip * init_player_ship (
    GameAttributes * game_attributes,
    ShipDTT * ship_dtt,
    TextureData texture_data,
    SpriteMapData sprite_map_data )
```

init\_player\_ship

Inicializálja a jatekos hajojat.

## Parameters

|    |                         |  |
|----|-------------------------|--|
| in | <i>*game_attributes</i> | A jatek attributumainak taroloja.            |
| in | <i>*ship_dtt</i>        | A hajo adatainak ideiglenes taroloja.        |
| in | <i>texture_data</i>     | A textura megjelenitesehez szukseges adatok. |
| in | <i>sprite_map_data</i>  | A spritemap beolvasasahoz szukseges adatok.  |

## Returns

*\*ps* A jatekos hajojanak pointere.

## 4.39.2.3 move\_player\_ship()

```
void move_player_ship (
    PlayerShip * ps,
    InputStateInterface * isi,
    int width,
    int height )
```

move\_player\_ship

A jatekos hajojanak mozgasaert felelos szamitasokat vegzi.

## Parameters

|         |               |                                      |
|---------|---------------|--------------------------------------|
| in, out | <i>*ps</i>    | A jatekos hajojanak pointere.        |
| in      | <i>*isi</i>   | A jatek belso allapotainak taroloja. |
| in      | <i>width</i>  | Jatekablak szelessege.               |
| in      | <i>height</i> | Jatekablak magassaga.                |

## Returns

void

## 4.40 player\_ship.h

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

```

1
2
3
4
5 #ifndef PLAYER_SHIP_H_INCLUDED
6 #define PLAYER_SHIP_H_INCLUDED
7
8 #include "game_attributes.h"
9 #include "input_state_interface.h"
10 #include "data_transfer_types.h"
11 #include "texture_data.h"
12 #include "player_ship.h"
13
14 #include <stdbool.h>
15 #include <stdio.h>
16
17 #include "debugmalloc.h"
18
19
20
21
22
23 typedef struct playership{
24     int y_coor;
25     int x_coor;
26     int hitbox_beg_coor;
27     int hitbox_end_coor;
28     int centerline_y_coor;
29     TextureData texture_data;
30     SpriteMapData sprite_map_data;
31     int speed;
32     int health;
33     int score_value;
34 }PlayerShip;
35
36 PlayerShip *init_player_ship(GameAttributes *game_attributes,
37                               ShipDTT *ship_dtt,
38                               TextureData texture_data,
39                               SpriteMapData sprite_map_data);
40
41 void move_player_ship(PlayerShip *ps,
42                       InputStateInterface *isi,
43                       int width,
44                       int height);
45
46 void free_player_ship(PlayerShip *ps);
47
48 #endif // PLAYER_SHIP_H_INCLUDED

```

## 4.41 random\_number\_in\_interval.c File Reference

```
#include "random_number_in_interval.h"
```

### Functions

- int [random\\_number\\_in\\_range](#) (int lower, int upper)  
*random\_number\_in\_range*

#### 4.41.1 Function Documentation

#### 4.41.1.1 random\_number\_in\_range()

```
int random_number_in_range (
    int lower,
    int upper )
```

random\_number\_in\_range

Egy random számmal ter vissza egy meghatározott intervallumon belül. Csak ez a modul hívhatja.

##### Parameters

|    |              |                              |
|----|--------------|------------------------------|
| in | <i>Lower</i> | az intervallum also hatara.  |
| in | <i>Upper</i> | az intervallum felso hatara. |

##### Returns

int A generalt random szam.

## 4.42 random\_number\_in\_interval.h File Reference

```
#include "debugmalloc.h"
```

### Functions

- int [random\\_number\\_in\\_range](#) (int lower, int upper)  
*random\_number\_in\_range*

#### 4.42.1 Function Documentation

##### 4.42.1.1 random\_number\_in\_range()

```
int random_number_in_range (
    int lower,
    int upper )
```

random\_number\_in\_range

Egy random számmal ter vissza egy meghatározott intervallumon belül. Csak ez a modul hívhatja.

##### Parameters

|    |              |                              |
|----|--------------|------------------------------|
| in | <i>Lower</i> | az intervallum also hatara.  |
| in | <i>Upper</i> | az intervallum felso hatara. |



### Returns

int A generalt random szam.

## 4.43 random\_number\_in\_interval.h

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

```
1 #ifndef RANDOM_NUMBER_IN_INTERVAL_H_INCLUDED
2 #define RANDOM_NUMBER_IN_INTERVAL_H_INCLUDED
3
4 #include "debugmalloc.h"
5
6 int random_number_in_range(int lower, int upper);
7
8 #endif // RANDOM_NUMBER_IN_INTERVAL_H_INCLUDED
```

## 4.44 star\_map.c File Reference

```
#include "star_map.h"
```

### Functions

- [StarMap](#) \* [starmap\\_init](#) (int width, int height)  
*starmap\_init*
- void [advance\\_starmap\\_frame](#) ([StarMap](#) \*sm, int width, int height)  
*advance\_starmap\_frame*
- void [free\\_starmap](#) ([StarMap](#) \*sm)  
*free\_starmap*

### 4.44.1 Function Documentation

#### 4.44.1.1 advance\_starmap\_frame()

```
void advance_starmap_frame (  
    StarMap * sm,  
    int width,  
    int height )
```

*advance\_starmap\_frame*

A csillagterkepet eloremozditja egy kockaval. Vegigmegy a csillagokat tartalmazo dinamikus listan, es mindnek egyel noveli az y koordinatajat, amennyiben az nem 10-el nagyobb az ablak magassaganal. Amennyiben ennél az értéknél magasabb az adott csillag y értéke, úgy az y koordinatát 0-ra, az x koordinatát pedig egy, a képernyő szélességében található random értékre állítja.

**Parameters**

|     |               |  |
|-----|---------------|--|
| out | <i>*sm</i>    | Egy StarMap típusú pointer, a jatek StarMap típusában tarolt csillagok koordinatait tarolja. |
| in  | <i>width</i>  | A képernyő szélessége. Erre a random szám generálásához van szükség.                         |
| in  | <i>height</i> | A képernyő magassága. Erre a csillag y koordinatájának ellenőrzéséhez van szükség.           |

**Returns**

void

**4.44.1.2 free\_starmap()**

