

Frogger

Generated by Doxygen 1.9.3

1 Data Structure Index	1
1.1 Data Structures	1
2 File Index	3
2.1 File List	3
3 Data Structure Documentation	5
3.1 ALGIF_ANIMATION Struct Reference	5
3.2 ALGIF_BITMAP Struct Reference	5
3.3 ALGIF_FRAME Struct Reference	6
3.4 ALGIF_PALETTE Struct Reference	6
3.5 ALGIF_RGB Struct Reference	6
3.6 allegro_t Struct Reference	7
3.7 car_t Struct Reference	7
3.8 coin_t Struct Reference	7
3.9 data_t Struct Reference	8
3.10 dcoord_t Struct Reference	8
3.11 frog_t Struct Reference	8
3.12 jcoord_t Struct Reference	9
3.13 log_t Struct Reference	9
3.14 Mensaje Struct Reference	9
3.15 menu_t Struct Reference	10
3.16 nodeT Struct Reference	10
3.17 pair_xy_t Struct Reference	10
3.18 privateAudioDevice Struct Reference	10
3.19 renglon_t Union Reference	11
3.20 sound Struct Reference	11
3.21 sounds_t Struct Reference	11
3.22 sprites_menu_t Struct Reference	12
3.23 sprites_t Struct Reference	12
3.23.1 Detailed Description	13
3.24 state_diagram_edge Struct Reference	13
3.25 turtle_pack_t Struct Reference	14
3.26 window_t Struct Reference	14
4 File Documentation	15
4.1 src/display.h File Reference	15
4.1.1 Detailed Description	16
4.1.2 Function Documentation	16
4.1.2.1 cargarRanking()	16
4.1.2.2 dejarTexto()	16
4.1.2.3 iniciarDisplay()	17
4.1.2.4 mostrarCreditos()	17

4.1.2.5 mostrarTexto()	17
4.2 display.h	17
4.3 src/fsm.c File Reference	18
4.3.1 Detailed Description	19
4.3.2 Function Documentation	20
4.3.2.1 fixHighCpuUsage()	20
4.3.2.2 fsm()	20
4.3.2.3 inicializarFsm()	20
4.3.3 Variable Documentation	20
4.3.3.1 en_game_over	21
4.3.3.2 en_game_over_esperando_opcion	21
4.3.3.3 en_menu_ppal	21
4.3.3.4 en_pausa	21
4.3.3.5 en_pausa_esperando_opcion	22
4.3.3.6 jugando	22
4.3.3.7 menu_ppal_esperando_opcion	22
4.3.3.8 poniendo_nombre	22
4.3.3.9 seleccionando_dificultad	23
4.3.3.10 viendo_creditos	23
4.3.3.11 viendo_ranking	23
4.4 src/fsm.h File Reference	23
4.4.1 Detailed Description	24
4.4.2 Function Documentation	24
4.4.2.1 fixHighCpuUsage()	24
4.4.2.2 fsm()	24
4.4.2.3 inicializarFsm()	24
4.5 fsm.h	25
4.6 src/game.h File Reference	25
4.6.1 Detailed Description	26
4.6.2 Function Documentation	26
4.6.2.1 getMaxPuntos()	27
4.6.2.2 getNivel()	27
4.6.2.3 getNombre()	27
4.6.2.4 getPuntos()	27
4.6.2.5 setDificultad()	27
4.6.2.6 setMaxPuntos()	28
4.6.2.7 setNombre()	28
4.6.2.8 tiempoRefrescoEntidades()	28
4.7 game.h	29
4.8 src/input.h File Reference	29
4.8.1 Detailed Description	30
4.8.2 Function Documentation	30

4.8.2.1 leerEntradas()	30
4.9 input.h	30
4.10 src/main.c File Reference	31
4.10.1 Detailed Description	31
4.11 menu.h	31
4.12 src/nombre.h File Reference	32
4.12.1 Detailed Description	32
4.12.2 Function Documentation	32
4.12.2.1 devolverNombre()	33
4.13 nombre.h	33
4.14 src/platform/pc/algif5/algif.h File Reference	33
4.14.1 Detailed Description	34
4.15 algif.h	34
4.16 src/platform/rpi/bitmap.c File Reference	35
4.16.1 Detailed Description	36
4.16.2 Function Documentation	36
4.16.2.1 copiarMatriz()	36
4.16.2.2 limpiarMatriz()	36
4.16.2.3 matrizAnd()	37
4.16.2.4 matrizNot()	37
4.16.2.5 matrizOr()	37
4.16.2.6 matrizXor()	38
4.16.2.7 printMatriz()	38
4.17 src/platform/pc/allegro_stuff.c File Reference	38
4.17.1 Detailed Description	42
4.17.2 Function Documentation	42
4.17.2.1 allegro_draw_hitbox()	42
4.17.2.2 allegro_draw_menu_background()	42
4.17.2.3 allegro_get_last_key()	43
4.17.2.4 allegro_get_next_event()	43
4.17.2.5 allegro_get_var_done()	43
4.17.2.6 allegro_get_var_event()	44
4.17.2.7 allegro_get_var_font()	44
4.17.2.8 allegro_get_var_font_h()	44
4.17.2.9 allegro_get_var_font_w()	44
4.17.2.10 allegro_get_var_redraw()	45
4.17.2.11 allegro_is_event_queueVacia()	45
4.17.2.12 allegro_set_last_key()	45
4.17.2.13 allegro_set_rick_flag()	45
4.17.2.14 allegro_set_var_done()	46
4.17.2.15 allegro_set_var_event()	46
4.17.2.16 allegro_set_var_redraw()	46

4.17.2.17 allegro_wait_for_event()	47
4.17.2.18 must_init()	47
4.18 src/platform/pc/allegro_stuff.h File Reference	47
4.18.1 Detailed Description	50
4.18.2 Function Documentation	50
4.18.2.1 allegro_draw_hitbox()	50
4.18.2.2 allegro_draw_menu_background()	51
4.18.2.3 allegro_get_last_key()	51
4.18.2.4 allegro_get_next_event()	51
4.18.2.5 allegro_get_var_done()	52
4.18.2.6 allegro_get_var_event()	52
4.18.2.7 allegro_get_var_font()	52
4.18.2.8 allegro_get_var_font_h()	52
4.18.2.9 allegro_get_var_font_w()	53
4.18.2.10 allegro_get_var_redraw()	53
4.18.2.11 allegro_is_event_queueVacia()	53
4.18.2.12 allegro_set_last_key()	53
4.18.2.13 allegro_set_rick_flag()	54
4.18.2.14 allegro_set_var_done()	54
4.18.2.15 allegro_set_var_event()	54
4.18.2.16 allegro_set_var_redraw()	54
4.18.2.17 allegro_wait_for_event()	55
4.18.2.18 must_init()	55
4.19 allegro_stuff.h	55
4.20 src/platform/pc/display.c File Reference	58
4.20.1 Detailed Description	59
4.20.2 Function Documentation	59
4.20.2.1 cargarRanking()	59
4.20.2.2 dejarTexto()	60
4.20.2.3 iniciarDisplay()	60
4.20.2.4 mostrarCreditos()	60
4.20.2.5 mostrarTexto()	61
4.21 src/platform/pc/entities.h File Reference	61
4.21.1 Detailed Description	61
4.21.2 Function Documentation	61
4.21.2.1 entities_move_frog()	61
4.22 entities.h	62
4.23 src/platform/pc/game.c File Reference	62
4.23.1 Detailed Description	63
4.23.2 Function Documentation	63
4.23.2.1 getMaxPuntos()	64
4.23.2.2 getNivel()	64

4.23.2.3	getNombre()	64
4.23.2.4	getPuntos()	64
4.23.2.5	setDificultad()	64
4.23.2.6	setMaxPuntos()	65
4.23.2.7	setNombre()	65
4.23.2.8	tiempoRefrescoEntidades()	65
4.24	src/platform/rpi/game.c File Reference	66
4.24.1	Detailed Description	67
4.24.2	Function Documentation	67
4.24.2.1	getMaxPuntos()	67
4.24.2.2	getNivel()	68
4.24.2.3	getNombre()	68
4.24.2.4	getPuntos()	68
4.24.2.5	setDificultad()	68
4.24.2.6	setMaxPuntos()	69
4.24.2.7	setNombre()	69
4.24.2.8	tiempoRefrescoEntidades()	69
4.25	src/platform/pc/game_data.c File Reference	69
4.25.1	Detailed Description	71
4.25.2	Function Documentation	71
4.25.2.1	game_data_add_name_letter()	72
4.25.2.2	game_data_are_goals_full()	72
4.25.2.3	game_data_get_diff()	72
4.25.2.4	game_data_get_frames()	72
4.25.2.5	game_data_get_game_over_flag()	73
4.25.2.6	game_data_get_goal_state()	73
4.25.2.7	game_data_get_lives()	73
4.25.2.8	game_data_get_name()	74
4.25.2.9	game_data_get_old_max_score()	74
4.25.2.10	game_data_get_run_number()	74
4.25.2.11	game_data_get_run_time_left()	74
4.25.2.12	game_data_get_score()	75
4.25.2.13	game_data_get_score_max()	75
4.25.2.14	game_data_get_time_left_flag()	75
4.25.2.15	game_data_get_timer_in_sec()	75
4.25.2.16	game_data_overwrite_name()	75
4.25.2.17	game_data_set_diff()	76
4.25.2.18	game_data_set_goal()	76
4.25.2.19	game_data_set_score_max()	76
4.26	src/platform/pc/game_data.h File Reference	76
4.26.1	Detailed Description	78
4.26.2	Function Documentation	78

4.26.2.1 game_data_add_name_letter()	78
4.26.2.2 game_data_are_goals_full()	79
4.26.2.3 game_data_get_diff()	79
4.26.2.4 game_data_get_frames()	79
4.26.2.5 game_data_get_game_over_flag()	80
4.26.2.6 game_data_get_goal_state()	80
4.26.2.7 game_data_get_lives()	80
4.26.2.8 game_data_get_name()	81
4.26.2.9 game_data_get_old_max_score()	81
4.26.2.10 game_data_get_run_number()	81
4.26.2.11 game_data_get_run_time_left()	81
4.26.2.12 game_data_get_score()	82
4.26.2.13 game_data_get_score_max()	82
4.26.2.14 game_data_get_time_left_flag()	82
4.26.2.15 game_data_get_timer_in_sec()	82
4.26.2.16 game_data_overwrite_name()	82
4.26.2.17 game_data_set_diff()	83
4.26.2.18 game_data_set_goal()	83
4.26.2.19 game_data_set_score_max()	83
4.27 game_data.h	83
4.28 src/platform/pc/geometry.c File Reference	85
4.28.1 Detailed Description	86
4.28.2 Function Documentation	86
4.28.2.1 collide()	86
4.28.2.2 collideShort()	87
4.28.2.3 get_rand_between()	87
4.28.2.4 getXYFromCarFrame()	88
4.28.2.5 getXYFromCoinFrame()	88
4.28.2.6 getXYFromFrogFrame()	88
4.28.2.7 getXYFromSplashFrame()	89
4.28.2.8 getXYFromTurtleFrame()	89
4.28.2.9 inside()	89
4.28.2.10 insideShort()	90
4.28.2.11 insideShortScaled()	91
4.28.2.12 mapInt()	91
4.28.2.13 matchUInt()	92
4.28.3 Variable Documentation	92
4.28.3.1 goal_cols	92
4.28.3.2 lanes_cars	93
4.28.3.3 lanes_logs	93
4.28.3.4 lanes_turtles	93
4.29 src/platform/pc/geometry.h File Reference	93

4.29.1 Detailed Description	96
4.29.2 Function Documentation	97
4.29.2.1 collide()	97
4.29.2.2 collideShort()	97
4.29.2.3 get_rand_between()	98
4.29.2.4 getXYFromCarFrame()	98
4.29.2.5 getXYFromCoinFrame()	99
4.29.2.6 getXYFromFrogFrame()	99
4.29.2.7 getXYFromSplashFrame()	99
4.29.2.8 getXYFromTurtleFrame()	100
4.29.2.9 inside()	100
4.29.2.10 insideShort()	101
4.29.2.11 insideShortScaled()	101
4.29.2.12 mapInt()	102
4.29.2.13 matchUInt()	102
4.29.3 Variable Documentation	103
4.29.3.1 goal_cols	103
4.29.3.2 lanes_cars	103
4.29.3.3 lanes_logs	103
4.29.3.4 lanes_turtles	103
4.30 geometry.h	104
4.31 src/platform/pc/input.c File Reference	106
4.31.1 Detailed Description	107
4.31.2 Function Documentation	107
4.31.2.1 leerEntradas()	107
4.32 src/platform/rpi/input.c File Reference	107
4.32.1 Detailed Description	108
4.32.2 Function Documentation	108
4.32.2.1 leerEntradas()	108
4.33 src/platform/pc/menu.c File Reference	108
4.33.1 Detailed Description	109
4.33.2 Function Documentation	109
4.33.2.1 getOpcion()	109
4.33.2.2 setMenu()	110
4.33.2.3 setOpcion()	111
4.34 src/platform/rpi/menu.c File Reference	111
4.34.1 Detailed Description	112
4.34.2 Function Documentation	112
4.34.2.1 getOpcion()	112
4.34.2.2 setMenu()	112
4.34.2.3 setOpcion()	112
4.35 src/platform/pc/nombre.c File Reference	113