```
void free_starmap (
    StarMap * sm )
```

free\_starmap

Ez a függvény a parameterként kapott csillagterkep csillagainak listáját, majd magát a csillagterkepet szabadítja fel.

**Parameters**

|    |            |  |
|----|------------|--|
| in | <i>*sm</i> | A felszabadítando csillagterkep pointer. |
|----|------------|--|

**Returns**

void

**4.44.1.3 starmap\_init()**

```
StarMap * starmap_init (
    int width,
    int height )
```

starmap\_init

Ez a függvény inicializálja a StarMap csillagterkep listáját. létrehoz egy, a csillagok vart számának megfelelő hosszúságú dinamikus tömböt, majd abban elhelyezi a sorban generált csillagokat. Visszateresi értéke egy csillagterkep.

**Parameters**

|    |               |  |
|----|---------------|--|
| in | <i>width</i>  | A képernyő szélessége. Erre a csillagok x koordinatájának generálásához van szükség. |
| in | <i>height</i> | A képernyő magassága. Erre a csillagok y koordinatájának generálásához van szükség.  |

## Returns

StarMap

## 4.45 star\_map.h File Reference

```
#include "random_number_in_interval.h"
#include "debugmalloc.h"
```

### Data Structures

- struct [starcolor](#)  
*StarColor.*
- struct [star](#)  
*Star.*
- struct [starmap](#)  
*StarMap.*

### Typedefs

- typedef struct [starcolor](#) [StarColor](#)  
*StarColor.*
- typedef struct [star](#) [Star](#)  
*Star.*
- typedef struct [starmap](#) [StarMap](#)  
*StarMap.*

### Functions

- [StarMap](#) \* [starmap\\_init](#) (int width, int height)  
*starmap\_init*
- void [advance\\_starmap\\_frame](#) ([StarMap](#) \*sm, int width, int height)  
*advance\_starmap\_frame*
- void [free\\_starmap](#) ([StarMap](#) \*sm)  
*free\_starmap*

#### 4.45.1 Typedef Documentation

##### 4.45.1.1 Star

```
typedef struct star Star
```

*Star.*

Ez az adatstruktúra tartja a hatter egy csillagát. Értékei a csillag kirajzolásához szükséges koordináták és sugár. Ez az adattartó a függvényhíváskor megadando parameterlistak leroviditeset, illetve az osszetartozo adatok egy helyen tartasat szolgaltja.

#### 4.45.1.2 StarColor

```
typedef struct starcolor StarColor
```

StarColor.

Ez az adatstruktúra tárolja a háttér csillagának színet. Értékei a csillag RGBA-ban meghatározott színértékei. Ez az adatterelő a függvényhíváskor megadando parameterlistak leroviditeset szolgálja.

#### 4.45.1.3 StarMap

```
typedef struct starmap StarMap
```

StarMap.

Ez az adatstruktúra tárolja a háttér összes csillagát. Értékei a lista hossza, a csillagokat tartalmazó lista, illetve azok színe. Ez az adattároló a háttér csillagainak könnyű létrehozását, tárolását és felszabadítást szolgálja.

### 4.45.2 Function Documentation

#### 4.45.2.1 advance\_starmap\_frame()

```
void advance_starmap_frame (
    StarMap * sm,
    int width,
    int height )
```

advance\_starmap\_frame

A csillagterképet előremozdítja egy kockával. Vegigmegy a csillagokat tartalmazó dinamikus listán, és mindnek egyet növeli az y koordinátáját, amennyiben az nem 10-el nagyobb az ablak magasságánál. Amennyiben ennél az értéknél magasabb az adott csillag y értéke, úgy az y koordinátát 0-ra, az x koordinátát pedig egy, a képernyő szélességében található random értékre állítja.

##### Parameters

|     |               |  |
|-----|---------------|--|
| out | <i>*sm</i>    | Egy StarMap típusú pointer, a játék StarMap típusában tárolt csillagok koordinátáit tárolja. |
| in  | <i>width</i>  | A képernyő szélessége. Erre a random szám generálásához van szükség.                         |
| in  | <i>height</i> | A képernyő magassága. Erre a csillag y koordinátájának ellenőrzéséhez van szükség.           |

##### Returns

void

### 4.45.2.2 free\_starmap()

```
void free_starmap (
    StarMap * sm )
```

free\_starmap

Ez a függvény a parameterkent kapott csillagterkep csillagainak listajat, majd magát a csillagterket szabadítja fel.

#### Parameters

|    |     |  |
|----|-----|--|
| in | *sm | A felszabadítando csillagterkep pointer. |
|----|-----|--|

#### Returns

void

### 4.45.2.3 starmap\_init()

```
StarMap * starmap_init (
    int width,
    int height )
```

starmap\_init

Ez a függvény inicializálja a StarMap csillagterkep listáját. létrehoz egy, a csillagok vart szamanak megfelelo hosszúsagu dinamikus tombot, majd abban elhelyezi a sorban generalt csillagokat. Visszateresi erteke egy csillagterkep.

#### Parameters

|    |        |  |
|----|--------|--|
| in | width  | A kepernyo szelessege. Erre a csillagok x koordinatajanak generalasahoz van szukseg. |
| in | height | A kepernyo magassaga. Erre a csillagok y koordinatajanak generalasahoz van szukseg.  |

#### Returns

StarMap

## 4.46 star\_map.h

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

```
1
5 #ifndef STAR_MAP_H_INCLUDED
6 #define STAR_MAP_H_INCLUDED
7
8 #include "random_number_in_interval.h"
9
10 #include "debugmalloc.h"
11
12 typedef struct starcolor{
```

```

19     int r;
20     int g;
21     int b;
22     int a;
23 }StarColor;
24
31 typedef struct star{
32     int y_coor;
33     int x_coor;
34     int radius;
35 }Star;
36
42 typedef struct starmap{
43     int length;
44     Star *stars;
45     StarColor color;
46 }StarMap;
47
48
49 StarMap *starmap_init(int width, int height);
50
51
52 void advance_starmap_frame(StarMap *sm, int width, int height);
53
54
55 void free_starmap(StarMap *sm);
56
57 #endif // STAR_MAP_H_INCLUDED

```

## 4.47 string\_operations.c File Reference

```
#include "string_operations.h"
```

### Functions

- bool [dinstr\\_alloc](#) ([DinStr](#) \*str, int size)  
*dinstr\_alloc*

### 4.47.1 Function Documentation

#### 4.47.1.1 dinstr\_alloc()

```
bool dinstr_alloc (
    DinStr * str,
    int size )
```

[dinstr\\_alloc](#)

dinamikus sztringet allokal

#### Parameters

|           |      |
|-----------|------|
| <i>[]</i> | str  |
| <i>[]</i> | size |

### Returns

bool

## 4.48 string\_operations.h File Reference

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

### Data Structures

- struct [DinStr](#)

### Typedefs

- typedef struct [DinStr](#) [DinStr](#)

#### 4.48.1 Typedef Documentation

##### 4.48.1.1 DinStr

```
typedef struct DinStr DinStr
```

## 4.49 string\_operations.h

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

```
1 #ifndef STRING_OPERATIONS_H_INCLUDED
2 #define STRING_OPERATIONS_H_INCLUDED
3
4 #include <string.h>
5
6 #include "debugmalloc.h"
7
8 typedef struct DinStr {
9     int size;
10    char *str;
11 } DinStr;
12
13
14 #endif // STRING_OPERATIONS_H_INCLUDED
```

## 4.50 texture\_data.h File Reference

```
#include "debugmalloc.h"
```

## Data Structures

- struct [spritemapdata](#)  
*SpriteMapData.*
- struct [texturedata](#)  
*TextureData.*

## Typedefs

- typedef struct [spritemapdata](#) [SpriteMapData](#)  
*SpriteMapData.*
- typedef struct [texturedata](#) [TextureData](#)  
*TextureData.*

### 4.50.1 Typedef Documentation

#### 4.50.1.1 SpriteMapData

```
typedef struct spritemapdata SpriteMapData
```

[SpriteMapData](#).

#### 4.50.1.2 TextureData

```
typedef struct texturedata TextureData
```

[TextureData](#).

## 4.51 texture\_data.h

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

```
1 #ifndef TEXTURE_DATA_H_INCLUDED
2 #define TEXTURE_DATA_H_INCLUDED
3
4 #include "debugmalloc.h"
5
10 typedef struct spritemapdata
11 {
12     int x\_coor;
13     int y\_coor;
14     int width;
15     int height;
16 }SpriteMapData;
17
18
23 typedef struct texturedata
24 {
25     int width;
26     int height;
27     int texture\_center\_x;
28     int texture\_center\_y;
29 }TextureData;
30
31 #endif // TEXTURE_DATA_H_INCLUDED
```



## 4.52 torpedo.c File Reference

```
#include "torpedo.h"
```

### Functions

- `TorpedoShot * add_torpedo_shot (TorpedoShot *torpedoes, int damage, int speed, int x_coor, int y_coor, bool is_enemy_torpedo)`  
*add\_torpedo\_shot*
- `void move_torpedoes (TorpedoShot **torpedo)`  
*move\_torpedoes*
- `void pop_torpedo_shot (TorpedoShot **torpedo)`  
*pop\_torpedo\_shot*
- `void free_torpedoes (TorpedoShot *torpedoes)`  
*free\_torpedoes*

### 4.52.1 Function Documentation

#### 4.52.1.1 add\_torpedo\_shot()

```
TorpedoShot * add_torpedo_shot (  
    TorpedoShot * torpedoes,  
    int damage,  
    int speed,  
    int x_coor,  
    int y_coor,  
    bool is_enemy_torpedo )
```

`add_torpedo_shot`

Hozzaad a kilott torpedok listajahoz egy ujabb elemet.

#### Parameters

|    |                         |  |
|----|-------------------------|--|
| in | <i>torpedoes</i>        |  |
| in | <i>damage</i>           |  |
| in | <i>speed</i>            |  |
| in | <i>x_coor</i>           |  |
| in | <i>y_coor</i>           |  |
| in | <i>is_enemy_torpedo</i> |  |

#### Returns

`TorpedoShot`

#### 4.52.1.2 free\_torpedoes()

```
void free_torpedoes (
    TorpedoShot * torpedoes )
```

free\_torpedoes

Felszabadítja a torpedók listáját.

##### Parameters

|    |                  |  |
|----|------------------|--|
| in | <i>torpedoes</i> |  |
|----|------------------|--|

##### Returns

void

#### 4.52.1.3 move\_torpedoes()

```
void move_torpedoes (
    TorpedoShot ** torpedo )
```

move\_torpedoes

A torpedók mozgáshoz szükséges számításokat végzi.

##### Parameters

|         |                          |  |
|---------|--------------------------|--|
| in, out | <b>**</b> <i>torpedo</i> |  |
|---------|--------------------------|--|

##### Returns

void

#### 4.52.1.4 pop\_torpedo\_shot()

```
void pop_torpedo_shot (
    TorpedoShot ** torpedo )
```

pop\_torpedo\_shot

amennyiben a torpedo eltalál valamit, vagy kimegy a játéktérből, törli azt.

## Parameters

|         |           |  |
|---------|-----------|--|
| in, out | **torpedo |  |
|---------|-----------|--|

## Returns

void

## 4.53 torpedo.h File Reference

```
#include "game_attributes.h"
#include <stdio.h>
#include <stdbool.h>
#include "debugmalloc.h"
```

### Data Structures

- struct [colordata](#)  
*ColorData.*
- struct [torpedocolors](#)  
*TorpedoColors.*
- struct [torpedoshot](#)  
*TorpedoShot.*

### Typedefs

- typedef struct [colordata](#) [ColorData](#)  
*ColorData.*
- typedef struct [torpedocolors](#) [TorpedoColors](#)  
*TorpedoColors.*
- typedef struct [torpedoshot](#) [TorpedoShot](#)  
*TorpedoShot.*

### Functions

- [TorpedoShot \\*](#) [add\\_torpedo\\_shot](#) ([TorpedoShot](#) \*torpedoes, int damage, int speed, int x\_coor, int y\_coor, bool is\_enemy\_torpedo)  
*add\_torpedo\_shot*
- void [move\\_torpedoes](#) ([TorpedoShot](#) \*\*torpedo)  
*move\_torpedoes*
- void [pop\\_torpedo\\_shot](#) ([TorpedoShot](#) \*\*torpedo)  
*pop\_torpedo\_shot*
- void [free\\_torpedoes](#) ([TorpedoShot](#) \*torpedoes)  
*free\_torpedoes*

### 4.53.1 Typedef Documentation

#### 4.53.1.1 ColorData