4.35.1 Detailed Description	113
4.35.2 Function Documentation	114
4.35.2.1 devolverNombre()	114
4.36 src/platform/rpi/nombre.c File Reference	114
4.36.1 Detailed Description	115
4.36.2 Function Documentation	115
4.36.2.1 devolverNombre()	115
4.37 src/platform/pc/sound.c File Reference	115
4.37.1 Detailed Description	116
4.37.2 Function Documentation	116
4.37.2.1 iniciarSonido()	116
4.37.2.2 reproducirEfecto()	116
4.37.2.3 reproducirMusica()	117
4.38 src/platform/rpi/sound.c File Reference	117
4.38.1 Detailed Description	117
4.38.2 Function Documentation	118
4.38.2.1 iniciarSonido()	118
4.38.2.2 reproducirEfecto()	118
4.38.2.3 reproducirMusica()	118
4.39 src/platform/rpi/bitmap.h File Reference	118
4.39.1 Detailed Description	119
4.39.2 Function Documentation	119
4.39.2.1 copiarMatriz()	119
4.39.2.2 limpiarMatriz()	120
4.39.2.3 matrizAnd()	120
4.39.2.4 matrizNot()	120
4.39.2.5 matrizOr()	121
4.39.2.6 matrizXor()	121
4.39.2.7 printMatriz()	121
4.40 bitmap.h	121
4.41 disdrv.h	122
4.42 joydrv.h	123
4.43 src/platform/rpi/mensajes.c File Reference	123
4.43.1 Detailed Description	125
4.43.2 Function Documentation	125
4.43.2.1 borrarRenglon()	125
4.43.2.2 charAMatriz()	125
4.43.2.3 charARenglon()	126
4.43.2.4 concatenarLetraMensaje()	126
4.43.2.5 copiarMatrizRenglon()	126
4.43.2.6 copiarRenglon()	127
4.43.2.7 mensaje()	127

4.43.2.8 moverMensaje()	127
4.43.2.9 printRenglon()	128
4.43.2.10 reemplazarLetra()	128
4.43.2.11 reemplazarUltLetraMensaje()	128
4.43.2.12 renglonAnd()	128
4.43.2.13 renglonIzquierdoLibre()	129
4.43.2.14 renglonNot()	129
4.43.2.15 renglonOr()	129
4.43.2.16 renglonShiftDer()	130
4.43.2.17 renglonShiftIzq()	130
4.43.2.18 uintARenglon()	130
4.44 src/platform/rpi/mensajes.h File Reference	131
4.44.1 Detailed Description	132
4.44.2 Function Documentation	132
4.44.2.1 borrarRenglon()	132
4.44.2.2 charAMatriz()	133
4.44.2.3 charARenglon()	133
4.44.2.4 concatenarLetraMensaje()	133
4.44.2.5 copiarMatrizRenglon()	134
4.44.2.6 copiarRenglon()	134
4.44.2.7 mensaje()	134
4.44.2.8 moverMensaje()	135
4.44.2.9 printRenglon()	135
4.44.2.10 reemplazarLetra()	135
4.44.2.11 reemplazarUltLetraMensaje()	136
4.44.2.12 renglonAnd()	136
4.44.2.13 renglonIzquierdoLibre()	136
4.44.2.14 renglonNot()	137
4.44.2.15 renglonOr()	137
4.44.2.16 renglonShiftDer()	137
4.44.2.17 renglonShiftIzq()	138
4.44.2.18 uintARenglon()	138
4.45 mensajes.h	138
4.46 audio.h	139
4.47 src/queue.c File Reference	141
4.47.1 Detailed Description	142
4.47.2 Function Documentation	142
4.47.2.1 queueSiguienteEvento()	142
4.47.2.2 queueVacía()	142
4.48 src/queue.h File Reference	143
4.48.1 Detailed Description	143
4.48.2 Function Documentation	143

4.48.2.1 queueSiguienteEvento()	144
4.48.2.2 queueVacía()	144
4.49 queue.h	144
4.50 src/ranking.c File Reference	145
4.50.1 Detailed Description	146
4.50.2 Function Documentation	146
4.50.2.1 actualizarRanking()	146
4.50.2.2 getJugadorRankingPuntos()	146
4.50.2.3 getRankingLineas()	147
4.50.2.4 getRankingNombres()	147
4.50.2.5 getRankingPuntos()	147
4.50.2.6 verificarJugadorRanking()	147
4.51 src/ranking.h File Reference	148
4.51.1 Detailed Description	148
4.51.2 Function Documentation	149
4.51.2.1 actualizarRanking()	149
4.51.2.2 getJugadorRankingPuntos()	149
4.51.2.3 getRankingLineas()	149
4.51.2.4 getRankingNombres()	150
4.51.2.5 getRankingPuntos()	150
4.51.2.6 verificarJugadorRanking()	150
4.52 ranking.h	150
4.53 src/sound.h File Reference	151
4.53.1 Detailed Description	152
4.53.2 Function Documentation	152
4.53.2.1 iniciarSonido()	152
4.53.2.2 reproducirEfecto()	152
4.53.2.3 reproducirMusica()	153
4.54 sound.h	153

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

ALGIF_ANIMATION	5
ALGIF_BITMAP	5
ALGIF_FRAME	6
ALGIF_PALETTE	6
ALGIF_RGB	6
allegro_t	7
car_t	7
coin_t	7
data_t	8
dcoord_t	8
frog_t	8
jcoord_t	9
log_t	9
Mensaje	9
menu_t	10
nodeT	10
pair_xy_t	10
privateAudioDevice	10
renglon_t	11
sound	11
sounds_t	11
sprites_menu_t	12
sprites_t	12
Estructura principal de spritesheets	12
state_diagram_edge	13
turtle_pack_t	14
window_t	14

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

src/display.h	Header del modulo display Vinculo entre la fsm y las plataformas en lo que respecta a la visualizacion del juego	15
src/fsm.c	Source del modulo fsm. Administra la máquina de estados, siendo el engine del juego	18
src/fsm.h	Header del modulo fsm. Contiene los prototipos de funciones necesarias para iniciar la fsm desde main.c	23
src/game.h	Header del modulo game Vinculo entre la fsm y las plataformas en lo que respecta a la informacion del jugador y el progreso del juego	25
src/input.h	Header del modulo input Vinculo entre la fsm y las plataformas en lo que respecta al manejo de acciones externas	29
src/main.c	Archivo principal. Inicia y pone a correr la maquian de estados (fsm)	31
src/menu.h	31
src/nombre.h	Header del modulo genérico nombre. Prototipos de funciones de relacionadas al ingreso del nombre del jugador	32
src/queue.c	Source del modulo queue. Funciones para el manejo de la cola de eventos	141
src/queue.h	Header del modulo queue. Prototipos de funciones de interaccion con la cola de eventos . . .	143
src/ranking.c	Source del modulo ranking. Funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente,	145
src/ranking.h	Header del modulo ranking. Prototipos de funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente,	148
src/sound.h	Header del modulo sound Vinculo entre la fsm y las plataformas en lo que respecta al sonido .	151
src/platform/pc/allegro_stuff.c	38
src/platform/pc/allegro_stuff.h	47
src/platform/pc/display.c	58

src/platform/pc/entities.h	61
src/platform/pc/game.c	62
src/platform/pc/game_data.c	69
src/platform/pc/game_data.h	76
src/platform/pc/geometry.c	85
src/platform/pc/geometry.h	93
src/platform/pc/input.c	106
src/platform/pc/menu.c	108
src/platform/pc/nombre.c	113
src/platform/pc/sound.c	115
src/platform/pc/algif5/algif.h	
Header para el uso de la libreria algif (algif5 is a gif loading library for Allegro 5)	33
src/platform/rpi/bitmap.c	
Archivo para manejo de matrices 16x16	35
src/platform/rpi/bitmap.h	
Encabezado del archivo para manejo de matrices 16x16	118
src/platform/rpi/dsdrv.h	122
src/platform/rpi/game.c	
Archivo para manejar la información del juego	66
src/platform/rpi/input.c	
Archivo para manejo del joystick en RPI	107
src/platform/rpi/joydrv.h	123
src/platform/rpi/mensajes.c	
Permite codificar strings en formato renglon para mostrar en display	123
src/platform/rpi/mensajes.h	
Encabezado de mensajes, con definiciones sobre tipos de datos y funciones	131
src/platform/rpi/menu.c	
Archivo para manejo de los menús en RPI	111
src/platform/rpi/nombre.c	
Archivo para manejo de información en el ingreso del nombre	114
src/platform/rpi/sound.c	
Archivo para manejo del sonido en RPI	117
src/platform/rpi/simpleSDL2audio/audio.h	139

Chapter 3

Data Structure Documentation

3.1 ALGIF_ANIMATION Struct Reference

Data Fields

- int **width**
- int **height**
- int **frames_count**
- int **background_index**
- int **loop**
- [ALGIF_PALETTE](#) **palette**
- [ALGIF_FRAME](#) * **frames**
- int **duration**
- ALLEGRO_BITMAP * **store**

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

- [src/platform/pc/algif5/algif.h](#)

3.2 ALGIF_BITMAP Struct Reference

Data Fields

- int **w**
- int **h**
- uint8_t * **data**

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

- [src/platform/pc/algif5/algif.h](#)

3.3 ALGIF_FRAME Struct Reference

Data Fields

- [ALGIF_BITMAP](#) * **bitmap_8_bit**
- [ALGIF_PALETTE](#) **palette**
- int **xoff**
- int **yoff**
- int **duration**
- int **disposal_method**
- int **transparent_index**
- ALLEGRO_BITMAP * **rendered**

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

- [src/platform/pc/algif5/algif.h](#)

3.4 ALGIF_PALETTE Struct Reference

Data Fields

- int **colors_count**
- [ALGIF_RGB](#) **colors** [256]

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

- [src/platform/pc/algif5/algif.h](#)

3.5 ALGIF_RGB Struct Reference

Data Fields

- uint8_t **r**
- uint8_t **g**
- uint8_t **b**

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

- [src/platform/pc/algif5/algif.h](#)

3.6 allegro_t Struct Reference

Data Fields

- ALLEGRO_TIMER * **timer**
- ALLEGRO_EVENT_QUEUE * **queue**
- ALLEGRO_DISPLAY * **disp**
- ALLEGRO_FONT * **font**
- int **font_h**
- int **font_w**
- ALLEGRO_EVENT **event**
- bool **done**
- bool **redraw**

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

- src/platform/pc/[allegro_stuff.c](#)

3.7 car_t Struct Reference

Data Fields

- int **x**
- int **y**
- int **lane**
- int **dx**
- CAR_TYPE **type**
- int **length**
- int **count**
- bool **fast**
- bool **used**

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

- src/platform/pc/entities.c

3.8 coin_t Struct Reference

Data Fields

- int **x**
- int **y**
- bool **used**
-

```
struct {  
    unsigned int frame_cont  
    unsigned int timeout  
    unsigned int blink_timer  
    unsigned int cont  
    bool flag  
} fx
```

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

- src/platform/pc/entities.c

3.9 data_t Struct Reference

Data Fields

- int **lives**
- unsigned long long **score**
- unsigned long long **score_max**
- ```
struct {
 int number
 int time_left
 int time
 long time_ref
} run
```
- unsigned long **frames**
- int **timer\_in\_sec**
- int **difficulty**
- char **name** [MAX\_NAME\_CHAR]
- unsigned char **flag**
- bool **goals** [MAX\_GOALS]

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

- [src/platform/pc/game\\_data.c](#)

## 3.10 dcoord\_t Struct Reference

### Data Fields

- uint8\_t **x**
- uint8\_t **y**

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

- [src/platform/rpi/disdrv.h](#)

## 3.11 frog\_t Struct Reference

### Data Fields

- int **x**
- int **y**
- int **moving**
- int **facing**
- int **steps**
- unsigned char **state**
- unsigned char **next\_action**

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

- [src/platform/pc/entities.c](#)

## 3.12 jcoord\_t Struct Reference

### Data Fields

- int8\_t **x**
- int8\_t **y**

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

- `src/platform/rpi/joydrv.h`

## 3.13 log\_t Struct Reference

### Data Fields

- int **x**
- int **y**
- int **lane**
- int **dx**
- bool **used**

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

- `src/platform/pc/entities.c`

## 3.14 Mensaje Struct Reference

### Data Fields

- char **msj** [L\_MAX]
- int **posicion**
- int **index**
- int **longitud**
- int **j**
- bool **habilitacion**
- bool **mover\_texto**
- bool **repetir\_msj**
- [renglon\\_t](#) **renglon**

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

- `src/platform/rpi/mensajes.h`

### 3.15 menu\_t Struct Reference

#### Data Fields

- [window\\_t](#) **window** [MENU\_WINDOW\_MAX]
- int **actual\_window**

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

- [src/platform/pc/menu.c](#)

### 3.16 nodeT Struct Reference

#### Data Fields

- event\_t **data**
- struct [nodeT](#) \* **next**

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

- [src/queue.c](#)

### 3.17 pair\_xy\_t Struct Reference

#### Data Fields

- int **x**
- int **y**

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

- [src/platform/pc/geometry.h](#)

### 3.18 privateAudioDevice Struct Reference

#### Data Fields

- SDL\_AudioDeviceID **device**
- SDL\_AudioSpec **want**
- uint8\_t **audioEnabled**

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

- [src/platform/rpi/simpleSDL2audio/audio.c](#)

## 3.19 renglon\_t Union Reference

### Data Fields

- uint32\_t **completo**
- ```
struct {  
    uint16_t mitad_der  
    uint16_t mitad_izq  
};
```