```
typedef struct colordata ColorData
```

ColorData.

#### 4.53.1.2 TorpedoColors

```
typedef struct torpedocolors TorpedoColors
```

TorpedoColors.

#### 4.53.1.3 TorpedoShot

```
typedef struct torpedoshot TorpedoShot
```

TorpedoShot.

### 4.53.2 Function Documentation

#### 4.53.2.1 add\_torpedo\_shot()

```
TorpedoShot * add_torpedo_shot (
    TorpedoShot * torpedoes,
    int damage,
    int speed,
    int x_coor,
    int y_coor,
    bool is_enemy_torpedo )
```

add\_torpedo\_shot

Hozzaad a kilott torpedok listajához egy újabb elemet.

#### Parameters

|    |                         |  |
|----|-------------------------|--|
| in | <i>torpedoes</i>        |  |
| in | <i>damage</i>           |  |
| in | <i>speed</i>            |  |
| in | <i>x_coor</i>           |  |
| in | <i>y_coor</i>           |  |
| in | <i>is_enemy_torpedo</i> |  |

**Returns**

TorpedoShot

**4.53.2.2 free\_torpedoes()**

```
void free_torpedoes (
    TorpedoShot * torpedoes )
```

free\_torpedoes

Felszabadítja a torpedók listáját.

**Parameters**

|    |                  |  |
|----|------------------|--|
| in | <i>torpedoes</i> |  |
|----|------------------|--|

**Returns**

void

**4.53.2.3 move\_torpedoes()**

```
void move_torpedoes (
    TorpedoShot ** torpedo )
```

move\_torpedoes

A torpedók mozgáshoz szükséges számításokat végzi.

**Parameters**

|         |                          |  |
|---------|--------------------------|--|
| in, out | <b>**</b> <i>torpedo</i> |  |
|---------|--------------------------|--|

**Returns**

void

**4.53.2.4 pop\_torpedo\_shot()**

```
void pop_torpedo_shot (
    TorpedoShot ** torpedo )
```

pop\_torpedo\_shot

amennyiben a torpedo eltalál valamit, vagy kimegy a játéktérből, törli azt.

## Parameters

|         |                  |  |
|---------|------------------|--|
| in, out | <b>**torpedo</b> |  |
|---------|------------------|--|

## Returns

void

## 4.54 torpedo.h

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

```

1 #ifndef TORPEDO_H_INCLUDED
2 #define TORPEDO_H_INCLUDED
3
4 #include "game_attributes.h"
5 #include <stdio.h>
6
7 #include <stdbool.h>
8
9 #include "debugmalloc.h"
10
11 typedef struct colordata
12 {
13     int r;
14     int g;
15     int b;
16     int a;
17 }ColorData;
18
19 typedef struct torpedocolors
20 {
21     ColorData outer_ring;
22     ColorData inner_ring;
23     ColorData center;
24 }TorpedoColors;
25
26 typedef struct torpedoshot
27 {
28     int x_coor;
29     int y_coor;
30     int damage;
31     int speed;
32     int dir;
33     TorpedoColors colors;
34     struct torpedoshot *next_torpedo;
35     struct torpedoshot *prev_torpedo;
36 }TorpedoShot;
37
38 TorpedoShot *add_torpedo_shot(TorpedoShot *torpedoes, int damage, int speed, int x_coor, int y_coor, bool
    is_enemy_torpedo);
39
40 void move_torpedoes(TorpedoShot **torpedo);
41
42 void pop_torpedo_shot(TorpedoShot **torpedo);
43
44 void free_torpedoes(TorpedoShot *torpedoes);
45
46 #endif // TORPEDO_H_INCLUDED

```

## 4.55 torpedo\_hit\_management.c File Reference

```
#include "torpedo_hit_management.h"
```

## Functions

- bool [is\\_torpedo\\_out\\_of\\_bounds](#) ([TorpedoShot](#) \*\*torpedo, [GameAttributes](#) \*game\_attributes)  
*is\_torpedo\_out\_of\_bounds*
- void [explode\\_torpedo](#) ([TorpedoShot](#) \*\*torpedo, [TorpedoShot](#) \*\*temp\_torpedo)  
*explode\_torpedo*

## 4.55.1 Function Documentation

### 4.55.1.1 explode\_torpedo()

```
void explode_torpedo (
    TorpedoShot ** torpedo,
    TorpedoShot ** temp_torpedo )
```

explode\_torpedo

Felszabadítja a jatekos felrobbant torpedókat és kezeli az ahhoz tartozó pointereket.

#### Parameters

|         |                       |  |
|---------|-----------------------|--|
| in, out | <b>**torpedo</b>      | A jatekos által kilőtt torpedókat tartalmazó lancolt lista aktuális elemének pointerre.          |
| in, out | <b>**temp_torpedo</b> | A jatekos által kilőtt torpedókat tartalmazó lancolt lista head pointerének ideiglenes tárolója. |

#### Returns

void

### 4.55.1.2 is\_torpedo\_out\_of\_bounds()

```
bool is_torpedo_out_of_bounds (
    TorpedoShot ** torpedo,
    GameAttributes * game_attributes )
```

is\_torpedo\_out\_of\_bounds

Erzékeli, ha a torpedó kilepett a játéktérből.

#### Parameters

|    |                         |   |
|----|-------------------------|---|
| in | <b>**torpedo</b>        | A torpedókat tartalmazó lancolt lista adott eleme.        |
| in | <b>*game_attributes</b> | A játék attribútumait tartalmazó adatszerkezet pointerre. |

#### Returns

bool igaz értéket ad, ha az adott torpedó kilepett a játéktérből, egyébként hamis értéket ad.

## 4.56 torpedo\_hit\_management.h File Reference

```
#include "game_attributes.h"
```

```
#include "torpedo.h"
#include <stdbool.h>
#include "debugmalloc.h"
```

## Functions

- bool [is\\_torpedo\\_out\\_of\\_bounds](#) ([TorpedoShot](#) \*\*torpedo, [GameAttributes](#) \*game\_attributes)  
*is\_torpedo\_out\_of\_bounds*
- void [explode\\_torpedo](#) ([TorpedoShot](#) \*\*player\_torpedo, [TorpedoShot](#) \*\*temp\_torpedo)  
*explode\_torpedo*

### 4.56.1 Function Documentation

#### 4.56.1.1 explode\_torpedo()

```
void explode_torpedo (
    TorpedoShot ** torpedo,
    TorpedoShot ** temp_torpedo )
```

*explode\_torpedo*

Felszabadítja a játékos felrobbant torpedókat és kezeli az ahhoz tartozó pointereket.

##### Parameters

|         |                        |  |
|---------|------------------------|--|
| in, out | ** <i>torpedo</i>      | A játékos által kilőtt torpedókat tartalmazó lancolt lista aktuális elemének pointerre.          |
| in, out | ** <i>temp_torpedo</i> | A játékos által kilőtt torpedókat tartalmazó lancolt lista head pointerének ideiglenes tárolója. |

##### Returns

void

#### 4.56.1.2 is\_torpedo\_out\_of\_bounds()

```
bool is_torpedo_out_of_bounds (
    TorpedoShot ** torpedo,
    GameAttributes * game_attributes )
```

*is\_torpedo\_out\_of\_bounds*

Erzékeli, ha a torpedó kilepett a játékterből.



## Parameters

|    |                          |  |
|----|--------------------------|--|
| in | ** <i>torpedo</i>        | A torpedokat tartalmazó lancolt lista adott eleme.       |
| in | * <i>game_attributes</i> | A játék attribútumait tartalmazó adatszerkezet pointere. |

## Returns

bool igaz értéket ad, ha az adott torpedo kilepett a játékból, egyébként hamis értéket ad.

## 4.57 torpedo\_hit\_management.h

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

```

1 #ifndef TORPEDO_HIT_MANAGEMENT_H_INCLUDED
2 #define TORPEDO_HIT_MANAGEMENT_H_INCLUDED
3
4 #include "game_attributes.h"
5 #include "torpedo.h"
6
7 #include <stdbool.h>
8
9 #include "debugmalloc.h"
10
11 bool is_torpedo_out_of_bounds(TorpedoShot **torpedo, GameAttributes *game_attributes);
12
13 void explode_torpedo(TorpedoShot **player_torpedo, TorpedoShot **temp_torpedo);
14
15
16 #endif // TORPEDO_HIT_MANAGEMENT_H_INCLUDED

```

## 4.58 ui\_input.c File Reference

```
#include "ui_input.h"
```

## Functions

- void [user\\_input](#) ([InputStateInterface](#) \*isi, [KeyMap](#) \*key\_map, SDL\_TimerID id)  
*user\_input*

### 4.58.1 Function Documentation

#### 4.58.1.1 user\_input()

```

void user_input (
    InputStateInterface * isi,
    KeyMap * key_map,
    SDL_TimerID id )

```

*user\_input*

A felhasználótól érkező billentyűparancsokat értelmezi, és egy interface-n keresztül adja át a program többi részének

## Parameters

|         |                 |  |
|---------|-----------------|--|
| in, out | <i>*isi</i>     | a jatek InputStateInterface-re mutato pointer. Ezen keresztül kommunikálnak egymással a vezermódulok.  |
| in      | <i>*key_map</i> | ez a vezermo KeyMap interfacen keresztül hasonlítja össze a bejovo billentyűparancsokat a valid vezermo gombokkal.   |
| in      | <i>id</i>       | egy SDL_TimerID típusu időzítő. Feladata, hogy general egy SDL_USEREVENTet, amennyiben az időzítő lejártával nincs beérkező esemény/parancs (enélkül a vezermo blokkolna a program futását, nem működne a háttér animáció, és semmi nem történne, amíg nincs felhasználói interakció). |

## Returns

void

## 4.59 ui\_input.h File Reference

```
#include "input_state_interface.h"
#include "keymap.h"
#include "SDL_timer.h"
#include <stdbool.h>
#include <SDL.h>
#include <SDL2_gfxPrimitives.h>
#include <stdio.h>
#include "debugmalloc.h"
```

### Functions

- void [user\\_input](#) ([InputStateInterface](#) \*isi, [KeyMap](#) \*key\_map, SDL\_TimerID id)  
*user\_input*

### 4.59.1 Function Documentation

#### 4.59.1.1 user\_input()

```
void user_input (
    InputStateInterface * isi,
    KeyMap * key_map,
    SDL_TimerID id )
```

*user\_input*

A felhasználótól érkező billentyűparancsokat értelmezi, és egy interface-n keresztül adja át a program többi részének

## Parameters

|         |                 |  |
|---------|-----------------|--|
| in, out | <i>*isi</i>     | a játék InputStateInterface-re mutató pointer. Ezen keresztül kommunikálnak egymással a vezérlőmodulok.  |
| in      | <i>*key_map</i> | ez a vezérlő KeyMap interfácen keresztül hasonlítja össze a bejövő billentyűparancsokat a valid vezérlő gombokkal.   |
| in      | <i>id</i>       | egy SDL_TimerID típusú időzítő. Feladata, hogy generáljon egy SDL_USEREVENT-et, amennyiben az időzítő lejártával nincs beérkező esemény/parancs (enélkül a vezérlő blokkolná a program futását, nem működne a háttér animáció, és semmi nem történne, amíg nincs felhasználói interakció). |

## Returns

void

## 4.60 ui\_input.h

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