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

- [src/platform/rpi/mensajes.h](#)

3.20 sound Struct Reference

Data Fields

- uint32_t **length**
- uint32_t **lengthTrue**
- uint8_t * **bufferTrue**
- uint8_t * **buffer**
- uint8_t **loop**
- uint8_t **fade**
- uint8_t **free**
- uint8_t **volume**
- SDL_AudioSpec **audio**
- struct [sound](#) * **next**

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

- [src/platform/rpi/simpleSDL2audio/audio.h](#)

3.21 sounds_t Struct Reference

Data Fields

- ALLEGRO_AUDIO_STREAM * **stream**
- unsigned char **stream_state**
-

```

struct {
    ALLEGRO_SAMPLE * jump
    ALLEGRO_SAMPLE * crash
    ALLEGRO_SAMPLE * goal
    ALLEGRO_SAMPLE * low_time
    ALLEGRO_SAMPLE * click
    ALLEGRO_SAMPLE * bonus
    ALLEGRO_SAMPLE * run_completed
    ALLEGRO_SAMPLE * drowned
    ALLEGRO_SAMPLE * menu_enter
    ALLEGRO_SAMPLE * new_max_score
    ALLEGRO_SAMPLE * exiting
    ALLEGRO_SAMPLE * no_time
    ALLEGRO_SAMPLE * coin_drop
} samples

```

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

- `src/platform/pc/allegro_stuff.c`

3.22 `sprites_menu_t` Struct Reference

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

- `src/platform/pc/allegro_stuff.h`

3.23 `sprites_t` Struct Reference

Estructura principal de spritesheets.

```
#include <allegro_stuff.h>
```

Data Fields

- ALLEGRO_BITMAP * **frog_uncut**
- ALLEGRO_BITMAP * **frog** [8]
- ALLEGRO_BITMAP * **background**
- ALLEGRO_BITMAP * **log**
- ALLEGRO_BITMAP * **cars_uncut**
- ALLEGRO_BITMAP * **car** [CAR_TYPE_N]
- ALLEGRO_BITMAP * **turtle_uncut**
- ALLEGRO_BITMAP * **turtle** [TURTLE_FRAMES]
- ALLEGRO_BITMAP * **heart**
-


```
struct {
    ALLEGRO_BITMAP * uncut
    ALLEGRO_BITMAP * option [MENU_STATE_MAX]
    ALLEGRO_BITMAP * background
} menu [MENU_WINDOW_MAX]
```

- ALLEGRO_BITMAP * **credits**
- ALLEGRO_BITMAP * **name**
- ALLEGRO_BITMAP * **icon**
- ALLEGRO_BITMAP * **dead**
-

```
struct {
    ALLEGRO_BITMAP * uncut
    ALLEGRO_BITMAP * frame [SPRITE_COIN_FRAMES]
} coin
```

-

```
struct {
    ALLEGRO_BITMAP * uncut
    ALLEGRO_BITMAP * frame [SPRITE_SPLASH_FRAMES]
} splash
```

- ALLEGRO_BITMAP * **border**

3.23.1 Detailed Description

Estructura principal de spritesheets.

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

- [src/platform/pc/allegro_stuff.h](#)

3.24 state_diagram_edge Struct Reference

Data Fields

- `event_t` **evento**
- `STATE` * **proximo_estado**
- `void(* p_rut_accion)(void)`

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

- [src/fsm.c](#)

3.25 turtle_pack_t Struct Reference

Data Fields

- int **x**
- int **y**
- int **lane**
- int **dx**
- bool **used**
- unsigned char **turtles_in_pack**
-

```
struct {  
    unsigned char frame  
    unsigned int timeout  
    unsigned int cont  
} fx
```

- int **wide**
- unsigned char **state**

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

- [src/platform/pc/entities.c](#)

3.26 window_t Struct Reference

Data Fields

- int **actual_state**
- int **max_states**

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

- [src/platform/pc/menu.c](#)

Chapter 4

File Documentation

4.1 src/display.h File Reference

Header del modulo display Vinculo entre la fsm y las plataformas en lo que respecta a la visualizacion del juego.

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

Enumerations

- enum **posiciones_mensajes** {
 POS_MSJ_MENU , **POS_MSJ_DIFICULTAD** , **POS_MSJ_RANKING** , **POS_MSJ_NOMBRE** ,
 POS_MSJ_PASAR , **POS_MSJ_PAUSA** , **POS_MSJ_NEW_HI_SCORE** , **POS_MSJ_GAME_OVER** ,
 POS_OPCION , **POS_RANKING_2** , **POS_CREDITOS** }

Functions

- bool **iniciarDisplay** ()
 Inicializa el display de la plataforma.
- void **actualizarDisplay** ()
 Actualiza el display de la plataforma.
- void **limpiarDisplay** ()
 Limpia el display de la plataforma.
- void **mostrarTexto** (char *txt, int pos)
 Muestra un texto dado en una posicion dada (retiene el flujo)
- void **dejarTexto** (char *txt, int pos, bool repetir)
 Deja el texto en la posición data (no retiene)
- void **cargarRanking** (void)
 Inicia muestreo de ranking en la plataforma.
- void **mostrarRanking** (void)
 Bucle que muestra el ranking. Finaliza desde dentro, y hace que finalice el thread de ranking.
- void **cargarCreditos** (void)
 Inicializa los cretidos en la plataforma.
- void **mostrarCreditos** (void)
 Bucle que muestra los creditos. Finaliza desde dentro, y hace que finalice el thread de ranking.
- void **reconfigurarDisplayON** (void)
 Reconfigura el display de la plataforma y lo habilita.
- void **reconfigurarDisplayOFF** (void)
 Reconfigura el display de la plataforma y lo deshabilita.

4.1.1 Detailed Description

Header del modulo display Vinculo entre la fsm y las plataformas en lo que respecta a la visualizacion del juego.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.1.2 Function Documentation

4.1.2.1 cargarRanking()

```
void cargarRanking (
    void )
```

Inicia muestreo de ranking en la plataforma.

Parameters

<i>txt</i>	
------------	--

4.1.2.2 dejarTexto()

```
void dejarTexto (
    char * txt,
    int pos,
    bool repetir )
```

Deja el texto en la posición data (no retiene)

Parameters

<i>txt</i>	
<i>pos</i>	
<i>repetir</i>	

4.1.2.3 iniciarDisplay()

```
bool iniciarDisplay ( )
```

Inicializa el display de la plataforma.

Returns

true Exito
false Error

4.1.2.4 mostrarCreditos()

```
void mostrarCreditos (
    void )
```

Bucle que muestra los creditos. Finaliza desde dentro, y hace que finalice el thread de ranking.

Returns

true No finaliz
false Finaliza

4.1.2.5 mostrarTexto()

```
void mostrarTexto (
    char * txt,
    int pos )
```

Muestra un texto dado en una posicion dada (retiene el flujo)

Parameters

<i>txt</i>	Texto
<i>pos</i>	Posicion

4.2 display.h

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

```
1
13 #ifndef _DISPLAY_H_
14 #define _DISPLAY_H_
15
16 /*****
```

```

17  * INCLUDE HEADER FILES
18  *****/
19
20 #include <stdbool.h>
21
22 /*****
23  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
24  *****/
25
26 // Posiciones de mensajes
27 enum posiciones_mensajes
28 {
29     POS_MSJ_MENU,
30     POS_MSJ_DIFICULTAD,
31     POS_MSJ_RANKING,
32     POS_MSJ_NOMBRE,
33     POS_MSJ_PASAR,
34     POS_MSJ_PAUSA,
35     POS_MSJ_NEW_HI_SCORE,
36     POS_MSJ_GAME_OVER,
37     POS_OPCION,
38     POS_RANKING_2,
39     POS_CREDITOS
40 };
41
42 /*****
43  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
44  *****/
45
46 bool iniciarDisplay();
47
48 void actualizarDisplay();
49
50 void limpiarDisplay();
51
52 void mostrarTexto(char *txt, int pos);
53
54 void dejarTexto(char *txt, int pos, bool repetir);
55
56 void cargarRanking(void);
57
58 void mostrarRanking(void);
59
60 void cargarCreditos(void);
61
62 void mostrarCreditos(void);
63
64 void reconfigurarDisplayON(void);
65
66 void reconfigurarDisplayOFF(void);
67
68 /*****
69  *****/
70
71 #endif // _DISPLAY_H_

```

4.3 src/fsm.c File Reference

Source del modulo fsm. Administra la máquina de estados, siendo el engine del juego.

```

#include "fsm.h"
#include "display.h"
#include "game.h"
#include "menu.h"
#include "input.h"
#include "nombre.h"
#include "sound.h"
#include "ranking.h"
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <pthread.h>
#include <unistd.h>

```

Data Structures

- struct [state_diagram_edge](#)

Macros

- #define **FIN_TABLA** 0xFF
- #define **CTE_OPCION** 100
- #define **FIX_CPU_USAGE_SLEEP_US** 500

Typedefs

- typedef struct [state_diagram_edge](#) **STATE**

Functions

- bool [inicializarFsm](#) (void)
Inicializa la FSM, notificando si tuvo exito.
- void [fsm](#) (event_t evento_actual)
Intérprete de la FSM. Se encarga de hacer correr la FSM a partir de un estado y evento dados.
- void [fixHighCpuUsage](#) (void)
Fixea consumo elevado de cpu en el while loop principal.

Variables

- [STATE en_menu_ppal](#) []
- [STATE menu_ppal_esperando_opcion](#) []
- [STATE seleccionando_dificultad](#) []
- [STATE viendo_ranking](#) []
- [STATE viendo_creditos](#) []
- [STATE poniendo_nombre](#) []
- [STATE jugando](#) []
- [STATE en_pausa](#) []
- [STATE en_pausa_esperando_opcion](#) []
- [STATE en_game_over](#) []
- [STATE en_game_over_esperando_opcion](#) []

4.3.1 Detailed Description

Source del modulo fsm. Administra la máquina de estados, siendo el engine del juego.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.3.2 Function Documentation

4.3.2.1 fixHighCpuUsage()

```
void fixHighCpuUsage (
    void )
```

Fixea consumo elevado de cpu en el while loop principal.

Not the best solucion, but sirve...

<https://softwareengineering.stackexchange.com/questions/256524/infinite-while-loop-cpu->

4.3.2.2 fsm()

```
void fsm (
    event_t evento_actual )
```

Intérprete de la FSM. Se encarga de hacer correr la FSM a partir de un estado y evento dados.

Parameters

<i>p_tabla_estado</i>	Estado actual.
<i>evento_actual</i>	Evento recibido.

4.3.2.3 inicializarFsm()

```
bool inicializarFsm (
    void )
```

Inicializa la FSM, notificando si tuvo exito.

Returns

true Exito
false Error

4.3.3 Variable Documentation

4.3.3.1 en_game_over

STATE en_game_over

Initial value:

```
=  
{  
    {ENTER, en_game_over_esperando_opcion, procesar_enter_menu},  
    {ARRIBA, en_game_over, subirOpcion},  
    {ABAJO, en_game_over, bajarOpcion},  
    {FORCE_SALIR, NULL, salir_del_juego},  
    {FIN_TABLA, en_game_over, do_nothing}}
```

4.3.3.2 en_game_over_esperando_opcion

STATE en_game_over_esperando_opcion

Initial value:

```
=  
{  
    {CTE_OPCION, jugando, iniciar_juego},  
    {CTE_OPCION + 1, en_menu_ppal, ir_a_menu_ppal},  
    {FIN_TABLA, en_game_over_esperando_opcion, do_nothing}}
```

4.3.3.3 en_menu_ppal

STATE en_menu_ppal

Initial value:

```
=  
{  
    {ENTER, menu_ppal_esperando_opcion, procesar_enter_menu},  
    {ARRIBA, en_menu_ppal, subirOpcion},  
    {ABAJO, en_menu_ppal, bajarOpcion},  
    {FORCE_SALIR, NULL, salir_del_juego},  
    {FIN_TABLA, en_menu_ppal, do_nothing}}
```

4.3.3.4 en_pausa

STATE en_pausa

Initial value:

```
=  
{  
    {ENTER, en_pausa_esperando_opcion, procesar_enter_menu},  
    {ARRIBA, en_pausa, subirOpcion},  
    {ABAJO, en_pausa, bajarOpcion},  
    {FORCE_SALIR, NULL, salir_del_juego},  
    {FIN_TABLA, en_pausa, do_nothing}}
```

4.3.3.5 en_pausa_esperando_opcion

STATE en_pausa_esperando_opcion

Initial value:

```
=
{
    {CTE_OPCION, jugando, continuar},
    {CTE_OPCION + 1, jugando, iniciar_juego},
    {CTE_OPCION + 2, en_menu_ppal, ir_a_menu_ppal},
    {FIN_TABLA, en_pausa_esperando_opcion, do_nothing}}
```

4.3.3.6 jugando

STATE jugando

Initial value:

```
=
{
    {ENTER, en_pausa, pausar},
    {GAME_OVER, en_game_over, procesar_game_over},
    {ARRIBA, jugando, moverAdelante},
    {ABAJO, jugando, moverAtras},
    {IZDA, jugando, moverIzda},
    {DCHA, jugando, moverDcha},
    {FORCE_SALIR, NULL, salir_del_juego},
    {FIN_TABLA, jugando, do_nothing}}
```