```
1 #ifndef UI_INPUT_H_INCLUDED
2 #define UI_INPUT_H_INCLUDED
3
4 #include "input_state_interface.h"
5 #include "keymap.h"
6
7 #include "SDL_timer.h"
8 #include <stdbool.h>
9 #include <SDL.h>
10 #include <SDL2_gfxPrimitives.h>
11
12 #include <stdio.h>
13
14 #include "debugmalloc.h"
15
16 void user_input(InputStateInterface *isi, KeyMap *key_map, SDL_TimerID id);
17
18 #endif // UI_INPUT_H_INCLUDED
```



# Index

## a

- colordata, [5](#)
- starcolor, [25](#)
- add\_torpedo\_shot
  - torpedo.c, [93](#)
  - torpedo.h, [96](#)
- advance\_starmap\_frame
  - star\_map.c, [85](#)
  - star\_map.h, [88](#)
- all\_alloc\_bytes
  - DebugmallocData, [6](#)
- all\_alloc\_count
  - DebugmallocData, [6](#)
- alloc\_bytes
  - DebugmallocData, [6](#)
- alloc\_count
  - DebugmallocData, [7](#)

## b

- colordata, [5](#)
- starcolor, [25](#)
- calloc
  - debugmalloc.h, [32](#)
- center
  - torpedocolors, [29](#)
- centerline\_y\_coor
  - enemyship, [11](#)
  - playership, [20](#)
- clear\_screen
  - graphics.c, [60](#)
  - graphics.h, [66](#)
- color
  - starmap, [27](#)
- ColorData
  - torpedo.h, [96](#)
- colordata, [5](#)
  - a, [5](#)
  - b, [5](#)
  - g, [5](#)
  - r, [6](#)
- colors
  - torpedoshot, [30](#)
- create\_font
  - graphics.c, [60](#)
  - graphics.h, [66](#)
- create\_textures
  - graphics.c, [60](#)
  - graphics.h, [67](#)
- create\_window

- graphics.c, [61](#)
- graphics.h, [67](#)

## damage

- torpedoshot, [30](#)
- data\_transfer\_types.h, [31](#)
  - ShipDTT, [31](#)
- debugmalloc.h, [32](#)
  - calloc, [32](#)
  - debugmalloc\_canary\_char, [34](#)
  - debugmalloc\_canary\_size, [34](#)
  - debugmalloc\_max\_block\_size\_default, [34](#)
  - debugmalloc\_tablesize, [34](#)
  - DebugmallocData, [33](#)
  - DebugmallocEntry, [33](#)
  - free, [32](#)
  - malloc, [33](#)
  - realloc, [33](#)
- debugmalloc\_canary\_char
  - debugmalloc.h, [34](#)
- debugmalloc\_canary\_size
  - debugmalloc.h, [34](#)
- debugmalloc\_max\_block\_size\_default
  - debugmalloc.h, [34](#)
- debugmalloc\_tablesize
  - debugmalloc.h, [34](#)
- DebugmallocData, [6](#)
  - all\_alloc\_bytes, [6](#)
  - all\_alloc\_count, [6](#)
  - alloc\_bytes, [6](#)
  - alloc\_count, [7](#)
  - debugmalloc.h, [33](#)
  - head, [7](#)
  - logfile, [7](#)
  - max\_block\_size, [7](#)
  - tail, [7](#)
- DebugmallocEntry, [7](#)
  - debugmalloc.h, [33](#)
  - expr, [8](#)
  - file, [8](#)
  - func, [8](#)
  - line, [8](#)
  - next, [8](#)
  - prev, [8](#)
  - real\_mem, [9](#)
  - size, [9](#)
  - user\_mem, [9](#)
- destroy\_textures
  - graphics.c, [61](#)
  - graphics.h, [68](#)

- DinStr, 9
  - size, 9
  - str, 9
  - string\_operations.h, 91
- dinstr\_alloc
  - string\_operations.c, 90
- dir
  - torpedoshot, 30
- down
  - inputstateinterface, 16
- downkey
  - keymap, 18
- draw\_background
  - graphics.c, 61
  - graphics.h, 68
- draw\_end\_screen
  - graphics.c, 62
  - graphics.h, 68
- draw\_enemy\_ships
  - graphics.c, 62
  - graphics.h, 68
- draw\_LCARS\_bacground
  - graphics.c, 62
  - graphics.h, 69
- draw\_player\_ship
  - graphics.c, 63
  - graphics.h, 69
- draw\_score
  - graphics.c, 63
  - graphics.h, 70
- draw\_torpedo
  - graphics.c, 63
  - graphics.h, 70
- enemy\_armada
  - gameassets, 13
- enemy\_armada\_size
  - gameattributes, 15
- enemy\_hit\_management.h, 40
  - explode\_enemy\_ship\_if\_dead, 40
- enemy\_hit\_mananagement.c, 41
  - explode\_enemy\_ship\_if\_dead, 41
- enemy\_ship.c, 42
  - find\_max\_enemy\_armada\_y\_coor, 42
  - free\_enemy\_armada, 42
  - init\_enemy\_armada, 43
  - move\_enemy\_armada, 43
  - pop\_enemy\_ship, 44
- enemy\_ship.h, 44
  - EnemyShip, 45
  - find\_max\_enemy\_armada\_y\_coor, 45
  - free\_enemy\_armada, 46
  - init\_enemy\_armada, 46
  - move\_enemy\_armada, 46
  - pop\_enemy\_ship, 47
- enemy\_ships\_per\_row
  - gameattributes, 15
- enemy\_torpedoes
  - gameassets, 13
- EnemyShip
  - enemy\_ship.h, 45
- enemyship, 10
  - centerline\_y\_coor, 11
  - health, 11
  - hitbox\_beg\_coor, 11
  - hitbox\_end\_coor, 11
  - movement\_dir, 11
  - next\_ship, 11
  - prev\_ship, 12
  - score\_value, 12
  - speed, 12
  - sprite\_map\_data, 12
  - texture\_data, 12
  - x\_coor, 12
  - y\_coor, 13
- explode\_enemy\_ship\_if\_dead
  - enemy\_hit\_management.h, 40
  - enemy\_hit\_mananagement.c, 41
- explode\_player\_ship\_if\_dead
  - player\_hit\_management.c, 77
  - player\_hit\_management.h, 78
- explode\_torpedo
  - torpedo\_hit\_management.c, 99
  - torpedo\_hit\_management.h, 100
- expr
  - DebugmallocEntry, 8
- file
  - DebugmallocEntry, 8
- file\_management.c, 48
  - import\_ship\_dtt, 48
  - read\_texture\_data, 49
- file\_management.h, 49
  - import\_ship\_dtt, 50
  - read\_texture\_data, 50
- find\_max\_enemy\_armada\_y\_coor
  - enemy\_ship.c, 42
  - enemy\_ship.h, 45
- fire\_enemy\_torpedoes