4.3.3.7 menu_ppal_esperando_opcion

STATE menu_ppal_esperando_opcion

Initial value:

```
=
{
    {CTE_OPCION, poniendo_nombre, ir_a_poniendo_nombre},
    {CTE_OPCION + 1, seleccionando_dificultad, ir_a_seleccionando_dificultad},
    {CTE_OPCION + 2, viendo_ranking, ir_a_viendo_ranking},
    {CTE_OPCION + 3, viendo_creditos, ir_a_viendo_creditos},
    {CTE_OPCION + 4, NULL, salir_del_juego},
    {FIN_TABLA, menu_ppal_esperando_opcion, do_nothing}
}
```

4.3.3.8 poniendo_nombre

STATE poniendo_nombre

Initial value:

```
=
{
    {ESC, en_menu_ppal, ir_a_menu_ppal},
    {ENTER, jugando, iniciar_juego},
    {ARRIBA, poniendo_nombre, subirLetra},
    {ABAJO, poniendo_nombre, bajarLetra},
    {DCHA, poniendo_nombre, siguienteLetra},
    {FORCE_SALIR, NULL, salir_del_juego},
    {FIN_TABLA, poniendo_nombre, agregarLetra}
}
```

4.3.3.9 seleccionando_dificultad

STATE seleccionando_dificultad

Initial value:

```
=
{
    {ENTER, en_menu_ppal, procesar_enter_dificultad},
    {ARRIBA, seleccionando_dificultad, subirOpcion},
    {ABAJO, seleccionando_dificultad, bajarOpcion},
    {FORCE_SALIR, NULL, salir_del_juego},
    {FIN_TABLA, seleccionando_dificultad, do_nothing}}
```

4.3.3.10 viendo_creditos

STATE viendo_creditos

Initial value:

```
=
{
    {ENTER, en_menu_ppal, procesar_enter_creditos},
    {FORCE_SALIR, NULL, salir_del_juego},
    {FIN_TABLA, viendo_creditos, do_nothing}}
```

4.3.3.11 viendo_ranking

STATE viendo_ranking

Initial value:

```
=
{
    {ENTER, en_menu_ppal, procesar_enter_ranking},
    {FORCE_SALIR, NULL, salir_del_juego},
    {FIN_TABLA, viendo_ranking, do_nothing}}
```

4.4 src/fsm.h File Reference

Header del modulo fsm. Contiene los prototipos de funciones necesarias para iniciar la fsm desde [main.c](#).

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

Functions

- bool [inicializarFsm](#) (void)
Inicializa la FSM, notificando si tuvo exito.
- void [fsm](#) (event_t evento_actual)
Intérprete de la FSM. Se encarga de hacer correr la FSM a partir de un estado y evento dados.
- void [fixHighCpuUsage](#) (void)
Fixea consumo elevado de cpu en el while loop principal.

4.4.1 Detailed Description

Header del modulo fsm. Contiene los prototipos de funciones necesarias para iniciar la fsm desde [main.c](#).

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.4.2 Function Documentation

4.4.2.1 fixHighCpuUsage()

```
void fixHighCpuUsage (  
    void )
```

Fixea consumo elevado de cpu en el while loop principal.

Not the best solution, but sirve...

<https://softwareengineering.stackexchange.com/questions/256524/infinite-while-loop-cpu->

4.4.2.2 fsm()

```
void fsm (  
    event_t evento_actual )
```

Intérprete de la FSM. Se encarga de hacer correr la FSM a partir de un estado y evento dados.

Parameters

<i>p_tabla_estado</i>	Estado actual.
<i>evento_actual</i>	Evento recibido.

4.4.2.3 inicializarFsm()

```
bool inicializarFsm (  
    void )
```

Inicializa la FSM, notificando si tuvo exito.

Returns

true Exito
false Error

4.5 fsm.h

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

```

1
13 #ifndef _FSM_H_
14 #define _FSM_H_
15
16 /*****
17  * INCLUDE HEADER FILES
18  *****/
19
20 #include "queue.h"
21 #include <stdbool.h>
22
23 /*****
24  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
25  *****/
26
33 bool inicializarFsm(void);
34
41 void fsm(event_t evento_actual);
42
51 void fixHighCpuUsage(void);
52
53 /*****
54  *****/
55
56 #endif // _FSM_H_

```

4.6 src/game.h File Reference

Header del modulo game Vinculo entre la fsm y las plataformas en lo que respecta a la informacion del jugador y el progreso del juego.

```

#include <stdbool.h>
#include <stdlib.h>
#include <stdint.h>

```

Functions

- void [setNombre](#) (char *nombre)
Confirma el nombre del jugador.
- void [setMaxPuntos](#) (unsigned long long max)
Setea los puntos maximos del jugador.
- void [setDificultad](#) (int dif)
Setea la dificultad a usar.
- bool [tiempoRefrescoEntidades](#) (void)
Chequea si es tiempo de refrescar entidades según la plataforma.
- char * [getNombre](#) (void)
Devuelve el nombre del jugador.
- unsigned long long [getPuntos](#) (void)
Devuelve el puntaje del jugador.
- unsigned long long [getMaxPuntos](#) (void)

- Devuelve el puntaje máximo del jugador.*
 - int **getNivel** (void)
- Devuelve el nivel/run del jugador.*
 - void **inicializarJuego** (void)
- Rutina que se ejecuta al comenzar a jugar, luego de ingresar el nombre.*
 - void **reiniciarNivel** (void)
- Configuraciones para reiniciar el nivel.*
 - void **pausarJuego** (void)
- Pausa el juego.*
 - void **reanudarJuego** (void)
- Saca el juego de pausa.*
 - void **refrescar** (void)
- Actualizaciones relativas a actualizar las entidades.*
 - void **moverAdelante** (void)
- Avanza el jugador.*
 - void **moverAtras** (void)
- Retrocede el jugador.*
 - void **moverIzda** (void)
- Mueve el jugador a la izquierda.*
 - void **moverDcha** (void)
- Mueve el jugador a la derecha.*
 - void **respawn** (void)
- Respawnea el jugador.*
 - void **perderVida** (void)
- Resta una vida.*
 - void **actualizarInterfaz** (void)
- Actualizaciones relativas a lo visual.*

4.6.1 Detailed Description

Header del modulo game Vinculo entre la fsm y las plataformas en lo que respecta a la informacion del jugador y el progreso del juego.

Header del modulo genérico menu. Prototipos de funciones de interaccion con el menu del juego.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.6.2 Function Documentation

4.6.2.1 getMaxPuntos()

```
unsigned long long getMaxPuntos (
    void )
```

Devuelve el puntaje máximo del jugador.

Returns

unsigned long long

4.6.2.2 getNivel()

```
int getNivel (
    void )
```

Devuelve el nivel//run del jugador.

Returns

int

4.6.2.3 getNombre()

```
char * getNombre (
    void )
```

Devuelve el nombre del jugador.

Returns

char*

4.6.2.4 getPuntos()

```
unsigned long long getPuntos (
    void )
```

Devuelve el puntaje del jugador.

Returns

unsigned long long

4.6.2.5 setDificultad()

```
void setDificultad (
    int dif )
```

Setea la dificultad a usar.

Parameters

<i>dif</i>	
------------	--

4.6.2.6 setMaxPuntos()

```
void setMaxPuntos (
    unsigned long long max )
```

Setea los puntos maximos del jugador.

Parameters

<i>max</i>	
------------	--

4.6.2.7 setNombre()

```
void setNombre (
    char * nombre )
```

Confirma el nombre del jugador.

Parameters

<i>nombre</i>	
---------------	--

4.6.2.8 tiempoRefrescoEntidades()

```
bool tiempoRefrescoEntidades (
    void )
```

Chequea si es tiempo de refrescar entidades según la plataforma.

Returns

true

false

4.7 game.h

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

```

1
13 #ifndef _GAME_H_
14 #define _GAME_H_
15
16 /*****
17  * INCLUDE HEADER FILES
18  *****/
19
20 #include <stdbool.h>
21 #include <stdlib.h>
22 #include <stdint.h>
23
24
25 /*****
26  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
27  *****/
28
34 void setNombre(char* nombre);
35
36
42 void setMaxPuntos(unsigned long long max);
43
49 void setDificultad(int dif);
50
57 bool tiempoRefrescoEntidades(void);
58
64 char* getNombre(void);
65
71 unsigned long long getPuntos(void);
72
78 unsigned long long getMaxPuntos(void);
79
85 int getNivel(void);
86
91 void inicializarJuego(void);
92
97 void reiniciarNivel(void);
98
103 void pausarJuego(void);
104
109 void reanudarJuego(void);
110
115 void refrescar(void);
116
121 void moverAdelante(void);
122
127 void moverAtras(void);
128
133 void moverIzda(void);
134
139 void moverDcha(void);
140
145 void respawn(void);
146
151 void perderVida(void);
152
157 void actualizarInterfaz(void);
158
159
160 /*****
161  *****/
162
163 #endif // _GAME_H_

```

4.8 src/input.h File Reference

Header del modulo input Vinculo entre la fsm y las plataformas en lo que respecta al manejo de acciones externas.

```

#include "queue.h"
#include <stdbool.h>

```

Functions

- void **iniciarEntradas** (void)
Inicializa las entradas de la plataforma.
- event_t **leerEntradas** (void)
Devuelve una entrada válida.

4.8.1 Detailed Description

Header del modulo input Vínculo entre la fsm y las plataformas en lo que respecta al manejo de acciones externas.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.8.2 Function Documentation

4.8.2.1 leerEntradas()

```
event_t leerEntradas (  
    void )
```

Devuelve una entrada válida.

Returns

event_t enum eventos_tecla

4.9 input.h

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

```
1  
13 #ifndef _INPUT_H_  
14 #define _INPUT_H_  
15  
16 /*****  
17  * INCLUDE HEADER FILES  
18  *****/  
19  
20 #include "queue.h"  
21  
22 #include <stdbool.h>  
23  
24  
25 /*****  
26  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE  
27  *****/  
28  
34 void iniciarEntradas(void);  
35  
41 event_t leerEntradas(void);  
42  
43  
44 /*****  
45  *****/  
46  
47 #endif // _INPUT_H_
```

4.10 src/main.c File Reference

Archivo principal. Inicia y pone a correr la maquina de estados (fsm).

```
#include "fsm.h"
#include "queue.h"
#include <stdio.h>
#include <stdlib.h>
```

Functions

- int **main** (void)

4.10.1 Detailed Description

Archivo principal. Inicia y pone a correr la maquina de estados (fsm).

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.11 menu.h

```
1
12 #ifndef _MENU_H_
13 #define _MENU_H_
14
15 /*****
16  * INCLUDE HEADER FILES
17  *****/
18
19 #include "queue.h"
20
21
22 /*****
23  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
24  *****/
25
26 enum textos_menu
27 {
28     JUGAR = 0,
29     DIFICULTAD,
30     RANKING,
31     CREDITOS,
32     SALIRTXT,
33     CONTINUAR,
34     REINICIAR,
35     FACIL,
36     NORMAL,
37     DIFICIL
38 };
39
40
41 /*****
42  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
43  *****/
44
45 void iniciarMenu(void);
46
```

```

55 void destruirMenu(void);
56
63 void setMenu(int* a, unsigned int size);
64
70 void setOpcion(int opc);
71
77 int getOpcion(void);
78
83 void subirOpcion(void);
84
89 void bajarOpcion(void);
90
91 /*****
92 *****/
93
94 #endif // _MENU_H_

```

4.12 src/nombre.h File Reference

Header del modulo genérico nombre. Prototipos de funciones de relacionadas al ingreso del nombre del jugador.

Functions

- void **nuevoNombre** (void)
Se ejecuta una vez al ingresar a poner un nuevo nombre.
- void **subirLetra** (void)
Selecciona la siguiente letra superior.
- void **bajarLetra** (void)
Selecciona la letra inferior.
- void **siguienteLetra** (void)
Confirma la letra y pasa a seleccionar la siguiente.
- void **agregarLetra** (void)
Confirma la letra.
- void **subirNombre** (void)
- char * **devolverNombre** (void)
Devuelve puntero al nombre.

4.12.1 Detailed Description

Header del modulo genérico nombre. Prototipos de funciones de relacionadas al ingreso del nombre del jugador.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.12.2 Function Documentation

4.12.2.1 devolverNombre()

```
char * devolverNombre (
    void )
```

Devuelve puntero al nombre.

Returns

char* Puntero al nombre

4.13 nombre.h

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

```
1
13 #ifndef _NOMBRE_H_
14 #define _NOMBRE_H_
15
16
17 /*****
18  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
19  *****/
20
25 void nuevoNombre(void);
26
31 void subirLetra(void);
32
37 void bajarLetra(void);
38
43 void siguienteLetra(void);
44
49 void agregarLetra(void);
50
55 void subirNombre(void);
56
62 char* devolverNombre(void);
63
64
65 /*****
66  *****/
67
68 #endif // _NOMBRE_H_
```

4.14 src/platform/pc/algif5/algif.h File Reference

Header para el uso de la libreria algif (algif5 is a gif loading library for Allegro 5)

```
#include <allegro5/allegro5.h>
```

Data Structures

- struct [ALGIF_RGB](#)
- struct [ALGIF_PALETTE](#)
- struct [ALGIF_BITMAP](#)
- struct [ALGIF_ANIMATION](#)
- struct [ALGIF_FRAME](#)

Typedefs

- typedef struct [ALGIF_ANIMATION](#) **ALGIF_ANIMATION**
- typedef struct [ALGIF_FRAME](#) **ALGIF_FRAME**
- typedef struct [ALGIF_PALETTE](#) **ALGIF_PALETTE**
- typedef struct [ALGIF_BITMAP](#) **ALGIF_BITMAP**
- typedef struct [ALGIF_RGB](#) **ALGIF_RGB**

Functions

- [ALGIF_ANIMATION](#) * **algif_load_raw** (ALLEGRO_FILE *file)
- [ALGIF_ANIMATION](#) * **algif_load_animation_f** (ALLEGRO_FILE *file)
- [ALGIF_ANIMATION](#) * **algif_load_animation** (char const *filename)
- void **algif_render_frame** ([ALGIF_ANIMATION](#) *gif, int frame, int xpos, int ypos)
- void **algif_destroy_animation** ([ALGIF_ANIMATION](#) *gif)
- [ALGIF_BITMAP](#) * **algif_create_bitmap** (int w, int h)
- void **algif_destroy_bitmap** ([ALGIF_BITMAP](#) *bitmap)
- void **algif_blit** ([ALGIF_BITMAP](#) *from, [ALGIF_BITMAP](#) *to, int xf, int yf, int xt, int yt, int w, int h)
- ALLEGRO_BITMAP * **algif_get_bitmap** ([ALGIF_ANIMATION](#) *gif, double seconds)
- ALLEGRO_BITMAP * **algif_get_frame_bitmap** ([ALGIF_ANIMATION](#) *gif, int i)
- double **algif_get_frame_duration** ([ALGIF_ANIMATION](#) *gif, int i)

4.14.1 Detailed Description

Header para el uso de la libreria algif (algif5 is a gif loading library for Allegro 5)

Author

allefant (<https://github.com/allefant>)
<https://github.com/allefant/algif5>

4.15 algif.h

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

```

1
10 #ifndef _GIF_H_
11 #define _GIF_H_
12
13 #include <allegro5/allegro5.h>
14
15 #ifdef __cplusplus
16 extern "C"
17 {
18 #endif
19
20     typedef struct ALGIF_ANIMATION ALGIF_ANIMATION;
21     typedef struct ALGIF_FRAME ALGIF_FRAME;
22     typedef struct ALGIF_PALETTE ALGIF_PALETTE;
23     typedef struct ALGIF_BITMAP ALGIF_BITMAP;
24     typedef struct ALGIF_RGB ALGIF_RGB;
25
26     struct ALGIF_RGB
27     {
28         uint8_t r, g, b;
29     };
30
31     struct ALGIF_PALETTE
32     {
33         int colors_count;
34         ALGIF_RGB colors[256];

```

```

35     };
36
37     struct ALGIF_BITMAP
38     {
39         int w, h;
40         uint8_t *data;
41     };
42
43     struct ALGIF_ANIMATION
44     {
45         int width, height;
46         int frames_count;
47         int background_index;
48         int loop; /* -1 = no, 0 = forever, 1..65535 = that many times */
49         ALGIF_PALETTE palette;
50         ALGIF_FRAME *frames;
51
52         int duration;
53         ALLEGRO_BITMAP *store;
54     };
55
56     struct ALGIF_FRAME
57     {
58         ALGIF_BITMAP *bitmap_8_bit;
59         ALGIF_PALETTE palette;
60         int xoff, yoff;
61         int duration; /* in 1/100th seconds */
62         int disposal_method; /* 0 = don't care, 1 = keep, 2 = background, 3 = previous */
63         int transparent_index;
64
65         ALLEGRO_BITMAP *rendered;
66     };
67
68     ALGIF_ANIMATION *algif_load_raw(ALLEGRO_FILE *file);
69     ALGIF_ANIMATION *algif_load_animation_f(ALLEGRO_FILE *file);
70     ALGIF_ANIMATION *algif_load_animation(char const *filename);
71     void algif_render_frame(ALGIF_ANIMATION *gif, int frame, int xpos, int ypos);
72     void algif_destroy_animation(ALGIF_ANIMATION *gif);
73
74     ALGIF_BITMAP *algif_create_bitmap(int w, int h);
75     void algif_destroy_bitmap(ALGIF_BITMAP *bitmap);
76     void algif_blit(ALGIF_BITMAP *from, ALGIF_BITMAP *to, int xf, int yf, int xt, int yt,
77                    int w, int h);
78     ALLEGRO_BITMAP *algif_get_bitmap(ALGIF_ANIMATION *gif, double seconds);
79     ALLEGRO_BITMAP *algif_get_frame_bitmap(ALGIF_ANIMATION *gif, int i);
80     double algif_get_frame_duration(ALGIF_ANIMATION *gif, int i);
81
82 #ifdef __cplusplus
83 }
84 #endif
85
86 #endif // _GIF_H_

```

4.16 src/platform/rpi/bitmap.c File Reference

Archivo para manejo de matrices 16x16.

```

#include "bitmap.h"
#include <stdio.h>

```

Functions

- void [printMatriz](#) (matriz_t a)
Imprime una matriz en consola (para debug)
- void [limpiarMatriz](#) (matriz_t a)
Borra el contenido de una matriz.
- void [copiarMatriz](#) (matriz_t destino, const matriz_t desde)
Copia el contenido de una matriz en otra.
- void [matrizAnd](#) (matriz_t a, matriz_t b)

- Dadas dos matrices A y B , se hará la operación " $A \&= B$ ".*
- void `matrizOr` (`matriz_t a`, `matriz_t b`)
- Dadas dos matrices A y B , se hará la operación " $A |= B$ ".*
- void `matrizXor` (`matriz_t a`, `matriz_t b`)
- Dadas dos matrices A y B , se hará la operación " $A \wedge= B$ ".*
- void `matrizNot` (`matriz_t a`)
- Dadas una matriz A , se hará la operación " $A = \sim A$ ".*

4.16.1 Detailed Description

Archivo para manejo de matrices 16x16.

Archivo para manejo del display de RPI.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.16.2 Function Documentation

4.16.2.1 `copiarMatriz()`

```
void copiarMatriz (
    matriz_t destino,
    const matriz_t desde )
```

Copia el contenido de una matriz en otra.

Parameters

<i>destino</i>	
<i>desde</i>	

4.16.2.2 `limpiarMatriz()`

```
void limpiarMatriz (
    matriz_t A )
```

Borra el contenido de una matriz.

Parameters

<i>A</i>	
----------	--

4.16.2.3 matrizAnd()

```
void matrizAnd (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación " $A \&= B$ ".

Parameters

<i>A</i>	
<i>B</i>	

4.16.2.4 matrizNot()

```
void matrizNot (
    matriz_t A )
```

Dadas una matriz A, se hará la operación " $A = \sim A$ ".

Parameters

<i>A</i>	
----------	--

4.16.2.5 matrizOr()

```
void matrizOr (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación " $A |= B$ ".

Parameters

<i>A</i>	
<i>B</i>	

4.16.2.6 matrizXor()

```
void matrizXor (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación "A ^= B".

Parameters

A	
B	

4.16.2.7 printMatriz()

```
void printMatriz (
    matriz_t A )
```

Imprime una matriz en consola (para debug)

Parameters

A	
---	--

4.17 src/platform/pc/allegro_stuff.c File Reference

```
#include "allegro_stuff.h"
#include "geometry.h"
#include <string.h>
#include "../algif5/algif.h"
```

Data Structures

- struct [allegro_t](#)
- struct [sounds_t](#)

Macros

- #define **FONT_HEIGHT** 16
- #define **SOUND_STREAM_FILE_CREDITS** "credits_theme"
- #define **SOUND_STREAM_FILE_MAIN** "main_menu_theme"

- #define **SOUND_STREAM_FILE_PAUSE** "pause_menu_theme"
- #define **SOUND_STREAM_FILE_PLAYING** "playing_theme"
- #define **SOUND_STREAM_FILE_RANKING** "ranking_theme"
- #define **SOUND_STREAM_FILE_RICK** "rick"
- #define **SOUND_STREAM_FILE_GAME_OVER** "game_over"
- #define **FONT_FILE_NAME** "PublicPixel.ttf"
- #define **SPRITE_HEART** "minecraft_heart"
- #define **SPRITE_BACKGROUND** "sprite_background"
- #define **SPRITE_CAR** "sprite_cars"
- #define **SPRITE_FROG** "sprite_frog"
- #define **SPRITE_LOG** "sprite_log"
- #define **SPRITE_TURTLES** "sprite_turtles"
- #define **SPRITE_MENU_HOME_BACK** "sprite_menu_home_background"
- #define **SPRITE_MENU_HOME** "sprite_menu_home"
- #define **SPRITE_MENU_DIFF_BACK** "sprite_menu_diff_background"
- #define **SPRITE_MENU_DIFF** "sprite_menu_diff"
- #define **SPRITE_MENU_PAUSE_BACK** "sprite_menu_pause_background"
- #define **SPRITE_MENU_PAUSE** "sprite_menu_pause"
- #define **SPRITE_MENU_GAME_OVER_BACK** "sprite_menu_gameover_background"
- #define **SPRITE_MENU_GAME_OVER** "sprite_menu_gameover"
- #define **SPRITE_CREDITS** "sprite_credits"
- #define **SPRITE_NAME** "sprite_name"
- #define **SPRITE_ICON** "icon"
- #define **SPRITE_DEAD** "sprite_dead"
- #define **SPRITE_BORDER** "sprite_border"
- #define **SPRITE_SPLASH** "sprite_splash"
- #define **SPRITE_COIN** "sprite_coin"
- #define **EXTENSION_SOUND_SAMPLE** ".wav"
- #define **EXTENSION_SOUND_STREAM** ".opus"
- #define **EXTENSION_SPRITES** ".png"
- #define **PATH_SOUND_STREAMS** "../res/sounds/streams/"
- #define **PATH_SOUND_SAMPLES** "../res/sounds/samples/"
- #define **PATH_FONTS** "../res/fonts/"
- #define **PATH_SPRITES** "../res/sprites/"
- #define **PATH_GIFS** "../res/gifs/"
- #define **GLOBAL_STREAM_VOLUME** (double)0.5

Enumerations

- enum **SOUND_STREAM_STATES** { **SOUND_STREAM_STATE_NO_INIT** , **SOUND_STREAM_STATE_↵**
INIT , **SOUND_STREAM_STATE_PAUSE** , **SOUND_STREAM_STATE_PLAY** }

Functions

- void **must_init** (bool test, const char *description)
Inicializa "cosas", y sale del programa si hay problemas, avisando dónde estuvo.
- void **allegro_inits** (void)
Inicializaciones de allegro. Si algo falla, lo notifica por consola y finaliza el programa.
- void **allegro_deinits** (void)
Desinicializaciones de allegro.
- void **allegro_reinit_display** (void)
Reinicializa el display de allegro.

- void **allegro_deinit_display** (void)
Desinicializa el display.
- unsigned char **allegro_get_last_key** (void)
Devuelve la ultima tecla presionada registrada.
- void **allegro_set_last_key** (unsigned char allegro_key_code)
Setea una ultima tecla presionada.
- ALLEGRO_EVENT_TYPE **allegro_wait_for_event** (void)
Espera a que ocurra un evento y lo devuelve.
- ALLEGRO_EVENT * **allegro_get_next_event** (void)
Devuelve el proximo evento de la cola, si es que existe. De no haber, devuelve false.
- ALLEGRO_EVENT **allegro_get_var_event** (void)
Devuelve el evento de allegro.
- bool **allegro_get_var_done** (void)
Devuelve flag de finalización del programa.
- bool **allegro_get_var_redraw** (void)
Devuelve flag de renderización.
- void **allegro_set_var_done** (bool state)
Setea flag de finalización del programa.
- void **allegro_set_var_redraw** (bool state)
Setea flag de renderizacion.
- ALLEGRO_FONT * **allegro_get_var_font** (void)
Devuelve la fuente de allegro.
- int **allegro_get_var_font_h** (void)
Devuelve el alto de un caracter de la fuente usada.
- int **allegro_get_var_font_w** (void)
Devuelve ancho de un caracter de la fuente usada.
- void **allegro_clear_display** (void)
Pone negro el display.
- void **allegro_draw_background** (void)
Dibuja la imagen de fondo.
- void **allegro_draw_menu_background** (int window)
Dibuja imagen de fondo de un menu dado (sin highlight en ninguna opcion)
- bool **allegro_is_event_queueVacia** (void)
Informa si al cola de eventos está vacía o no.
- ALLEGRO_EVENT_QUEUE * **allegro_get_event_queue** (void)
Devuelve puntero a la cola de eventos.
- void **allegro_set_var_event** (ALLEGRO_EVENT event)
Carga un evento de allegro.
- void **allegro_sound_set_stream_credits** (void)
Selecciona musica de credits. Comienza pausada.
- void **allegro_sound_set_stream_main_menu** (void)
Selecciona musica de menu. Comienza pausada.
- void **allegro_sound_set_stream_pause_menu** (void)
Selecciona musica de pausa. Comienza pausada.
- void **allegro_sound_set_stream_ranking** (void)
Selecciona musica de ranking. Comienza pausada.
- void **allegro_sound_set_stream_playing** (void)
Selecciona musica de jugando. Comienza pausada.
- void **allegro_sound_set_stream_rick** (void)
- void **allegro_sound_set_stream_game_over** (void)
Selecciona musica de game over. Comienza pausada.

- void **allegro_sound_toggle_stream** (void)
Cambia estado de reproduccion de la musica actual, si hay alguna seleccionada.
- void **allegro_sound_play_stream** (void)
Reproduce la musica actual, si hay alguna seleccionada.
- void **allegro_sound_pause_stream** (void)
Pausa la musica actual, si hay alguna seleccionada.
- void **allegro_sound_restart_stream** (void)
Reinicia la musica actual, si hay alguna seleccionada.
- void **allegro_sound_set_stream_gain_up** (void)
Aumenta en 0.1 la ganancia de stream.
- void **allegro_sound_set_stream_gain_down** (void)
Reduce en 0.1 la ganancia de stream.
- void **allegro_sound_play_effect_bonus** (void)
Reproduce efecto de bonus.
- void **allegro_sound_play_effect_click** (void)
Reproduce efecto de click (seleccion de menu//aceptar//etc.)
- void **allegro_sound_play_effect_crash** (void)
Reproduce efecto de choque.
- void **allegro_sound_play_effect_drowned** (void)
Reproduce efecto de ahogado (toco agua)
- void **allegro_sound_play_effect_goal** (void)
Reproduce efecto de 'llego a la meta'.
- void **allegro_sound_play_effect_jump** (void)
Reproduce efecto de salto.
- void **allegro_sound_play_effect_low_time** (void)
Reproduce efecto de 'queda poco tiempo'.
- void **allegro_sound_play_effect_run_completed** (void)
Reproduce efecto de 'run completada' (llego 5 veces a la meta)
- void **allegro_sound_play_effect_menu_enter** (void)
Reproduce efecto de 'menu enter'.
- void **allegro_sound_play_effect_new_max_score** (void)
Reproduce efecto de 'new_max_score'.
- void **allegro_sound_play_effect_exiting** (void)
Reproduce efecto de 'saliendo'.
- void **allegro_sound_play_effect_no_time** (void)
Reproduce efecto de 'sin tiempo'.
- void **allegro_sound_play_effect_coin_drop** (void)
Reproduce efecto de moneda tirada
- void **allegro_draw_hitbox** (int x, int y, int w, int h)
Dibuja un contorno rectangular.
- void **allegro_rick_on** (void)
- bool **allegro_get_rick_flag** (void)
- void **allegro_set_rick_flag** (bool state)
- void **allegro_rick_off** (void)
- void **allegro_rick_draw** (void)

Variables

- **sprites_t** sprites

4.17.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-10

Copyright

Copyright (c) 2022

4.17.2 Function Documentation

4.17.2.1 `allegro_draw_hitbox()`