  - fire\_management.h, 51
  - fire\_managemet.c, 53
- fire\_management.h, 51
  - fire\_enemy\_torpedoes, 51
  - fire\_player\_torpedo, 52
- fire\_managemet.c, 52
  - fire\_enemy\_torpedoes, 53
  - fire\_player\_torpedo, 53
- fire\_player\_torpedo
  - fire\_management.h, 52
  - fire\_managemet.c, 53
- free
  - debugmalloc.h, 32
- free\_enemy\_armada
  - enemy\_ship.c, 42
  - enemy\_ship.h, 46
- free\_player\_ship
  - player\_ship.c, 79
  - player\_ship.h, 81

- free\_ship\_dtt
  - game\_engine.c, 56
- free\_starmap
  - star\_map.c, 86
  - star\_map.h, 88
- free\_torpedoes
  - torpedo.c, 93
  - torpedo.h, 97
- func
  - DebugmallocEntry, 8
- g
  - colordata, 5
  - starcolor, 26
- game\_assets.h, 54
  - GameAssets, 54
- game\_attributes.h, 55
  - GameAttributes, 55
- game\_engine.c, 55
  - free\_ship\_dtt, 56
  - input\_timer, 56
  - runtime, 57
- game\_engine.h, 57
  - input\_timer, 58
  - runtime, 58
- game\_over
  - inputstateinterface, 16
- game\_score
  - gameattributes, 15
- GameAssets
  - game\_assets.h, 54
- gameassets, 13
  - enemy\_armada, 13
  - enemy\_torpedoes, 13
  - player\_ship, 14
  - player\_torpedoes, 14
  - star\_map, 14
- GameAttributes
  - game\_attributes.h, 55
- gameattributes, 14
  - enemy\_armada\_size, 15
  - enemy\_ships\_per\_row, 15
  - game\_score, 15
  - height, 15
  - id, 15
  - isi, 15
  - num\_of\_rows, 15
  - width, 15
- graphics.c, 59
  - clear\_screen, 60
  - create\_font, 60
  - create\_textures, 60
  - create\_window, 61
  - destroy\_textures, 61
  - draw\_background, 61
  - draw\_end\_screen, 62
  - draw\_enemy\_ships, 62
  - draw\_LCARS\_bacground, 62
  - draw\_player\_ship, 63
  - draw\_score, 63
  - draw\_torpedo, 63
  - load\_sdl\_texture, 64
  - open\_font, 64
  - render\_screen, 64
  - sdl\_init, 65
- graphics.h, 65
  - clear\_screen, 66
  - create\_font, 66
  - create\_textures, 67
  - create\_window, 67
  - destroy\_textures, 68
  - draw\_background, 68
  - draw\_end\_screen, 68
  - draw\_enemy\_ships, 68
  - draw\_LCARS\_bacground, 69
  - draw\_player\_ship, 69
  - draw\_score, 70
  - draw\_torpedo, 70
  - render\_screen, 70
- head
  - DebugmallocData, 7
- health
  - enemyship, 11
  - playership, 20
  - shipdtt, 22
- height
  - gameattributes, 15
  - spritemapdata, 23
  - texturedata, 28
- hit\_management.c, 71
  - manage\_enemy\_hits, 72
  - manage\_player\_hits, 72
  - remove\_torpedo\_if\_out\_of\_bounds, 72
- hit\_management.h, 73
  - manage\_enemy\_hits, 73
  - remove\_torpedo\_if\_out\_of\_bounds, 74
- hitbox\_beg\_coor
  - enemyship, 11
  - playership, 20
- hitbox\_end\_coor
  - enemyship, 11
  - playership, 20
- id
  - gameattributes, 15
- import\_ship\_dtt
  - file\_management.c, 48
  - file\_management.h, 50
- init\_enemy\_armada
  - enemy\_ship.c, 43
  - enemy\_ship.h, 46
- init\_player\_ship
  - player\_ship.c, 79
  - player\_ship.h, 82
- inner\_ring
  - torpedocolors, 29
- input\_state\_interface.h, 75

- InputStateInterface, 75
- input\_timer
  - game\_engine.c, 56
  - game\_engine.h, 58
- InputStateInterface
  - input\_state\_interface.h, 75
- inputstateinterface, 16
  - down, 16
  - game\_over, 16
  - left, 16
  - n, 17
  - quit, 17
  - restart, 17
  - right, 17
  - torpedo, 17
  - torpedo\_ready, 17
  - up, 17
  - y, 17
- is\_torpedo\_out\_of\_bounds
  - torpedo\_hit\_management.c, 99
  - torpedo\_hit\_management.h, 100
- isi
  - gameattributes, 15
- KeyMap
  - keymap.h, 76
- keymap, 18
  - downkey, 18
  - leftkey, 18
  - rightkey, 18
  - torpedokey, 19
  - upkey, 19
- keymap.h, 76
  - KeyMap, 76
- left
  - inputstateinterface, 16
- leftkey
  - keymap, 18
- length
  - starmap, 27
- line
  - DebugmallocEntry, 8
- load\_sdl\_texture
  - graphics.c, 64
- logfile
  - DebugmallocData, 7
- main
  - main.c, 77
- main.c, 76
  - main, 77
- malloc
  - debugmalloc.h, 33
- manage\_enemy\_hits
  - hit\_management.c, 72
  - hit\_management.h, 73
- manage\_player\_hits
  - hit\_management.c, 72
- max\_block\_size
  - DebugmallocData, 7
- move\_enemy\_armada
  - enemy\_ship.c, 43
  - enemy\_ship.h, 46
- move\_player\_ship
  - player\_ship.c, 80
  - player\_ship.h, 82
- move\_torpedoes
  - torpedo.c, 94
  - torpedo.h, 97
- movement\_dir
  - enemyship, 11
- n
  - inputstateinterface, 17
- next
  - DebugmallocEntry, 8
- next\_ship
  - enemyship, 11
- next\_torpedo
  - torpedoshot, 30
- num\_of\_rows
  - gameattributes, 15
- open\_font
  - graphics.c, 64
- outter\_ring
  - torpedocolors, 29
- player\_hit\_management.c, 77
  - explode\_player\_ship\_if\_dead, 77
- player\_hit\_management.h, 78
  - explode\_player\_ship\_if\_dead, 78
- player\_ship
  - gameassets, 14
- player\_ship.c, 79
  - free\_player\_ship, 79
  - init\_player\_ship, 79
  - move\_player\_ship, 80
- player\_ship.h, 80
  - free\_player\_ship, 81
  - init\_player\_ship, 82