```
void allegro_draw_hitbox (
    int x,
    int y,
    int w,
    int h )
```

Dibuja un contorno rectangular.

Parameters

<i>x</i>	Topleft x
<i>y</i>	Topleft y
<i>w</i>	Ancho
<i>h</i>	Largo

4.17.2.2 `allegro_draw_menu_background()`

```
void allegro_draw_menu_background (
    int window )
```

Dibuja imagen de fondo de un menu dado (sin highlight en ninguna opcion)

Parameters

<i>window</i>	enum MENU_WINDOWS
---------------	-------------------

4.17.2.3 `allegro_get_last_key()`

```
unsigned char allegro_get_last_key (  
    void )
```

Devuelve la ultima tecla presionada registrada.

Returns

unsigned char ALLEGRO_KEY_CODE

4.17.2.4 `allegro_get_next_event()`

```
ALLEGRO_EVENT * allegro_get_next_event (  
    void )
```

Devuelve el proximo evento de la cola, si es que existe. De no haber, devuele false.

Returns

ALLEGRO_EVENT*

4.17.2.5 `allegro_get_var_done()`

```
bool allegro_get_var_done (  
    void )
```

Devuelve flag de finalización del programa.

Returns

true Finaliza

false No finaliza

4.17.2.6 `allegro_get_var_event()`

```
ALLEGRO_EVENT allegro_get_var_event (  
    void )
```

Devuelve el evento de allegro.

Returns

ALLEGRO_EVENT

4.17.2.7 `allegro_get_var_font()`

```
ALLEGRO_FONT * allegro_get_var_font (  
    void )
```

Devuelve la fuente de allegro.

Returns

ALLEGRO_FONT

4.17.2.8 `allegro_get_var_font_h()`

```
int allegro_get_var_font_h (  
    void )
```

Devuelve el alto de un caracter de la funete usada.

Returns

int alto

4.17.2.9 `allegro_get_var_font_w()`

```
int allegro_get_var_font_w (  
    void )
```

Devuelve ancho de un caracter de la fuente usada.

Returns

int ancho

4.17.2.10 `allegro_get_var_redraw()`

```
bool allegro_get_var_redraw (
    void )
```

Devuelve flag de renderización.

Returns

true Renderiza
false No renderiza

4.17.2.11 `allegro_is_event_queueVacia()`

```
bool allegro_is_event_queueVacia (
    void )
```

Informa si al cola de eventos está vacía o no.

Returns

true Vacía
false No vacía

4.17.2.12 `allegro_set_last_key()`

```
void allegro_set_last_key (
    unsigned char allegro_key_code )
```

Setea una ultima tecla presionada.

Parameters

<i>allegro_key_code</i>	ALLEGRO_KEY_CODE
-------------------------	------------------

4.17.2.13 `allegro_set_rick_flag()`

```
void allegro_set_rick_flag (
    bool state )
```

Parameters

<i>state</i>	
--------------	--

4.17.2.14 allegro_set_var_done()

```
void allegro_set_var_done (
    bool state )
```

Setea flag de finalización del programa.

Parameters

<i>state</i>	true or false
--------------	---------------

4.17.2.15 allegro_set_var_event()

```
void allegro_set_var_event (
    ALLEGRO_EVENT event )
```

Carga un evento de allegro.

Parameters

<i>event</i>	
--------------	--

4.17.2.16 allegro_set_var_redraw()

```
void allegro_set_var_redraw (
    bool state )
```

Setea flag de renderizacion.

Parameters

<i>state</i>	true or false
--------------	---------------

4.17.2.17 `allegro_wait_for_event()`

```
ALLEGRO_EVENT_TYPE allegro_wait_for_event (
    void )
```

Espera a que ocurra un evento y lo devuelve.

Returns

ALLEGRO_EVENT_TYPE

4.17.2.18 `must_init()`

```
void must_init (
    bool test,
    const char * description )
```

Inicializa "cosas", y sale del programa si hay problemas, avisando dónde estuvo.

Parameters

<i>test</i>	Handler//booleano con status de la inicialización.
<i>description</i>	String con la descripción/nombre de la "cosa" a inicializar.

4.18 src/platform/pc/allegro_stuff.h File Reference

```
#include <allegro5/allegro5.h>
#include <allegro5/allegro_font.h>
#include <allegro5/allegro_ttf.h>
#include <allegro5/allegro_primitives.h>
#include <allegro5/allegro_image.h>
#include <allegro5/allegro_audio.h>
#include <allegro5/allegro_acodec.h>
#include <stdio.h>
#include "entities.h"
#include "geometry.h"
```

Data Structures

- struct [sprites_menu_t](#)
- struct [sprites_t](#)

Estructura principal de spritesheets.

Macros

- `#define FPS 60`

Enumerations

- enum **KEY_STATES** { **KEY_RELEASED** , **KEY_JUST_PRESSED** , **KEY_PRESSED** }

Functions

- void **must_init** (bool test, const char *description)
Inicializa "cosas", y sale del programa si hay problemas, avisando dónde estuvo.
- void **allegro_inits** (void)
Inicializaciones de allegro. Si algo falla, lo notifica por consola y finaliza el programa.
- void **allegro_deinits** (void)
Desinicializaciones de allegro.
- void **allegro_reinit_display** (void)
Reinicializa el display de allegro.
- void **allegro_deinit_display** (void)
Desinicializa el display.
- unsigned char **allegro_get_last_key** (void)
Devuelve la ultima tecla presionada registrada.
- void **allegro_set_last_key** (unsigned char allegro_key_code)
Setea una ultima tecla presionada.
- ALLEGRO_EVENT_TYPE **allegro_wait_for_event** (void)
Espera a que ocurra un evento y lo devuelve.
- ALLEGRO_EVENT * **allegro_get_next_event** (void)
Devuelve el proximo evento de la cola, si es que existe. De no haber, devuelve false.
- ALLEGRO_EVENT **allegro_get_var_event** (void)
Devuelve el evento de allegro.
- bool **allegro_get_var_done** (void)
Devuelve flag de finalización del programa.
- bool **allegro_get_var_redraw** (void)
Devuelve flag de renderización.
- void **allegro_set_var_done** (bool state)
Setea flag de finalización del programa.
- void **allegro_set_var_redraw** (bool state)
Setea flag de renderizacion.
- ALLEGRO_FONT * **allegro_get_var_font** (void)
Devuelve la fuente de allegro.
- int **allegro_get_var_font_h** (void)
Devuelve el alto de un caracter de la fuente usada.
- int **allegro_get_var_font_w** (void)
Devuelve ancho de un caracter de la fuente usada.
- void **allegro_clear_display** (void)
Pone negro el display.
- void **allegro_draw_background** (void)
Dibuja la imagen de fondo.
- void **allegro_draw_menu_background** (int window)
Dibuja imagen de fondo de un menu dado (sin highlight en ninguna opcion)
- bool **allegro_is_event_queueVacia** (void)
Informa si al cola de eventos está vacía o no.
- ALLEGRO_EVENT_QUEUE * **allegro_get_event_queue** (void)
Devuelve puntero a la cola de eventos.

- void **allegro_set_var_event** (ALLEGRO_EVENT event)
Carga un evento de allegro.
- void **allegro_sound_set_stream_credits** (void)
Selecciona musica de credits. Comienza pausada.
- void **allegro_sound_set_stream_main_menu** (void)
Selecciona musica de menu. Comienza pausada.
- void **allegro_sound_set_stream_pause_menu** (void)
Selecciona musica de pausa. Comienza pausada.
- void **allegro_sound_set_stream_ranking** (void)
Selecciona musica de ranking. Comienza pausada.
- void **allegro_sound_set_stream_playing** (void)
Selecciona musica de jugando. Comienza pausada.
- void **allegro_sound_set_stream_rick** (void)
- void **allegro_sound_set_stream_game_over** (void)
Selecciona musica de game over. Comienza pausada.
- void **allegro_sound_toggle_stream** (void)
Cambia estado de reproduccion de la musica actual, si hay alguna seleccionada.
- void **allegro_sound_play_stream** (void)
Reproduce la musica actual, si hay alguna seleccionada.
- void **allegro_sound_pause_stream** (void)
Pausa la musica actual, si hay alguna seleccionada.
- void **allegro_sound_restart_stream** (void)
Reinicia la musica actual, si hay alguna seleccionada.
- void **allegro_sound_set_stream_gain_up** (void)
Aumenta en 0.1 la ganancia de stream.
- void **allegro_sound_set_stream_gain_down** (void)
Reduce en 0.1 la ganancia de stream.
- void **allegro_sound_play_effect_bonus** (void)
Reproduce efecto de bonus.
- void **allegro_sound_play_effect_click** (void)
Reproduce efecto de click (seleccion de menu/aceptar/etc.)
- void **allegro_sound_play_effect_crash** (void)
Reproduce efecto de choque.
- void **allegro_sound_play_effect_drowned** (void)
Reproduce efecto de ahogado (toco agua)
- void **allegro_sound_play_effect_goal** (void)
Reproduce efecto de 'llego a la meta'.
- void **allegro_sound_play_effect_jump** (void)
Reproduce efecto de salto.
- void **allegro_sound_play_effect_low_time** (void)
Reproduce efecto de 'queda poco tiempo'.
- void **allegro_sound_play_effect_run_completed** (void)
Reproduce efecto de 'run completada' (llego 5 veces a la meta)
- void **allegro_sound_play_effect_menu_enter** (void)
Reproduce efecto de 'menu enter'.
- void **allegro_sound_play_effect_new_max_score** (void)
Reproduce efecto de 'new_max_score'.
- void **allegro_sound_play_effect_exiting** (void)
Reproduce efecto de 'saliendo'.
- void **allegro_sound_play_effect_no_time** (void)
Reproduce efecto de 'sin tiempo'.

- void **allegro_sound_play_effect_coin_drop** (void)
Reproduce efecto de moneda tirada
- void **allegro_draw_hitbox** (int x, int y, int w, int h)
Dibuja un contorno rectangular.
- void **allegro_rick_on** (void)
- bool **allegro_get_rick_flag** (void)
- void **allegro_set_rick_flag** (bool state)
- void **allegro_rick_off** (void)
- void **allegro_rick_draw** (void)

Variables

- **sprites_t** **sprites**

4.18.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-10

Copyright

Copyright (c) 2022

4.18.2 Function Documentation

4.18.2.1 **allegro_draw_hitbox()**

```
void allegro_draw_hitbox (  
    int x,  
    int y,  
    int w,  
    int h )
```

Dibuja un contorno rectangular.

Parameters

<i>x</i>	Topleft x
<i>y</i>	Topleft y
<i>w</i>	Ancho
<i>h</i>	Largo

4.18.2.2 allegro_draw_menu_background()

```
void allegro_draw_menu_background (
    int window )
```

Dibuja imagen de fondo de un menu dado (sin highlight en ninguna opcion)

Parameters

<i>window</i>	enum MENU_WINDOWS
---------------	-------------------

4.18.2.3 allegro_get_last_key()

```
unsigned char allegro_get_last_key (
    void )
```

Devuelve la ultima tecla presionada registrada.

Returns

unsigned char ALLEGRO_KEY_CODE

4.18.2.4 allegro_get_next_event()

```
ALLEGRO_EVENT * allegro_get_next_event (
    void )
```

Devuelve el proximo evento de la cola, si es que existe. De no haber, devuele false.

Returns

ALLEGRO_EVENT*

4.18.2.5 `allegro_get_var_done()`

```
bool allegro_get_var_done (
    void )
```

Devuelve flag de finalización del programa.

Returns

true Finaliza
false No finaliza

4.18.2.6 `allegro_get_var_event()`

```
ALLEGRO_EVENT allegro_get_var_event (
    void )
```

Devuelve el evento de allegro.

Returns

ALLEGRO_EVENT

4.18.2.7 `allegro_get_var_font()`

```
ALLEGRO_FONT * allegro_get_var_font (
    void )
```

Devuelve la fuente de allegro.

Returns

ALLEGRO_FONT

4.18.2.8 `allegro_get_var_font_h()`

```
int allegro_get_var_font_h (
    void )
```

Devuelve el alto de un caracter de la funete usada.

Returns

int alto

4.18.2.9 `allegro_get_var_font_w()`

```
int allegro_get_var_font_w (  
    void )
```

Devuelve ancho de un caracter de la fuente usada.

Returns

int ancho

4.18.2.10 `allegro_get_var_redraw()`

```
bool allegro_get_var_redraw (  
    void )
```

Devuelve flag de renderización.

Returns

true Renderiza

false No renderiza

4.18.2.11 `allegro_is_event_queueVacia()`

```
bool allegro_is_event_queueVacia (  
    void )
```

Informa si al cola de eventos está vacía o no.

Returns

true Vacía

false No vacía

4.18.2.12 `allegro_set_last_key()`

```
void allegro_set_last_key (  
    unsigned char allegro_key_code )
```

Setea una ultima tecla presionada.

Parameters

<i>allegro_key_code</i>	ALLEGRO_KEY_CODE
-------------------------	------------------

4.18.2.13 allegro_set_rick_flag()

```
void allegro_set_rick_flag (  
    bool state )
```

Parameters

<i>state</i>	
--------------	--

4.18.2.14 allegro_set_var_done()

```
void allegro_set_var_done (  
    bool state )
```

Setea flag de finalización del programa.

Parameters

<i>state</i>	true or false
--------------	---------------

4.18.2.15 allegro_set_var_event()

```
void allegro_set_var_event (  
    ALLEGRO_EVENT event )
```

Carga un evento de allegro.

Parameters

<i>event</i>	
--------------	--

4.18.2.16 allegro_set_var_redraw()

```
void allegro_set_var_redraw (  
    bool state )
```

Setea flag de renderizacion.

Parameters

<i>state</i>	true or false
--------------	---------------

4.18.2.17 allegro_wait_for_event()

```
ALLEGRO_EVENT_TYPE allegro_wait_for_event (
    void )
```

Espera a que ocurra un evento y lo devuelve.

Returns

ALLEGRO_EVENT_TYPE

4.18.2.18 must_init()

```
void must_init (
    bool test,
    const char * description )
```

Inicializa "cosas", y sale del programa si hay problemas, avisando dónde estuvo.

Parameters

<i>test</i>	Handler//booleano con status de la inicialización.
<i>description</i>	String con la descripción/nombre de la "cosa" a inicializar.

4.19 allegro_stuff.h

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

```
1
2 #ifndef _ALLEGRO_STUFF_H_
3 #define _ALLEGRO_STUFF_H_
4
5 /*****
6  * INCLUDE HEADER FILES
7  *****/
8
9 #include <allegro5/allegro5.h>
10 #include <allegro5/allegro_font.h>
11 #include <allegro5/allegro_ttf.h>
12 #include <allegro5/allegro_primitives.h>
13 #include <allegro5/allegro_image.h>
14
15 #include <allegro5/allegro_audio.h>
16 #include <allegro5/allegro_acodec.h>
```

```

27
28 #include <stdio.h>
29
30 #include "entities.h"
31 #include "geometry.h"
32
33 /*****
34  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
35  *****/
36
37 #define FPS 60
38
39 /*****
40  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
41  *****/
42
43 enum KEY_STATES
44 {
45     KEY_RELEASED,
46     KEY_JUST_PRESSED,
47     KEY_PRESSED
48 };
49
50 typedef struct
51 {
52
53 } sprites_menu_t;
54
55 typedef struct
56 {
57     ALLEGRO_BITMAP *frog_uncut;
58     ALLEGRO_BITMAP *frog[8];
59
60     ALLEGRO_BITMAP *background;
61
62     ALLEGRO_BITMAP *log;
63
64     ALLEGRO_BITMAP *cars_uncut;
65     ALLEGRO_BITMAP *car[CAR_TYPE_N];
66
67     ALLEGRO_BITMAP *turtle_uncut;
68     ALLEGRO_BITMAP *turtle[TURTLE_FRAMES];
69
70     ALLEGRO_BITMAP *heart;
71
72     struct
73     {
74         ALLEGRO_BITMAP *uncut;
75         ALLEGRO_BITMAP *option[MENU_STATE_MAX];
76         ALLEGRO_BITMAP *background;
77     } menu[MENU_WINDOW_MAX];
78
79     ALLEGRO_BITMAP *credits;
80
81     ALLEGRO_BITMAP *name;
82
83     ALLEGRO_BITMAP *icon;
84
85     ALLEGRO_BITMAP *dead;
86
87     struct
88     {
89         ALLEGRO_BITMAP *uncut;
90         ALLEGRO_BITMAP *frame[SPRITE_COIN_FRAMES];
91     } coin;
92
93     struct
94     {
95         ALLEGRO_BITMAP *uncut;
96         ALLEGRO_BITMAP *frame[SPRITE_SPLASH_FRAMES];
97     } splash;
98
99     ALLEGRO_BITMAP *border;
100 } sprites_t;
101
102 /*****
103  * VARIABLE PROTOTYPES WITH GLOBAL SCOPE
104  *****/
105
106 // estructura con punteros a sprites
107 extern sprites_t sprites;
108
109 /*****
110  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
111  *****/
112
113

```

```
124 void must_init(bool test, const char *description);
125
130 void allegro_inits(void);
131
136 void allegro_deinits(void);
137
142 void allegro_reinit_display(void);
143
148 void allegro_deinit_display(void);
149
155 unsigned char allegro_get_last_key(void);
156
162 void allegro_set_last_key(unsigned char allegro_key_code);
163
169 ALLEGRO_EVENT_TYPE allegro_wait_for_event(void);
170
176 ALLEGRO_EVENT *allegro_get_next_event(void);
177
183 ALLEGRO_EVENT allegro_get_var_event(void);
184
191 bool allegro_get_var_done(void);
192
199 bool allegro_get_var_redraw(void);
200
206 void allegro_set_var_done(bool state);
207
213 void allegro_set_var_redraw(bool state);
214
220 ALLEGRO_FONT *allegro_get_var_font(void);
221
227 int allegro_get_var_font_h(void);
228
234 int allegro_get_var_font_w(void);
235
240 void allegro_clear_display(void);
241
246 void allegro_draw_background(void);
247
253 void allegro_draw_menu_background(int window);
254
261 bool allegro_is_event_queueVacua(void);
262
267 ALLEGRO_EVENT_QUEUE *allegro_get_event_queue(void);
268
274 void allegro_set_var_event(ALLEGRO_EVENT event);
275
276 #pragma region allegro_sound
277
278 #pragma region allegro_sound_set_stream
283 void allegro_sound_set_stream_credits(void);
284
289 void allegro_sound_set_stream_main_menu(void);
290
295 void allegro_sound_set_stream_pause_menu(void);
296
301 void allegro_sound_set_stream_ranking(void);
302
307 void allegro_sound_set_stream_playing(void);
308
313 void allegro_sound_set_stream_rick(void);
314
319 void allegro_sound_set_stream_game_over(void);
320
321 #pragma endregion allegro_sound_set_stream
322
323 #pragma region allegro_sound_control
328 void allegro_sound_toggle_stream(void);
329
334 void allegro_sound_play_stream(void);
335
340 void allegro_sound_pause_stream(void);
341
346 void allegro_sound_restart_stream(void);
347
352 void allegro_sound_set_stream_gain_up(void);
353
358 void allegro_sound_set_stream_gain_down(void);
359
360 #pragma endregion allegro_sound_control
361
362 #pragma region allegro_sound_play_sample
367 void allegro_sound_play_effect_bonus(void);
368
373 void allegro_sound_play_effect_click(void);
374
379 void allegro_sound_play_effect_crash(void);
380
```

```

385 void allegro_sound_play_effect_drowned(void);
386
391 void allegro_sound_play_effect_goal(void);
392
397 void allegro_sound_play_effect_jump(void);
398
403 void allegro_sound_play_effect_low_time(void);
404
409 void allegro_sound_play_effect_run_completed(void);
410
415 void allegro_sound_play_effect_menu_enter(void);
416
421 void allegro_sound_play_effect_new_max_score(void);
422
427 void allegro_sound_play_effect_exiting(void);
428
433 void allegro_sound_play_effect_no_time(void);
434
439 void allegro_sound_play_effect_coin_drop(void);
440
441 #pragma endregion allegro_sound_play_sample
442
443 #pragma endregion allegro_sound
444
453 void allegro_draw_hitbox(int x, int y, int w, int h);
454
459 void allegro_rick_on(void);
460
465 bool allegro_get_rick_flag(void);
466
472 void allegro_set_rick_flag(bool state);
473
478 void allegro_rick_off(void);
479
484 void allegro_rick_draw(void);
485
486 /*****
487 *****/
488
489 #endif // _ALLEGRO_STUFF_H_

```

4.20 src/platform/pc/display.c File Reference

```

#include "../display.h"
#include "../ranking.h"
#include "allegro_stuff.h"
#include "game_data.h"
#include <pthread.h>

```

Macros

- #define **CREDITS_SCROLL_SPEED** 1
- #define **RANKING_PLAYER_X** 90
- #define **RANKING_SCORE_X** 500
- #define **RANKING_START_Y** 100

Functions

- bool **iniciarDisplay** ()
Inicializa el display de la plataforma.
- void **actualizarDisplay** ()
Actualiza el display de la plataforma.
- void **limpiarDisplay** ()
Limpia el display de la plataforma.

- void `mostrarTexto` (char *txt, int pos)
Muestra un texto dado en una posición dada (retiene el flujo)
- void `dejarTexto` (char *txt, int pos, bool repetir)
Deja el texto en la posición dada (no retiene)
- void `cargarRanking` (void)
Inicia muestreo de ranking en la plataforma.
- void `mostrarRanking` (void)
Bucle que muestra el ranking. Finaliza desde dentro, y hace que finalice el thread de ranking.
- void `cargarCreditos` (void)
Inicializa los creditos en la plataforma.
- void `mostrarCreditos` (void)
Bucle que muestra los creditos. Finaliza desde dentro, y hace que finalice el thread de ranking.
- void `reconfigurarDisplayON` (void)
Reconfigura el display de la plataforma y lo habilita.
- void `reconfigurarDisplayOFF` (void)
Reconfigura el display de la plataforma y lo deshabilita.

4.20.1 Detailed Description

Author

your name (`you@domain.com`)

Version

0.1

Date

2022-01-22

Copyright

Copyright (c) 2022

4.20.2 Function Documentation

4.20.2.1 `cargarRanking()`

```
void cargarRanking (  
    void )
```

Inicia muestreo de ranking en la plataforma.

Parameters

<i>txt</i>	
------------	--

4.20.2.2 dejarTexto()

```
void dejarTexto (
    char * txt,
    int pos,
    bool repetir )
```

Deja el texto en la posición data (no retiene)

Parameters

<i>txt</i>	
<i>pos</i>	
<i>repetir</i>	

4.20.2.3 iniciarDisplay()

```
bool iniciarDisplay ( )
```

Inicializa el display de la plataforma.

Returns

true Exit
false Error

4.20.2.4 mostrarCreditos()

```
void mostrarCreditos (
    void )
```

Bucle que muestra los creditos. Finaliza desde dentro, y hace que finalice el thread de ranking.

Returns

true No finaliz
false Finaliza

4.20.2.5 mostrarTexto()

```
void mostrarTexto (
    char * txt,
    int pos )
```

Muestra un texto dado en una posicion dada (retiene el flujo)

Parameters

<i>txt</i>	Texto
<i>pos</i>	Posicion

4.21 src/platform/pc/entities.h File Reference

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

Functions

- void **entities_init** (void)
Inicializa las entidades.
- void **entities_update** (void)
Actualiza las entidades.
- void **entities_draw** (void)
Dibuja las entidades.
- void **entities_move_frog** (unsigned char direction)
Indica que la rana debe dar un salto en la direccion dada.

4.21.1 Detailed Description

Author

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.21.2 Function Documentation

4.21.2.1 entities_move_frog()

```
void entities_move_frog (
    unsigned char direction )
```

Indica que la rana debe dar un salto en la direccion dada.

Parameters

<i>direction</i>	enum DIRECTIONS
------------------	-----------------

4.22 entities.h

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

```

1
10 #ifndef _ENTITIES_H_
11 #define _ENTITIES_H_
12
13 /*****
14  * INCLUDE HEADER FILES
15  *****/
16
17 #include <stdbool.h>
18
19 /*****
20  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
21  *****/
22
27 void entities_init(void);
28
33 void entities_update(void);
34
39 void entities_draw(void);
40
46 void entities_move_frog(unsigned char direction);
47
48 /*****
49  *****/
50
51 #endif // _ENTITIES_H_

```

4.23 src/platform/pc/game.c File Reference