  - move\_player\_ship, 82
  - PlayerShip, 81
- player\_torpedoes
  - gameassets, 14
- PlayerShip
  - player\_ship.h, 81
- playership, 19
  - centerline\_y\_coor, 20
  - health, 20
  - hitbox\_beg\_coor, 20
  - hitbox\_end\_coor, 20
  - score\_value, 20
  - speed, 20
  - sprite\_map\_data, 21
  - texture\_data, 21
  - x\_coor, 21



- y\_coor, [21](#)
- pop\_enemy\_ship
  - enemy\_ship.c, [44](#)
  - enemy\_ship.h, [47](#)
- pop\_torpedo\_shot
  - torpedo.c, [94](#)
  - torpedo.h, [97](#)
- prev
  - DebugmallocEntry, [8](#)
- prev\_ship
  - enemyship, [12](#)
- prev\_torpedo
  - torpedoshot, [30](#)
- quit
  - inputstateinterface, [17](#)
- r
  - colordata, [6](#)
  - starcolor, [26](#)
- radius
  - star, [24](#)
- random\_number\_in\_interval.c, [83](#)
  - random\_number\_in\_range, [83](#)
- random\_number\_in\_interval.h, [84](#)
  - random\_number\_in\_range, [84](#)
- random\_number\_in\_range
  - random\_number\_in\_interval.c, [83](#)
  - random\_number\_in\_interval.h, [84](#)
- read\_texture\_data
  - file\_management.c, [49](#)
  - file\_management.h, [50](#)
- real\_mem
  - DebugmallocEntry, [9](#)
- realloc
  - debugmalloc.h, [33](#)
- remove\_torpedo\_if\_out\_of\_bounds
  - hit\_management.c, [72](#)
  - hit\_management.h, [74](#)
- render\_screen
  - graphics.c, [64](#)
  - graphics.h, [70](#)
- restart
  - inputstateinterface, [17](#)
- right
  - inputstateinterface, [17](#)
- rightkey
  - keymap, [18](#)
- runtime
  - game\_engine.c, [57](#)
  - game\_engine.h, [58](#)
- score\_value
  - enemyship, [12](#)
  - playership, [20](#)
  - shipdtt, [22](#)
- sdl\_init
  - graphics.c, [65](#)
- ShipDTT
  - data\_transfer\_types.h, [31](#)
- shipdtt, [21](#)
  - health, [22](#)
  - score\_value, [22](#)
  - speed, [22](#)
- size
  - DebugmallocEntry, [9](#)
  - DinStr, [9](#)
- speed
  - enemyship, [12](#)
  - playership, [20](#)
  - shipdtt, [22](#)
  - torpedoshot, [30](#)
- sprite\_map\_data
  - enemyship, [12](#)
  - playership, [21](#)
- SpriteMapData
  - texture\_data.h, [92](#)
- spritemapdata, [23](#)
  - height, [23](#)
  - width, [23](#)
  - x\_coor, [23](#)
  - y\_coor, [23](#)
- Star
  - star\_map.h, [87](#)
- star, [24](#)
  - radius, [24](#)
  - x\_coor, [24](#)
  - y\_coor, [24](#)
- star\_map
  - gameassets, [14](#)
- star\_map.c, [85](#)
  - advance\_starmap\_frame, [85](#)
  - free\_starmap, [86](#)
  - starmap\_init, [86](#)
- star\_map.h, [87](#)
  - advance\_starmap\_frame, [88](#)
  - free\_starmap, [88](#)
  - Star, [87](#)
  - StarColor, [87](#)
  - StarMap, [88](#)
  - starmap\_init, [89](#)
- StarColor
  - star\_map.h, [87](#)
- starcolor, [25](#)
  - a, [25](#)
  - b, [25](#)
  - g, [26](#)
  - r, [26](#)
- StarMap
  - star\_map.h, [88](#)
- starmap, [26](#)
  - color, [27](#)
  - length, [27](#)
  - stars, [27](#)
- starmap\_init
  - star\_map.c, [86](#)
  - star\_map.h, [89](#)

- stars
  - starmap, 27
- str
  - DinStr, 9
- string\_operations.c, 90
  - dinstr\_alloc, 90
- string\_operations.h, 91
  - DinStr, 91
- tail
  - DebugmallocData, 7
- texture\_center\_x
  - texturedata, 28
- texture\_center\_y
  - texturedata, 28
- texture\_data
  - enemyship, 12
  - playership, 21
- texture\_data.h, 91
  - SpriteMapData, 92
  - TextureData, 92
- TextureData
  - texture\_data.h, 92
- texturedata, 27
  - height, 28
  - texture\_center\_x, 28
  - texture\_center\_y, 28
  - width, 28
- torpedo
  - inputstateinterface, 17
- torpedo.c, 93
  - add\_torpedo\_shot, 93
  - free\_torpedoes, 93
  - move\_torpedoes, 94
  - pop\_torpedo\_shot, 94
- torpedo.h, 95
  - add\_torpedo\_shot, 96
  - ColorData, 96
  - free\_torpedoes, 97
  - move\_torpedoes, 97
  - pop\_torpedo\_shot, 97
  - TorpedoColors, 96
  - TorpedoShot, 96
- torpedo\_hit\_management.c, 98
  - explode\_torpedo, 99
  - is\_torpedo\_out\_of\_bounds, 99
- torpedo\_hit\_management.h, 99
  - explode\_torpedo, 100
  - is\_torpedo\_out\_of\_bounds, 100
- torpedo\_ready
  - inputstateinterface, 17
- TorpedoColors
  - torpedo.h, 96
- torpedocolors, 28
  - center, 29
  - inner\_ring, 29
  - outter\_ring, 29
- torpedokey
  - keymap, 19
- TorpedoShot
  - torpedo.h, 96
- torpedoshot, 29
  - colors, 30
  - damage, 30
  - dir, 30
  - next\_torpedo, 30
  - prev\_torpedo, 30
  - speed, 30
  - x\_coor, 30
  - y\_coor, 30
- ui\_input.c, 101
  - user\_input, 101
- ui\_input.h, 102
  - user\_input, 102
- up
  - inputstateinterface, 17
- upkey
  - keymap, 19
- user\_input
  - ui\_input.c, 101
  - ui\_input.h, 102
- user\_mem
  - DebugmallocEntry, 9
- width
  - gameattributes, 15
  - spritemapdata, 23
  - texturedata, 28
- x\_coor
  - enemyship, 12
  - playership, 21
  - spritemapdata, 23
  - star, 24
  - torpedoshot, 30
- y
  - inputstateinterface, 17
- y\_coor
  - enemyship, 13
  - playership, 21
  - spritemapdata, 23
  - star, 24
  - torpedoshot, 30