```

#include "../..game.h"
#include "../..menu.h"
#include "../..queue.h"
#include "../..sound.h"
#include "game_data.h"
#include "entities.h"
#include "allegro_stuff.h"

```

Functions

- void [setNombre](#) (char *nombre)
Confirma el nombre del jugador.
- void [setMaxPuntos](#) (unsigned long long max)
Setea los puntos maximos del jugador.
- void [setDificultad](#) (int diff)
Setea la dificultad a usar.
- bool [tiempoRefrescoEntidades](#) (void)
Chequea si es tiempo de refrescar entidades según la plataforma.
- char * [getNombre](#) (void)
Devuelve el nombre del jugador.

- unsigned long long **getPuntos** (void)
Devuelve el puntaje del jugador.
- unsigned long long **getMaxPuntos** (void)
Devuelve el puntaje máximo del jugador.
- int **getNivel** (void)
Devuelve el nivel/run del jugador.
- void **inicializarJuego** (void)
Rutina que se ejecuta al comenzar a jugar, luego de ingresar el nombre.
- void **pausarJuego** (void)
Pausa el juego.
- void **reiniciarNivel** (void)
Configuraciones para reiniciar el nivel.
- void **refrescar** (void)
Actualizaciones relativas a actualizar las entidades.
- void **moverAdelante** (void)
Avanza el jugador.
- void **moverAtras** (void)
Retrocede el jugador.
- void **moverIzda** (void)
Mueve el jugador a la izquierda.
- void **moverDcha** (void)
Mueve el jugador a la derecha.
- void **respawn** (void)
Respawnea el jugador.
- void **actualizarInterfaz** (void)
Actualizaciones relativas a lo visual.
- void **reanudarJuego** (void)
Saca el juego de pausa.

4.23.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-22

Copyright

Copyright (c) 2022

4.23.2 Function Documentation

4.23.2.1 getMaxPuntos()

```
unsigned long long getMaxPuntos (
    void )
```

Devuelve el puntaje máximo del jugador.

Returns

unsigned long long

4.23.2.2 getNivel()

```
int getNivel (
    void )
```

Devuelve el nivel//run del jugador.

Returns

int

4.23.2.3 getNombre()

```
char * getNombre (
    void )
```

Devuelve el nombre del jugador.

Returns

char*

4.23.2.4 getPuntos()

```
unsigned long long getPuntos (
    void )
```

Devuelve el puntaje del jugador.

Returns

unsigned long long

4.23.2.5 setDificultad()

```
void setDificultad (
    int dif )
```

Setea la dificultad a usar.

Parameters

<i>dif</i>	
------------	--

4.23.2.6 setMaxPuntos()

```
void setMaxPuntos (
    unsigned long long max )
```

Setea los puntos maximos del jugador.

Parameters

<i>max</i>	
------------	--

4.23.2.7 setNombre()

```
void setNombre (
    char * nombre )
```

Confirma el nombre del jugador.

Parameters

<i>nombre</i>	
---------------	--

4.23.2.8 tiempoRefrescoEntidades()

```
bool tiempoRefrescoEntidades (
    void )
```

Chequea si es tiempo de refrescar entidades según la plataforma.

Returns

true

false

4.24 src/platform/rpi/game.c File Reference

Archivo para manejar la información del juego.

```
#include "../game.h"
#include "bitmap.h"
#include "../display.h"
#include "../sound.h"
#include "../queue.h"
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <time.h>
```

Macros

- #define **POS_AUTOS_INICIO** 4
- #define **POS_AUTOS_FIN** 13
- #define **CANT_CARRILES** 5
- #define **L_MAX** 64

Functions

- void **setNombre** (char *nombre)
Confirma el nombre del jugador.
- void **setMaxPuntos** (unsigned long long max)
Setea los puntos maximos del jugador.
- void **limpiarMapa** ()
- void **moverCarriles** (int x)
- void **spawnearAutos** ()
- void **actualizarMapa** ()
- void **refrescar** ()
Actualizaciones relativas a actualizar las entidades.
- bool **tiempoRefrescoEntidades** (void)
Chequea si es tiempo de refrescar entidades según la plataforma.
- void **setDificultad** (int dificultad)
Setea la dificultad a usar.
- char * **getNombre** ()
Devuelve el nombre del jugador.
- unsigned long long **getPuntos** ()
Devuelve el puntaje del jugador.
- unsigned long long **getMaxPuntos** ()
Devuelve el puntaje máximo del jugador.
- int **getNivel** ()
Devuelve el nivel/run del jugador.
- void **reiniciarNivel** ()
Configuraciones para reiniciar el nivel.
- void **respawn** ()
Respawnea el jugador.

- void **moverAdelante** ()
Avanza el jugador.
- void **moverAtras** ()
Retrocede el jugador.
- void **moverIzda** ()
Mueve el jugador a la izquierda.
- void **moverDcha** ()
Mueve el jugador a la derecha.
- void **perderVida** ()
Resta una vida.
- void **inicializarJuego** ()
Rutina que se ejecuta al comenzar a jugar, luego de ingresar el nombre.
- void **pausarJuego** ()
Pausa el juego.
- void **actualizarInterfaz** ()
Actualizaciones relativas a lo visual.
- void **reanudarJuego** (void)
Saca el juego de pausa.

Variables

- matriz_t **disp_matriz**

4.24.1 Detailed Description

Archivo para manejar la información del juego.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.24.2 Function Documentation

4.24.2.1 getMaxPuntos()

```
unsigned long long getMaxPuntos (  
    void )
```

Devuelve el puntaje máximo del jugador.

Returns

unsigned long long

4.24.2.2 getNivel()

```
int getNivel (
    void )
```

Devuelve el nivel/run del jugador.

Returns

int

4.24.2.3 getNombre()

```
char * getNombre (
    void )
```

Devuelve el nombre del jugador.

Returns

char*

4.24.2.4 getPuntos()

```
unsigned long long getPuntos (
    void )
```

Devuelve el puntaje del jugador.

Returns

unsigned long long

4.24.2.5 setDificultad()

```
void setDificultad (
    int dif )
```

Setea la dificultad a usar.

Parameters

<i>dif</i>	
------------	--

4.24.2.6 setMaxPuntos()

```
void setMaxPuntos (
    unsigned long long max )
```

Setea los puntos maximos del jugador.

Parameters

<i>max</i>	
------------	--

4.24.2.7 setNombre()

```
void setNombre (
    char * nombre )
```

Confirma el nombre del jugador.

Parameters

<i>nombre</i>	
---------------	--

4.24.2.8 tiempoRefrescoEntidades()

```
bool tiempoRefrescoEntidades (
    void )
```

Chequea si es tiempo de refrescar entidades según la plataforma.

Returns

true
false

4.25 src/platform/pc/game_data.c File Reference

```
#include "game_data.h"
#include <stdio.h>
#include <time.h>
#include "geometry.h"
#include "allegro_stuff.h"
```

Data Structures

- struct [data_t](#)

Macros

- `#define MAX_NAME_CHAR 20`
- `#define MAX_LIVES 3`
- `#define SCORE_PER_GOAL 500`
- `#define SCORE_PER_GOAL_COIN 750`
- `#define SCORE_PER_RUN 1000`
- `#define INITIAL_RUN_TIME_LEFT 30`
- `#define EXTRA_TIME_PER_GOAL 10`
- `#define EXTRA_TIME_PER_BONUS_GOAL 15`
- `#define TIME_LEFT_WARNING 10`

Enumerations

- enum `DATA_FLAGS` { `DATA_FLAG_STARTING` , `DATA_FLAG_NEXT_RUN` , `DATA_FLAG_TIME_EXCEEDED` , `DATA_FLAG_GAME_OVER` }
- enum `HUD_EXTRAS` { `HUD_EXTRA_TIME` , `HUD_EXTRA_SCORE` , `HUD_EXTRA_LIFE` , `HUD_EXTRAS_MAX` }

Functions

- void **game_data_init** (void)
Inicializa datos internos del juego.
- void **game_data_update** (void)
Actualiza datos internos del juego.
- void **game_data_draw** (void)
Grafica datos del juego (HUD principalmente)
- int [game_data_get_lives](#) (void)
Devuelve vidas.
- void **game_data_subtract_live** (void)
Resta una vida.
- unsigned long long [game_data_get_score](#) (void)
Devuelve score.
- void **game_data_add_score** (void)
Agrega score por llegar a la meta.
- void **game_data_add_score_bonus** (void)
Agrega score por llegar a la meta con bonus (coins)
- void [game_data_set_score_max](#) (unsigned long long score)
Carga el score maximo del jugador actual.
- unsigned long long [game_data_get_score_max](#) (void)
Devuelve el score maximo del jugador actual.
- int [game_data_get_run_number](#) (void)
Devuelve el numero de run.
- void **game_data_next_run** (void)
Indica que se pase a la siguiente run.
- int [game_data_get_run_time_left](#) (void)

- Devuelve el tiempo restante de la run en segundos.*
- void **game_data_add_run_time_goal** (void)
Agrega tiempo a la run por llegar a una meta.
- void **game_data_add_run_time_goal_bonus** (void)
Agrega tiempo (más) a la run por llegar a una meta con coin.
- unsigned long **game_data_get_frames** (void)
Devuelve los frames transcurridos del juego.
- int **game_data_get_timer_in_sec** (void)
Devuelve el tiempo transcurrido en segundos.
- void **game_data_set_diff** (int diff)
Setea dificultad.
- int **game_data_get_diff** (void)
Devuelve dificultad.
- void **game_data_clear_name** (void)
Limpia el nombre del jugador.
- void **game_data_overwrite_name** (char *name)
Sobreescribe el nombre del jugador.
- void **game_data_add_name_letter** (char letter)
Agrega una letra la nombre del jugador.
- char * **game_data_get_name** (void)
Devuelve puntero al nombre del jugador.
- bool **game_data_get_goal_state** (unsigned int goal)
Revisa si un punto de llegada es valido o no (vacío o lleno)
- void **game_data_set_goal** (unsigned int goal)
Setea un goal como completado.
- void **game_data_reset_goals** (void)
Habilita todos los goals.
- bool **game_data_get_time_left_flag** (void)
Avisa si se excedió el tiempo de juego.
- bool **game_data_get_game_over_flag** (void)
Devuelve flag de game over.
- bool **game_data_are_goals_full** (void)
Avisa si estan todas las metas completas.
- unsigned long long **game_data_get_old_max_score** (void)
Devuelve el score maximo sin actualizar al terminar el juego.

4.25.1 Detailed Description

Author

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.25.2 Function Documentation

4.25.2.1 `game_data_add_name_letter()`

```
void game_data_add_name_letter (
    char letter )
```

Agrega una letra la nombre del jugador.

Parameters

<i>letter</i>	Letra
---------------	-------

4.25.2.2 `game_data_are_goals_full()`

```
bool game_data_are_goals_full (
    void )
```

Avisa si estan todas las metas completas.

Returns

true Si
false No

4.25.2.3 `game_data_get_diff()`

```
int game_data_get_diff (
    void )
```

Devuelve dificultad.

Returns

int

4.25.2.4 `game_data_get_frames()`

```
unsigned long game_data_get_frames (
    void )
```

Devuelve los frames transcurridos del juego.

Returns

unsigned long Frames transcurridos

4.25.2.5 game_data_get_game_over_flag()

```
bool game_data_get_game_over_flag (
    void )
```

Devuelve flag de game over.

Returns

true Game over
false No game over

4.25.2.6 game_data_get_goal_state()

```
bool game_data_get_goal_state (
    unsigned int goal )
```

Revisa si un punto de llegada es valido o no (vacio o lleno)

Parameters

<i>goal</i>	0 a MAX_GOALS-1
-------------	-----------------

Returns

true Invalido
false Valido

4.25.2.7 game_data_get_lives()

```
int game_data_get_lives (
    void )
```

Devuelve vidas.

Returns

int vidas

4.25.2.8 game_data_get_name()

```
char * game_data_get_name (  
    void )
```

Devuelve puntero al nombre del jugador.

Returns

char*

4.25.2.9 game_data_get_old_max_score()

```
unsigned long long game_data_get_old_max_score (  
    void )
```

Devuelve el score maximo sin actualizar al terminar el juego.

Returns

unsigned long long

4.25.2.10 game_data_get_run_number()

```
int game_data_get_run_number (  
    void )
```

Devuelve el numero de run.

Returns

int Numero de run

4.25.2.11 game_data_get_run_time_left()

```
int game_data_get_run_time_left (  
    void )
```

Devuelve el tiempo restante de la run en segundos.

Returns

int Tiempo restante

4.25.2.12 game_data_get_score()

```
unsigned long long game_data_get_score (
    void )
```

Devuelve score.

Returns

int

4.25.2.13 game_data_get_score_max()

```
unsigned long long game_data_get_score_max (
    void )
```

Devuelve el score maximo del jugador actual.

Returns

unsigned long long Score maximo

4.25.2.14 game_data_get_time_left_flag()

```
bool game_data_get_time_left_flag (
    void )
```

Avisa si se excedio el tiempo de juego.

Returns

true Excedido

false No excedido

4.25.2.15 game_data_get_timer_in_sec()

```
int game_data_get_timer_in_sec (
    void )
```

Devuelve el tiempo transcurrido en segundos.

Returns

int Segundos transcurridos

4.25.2.16 game_data_overwrite_name()

```
void game_data_overwrite_name (
    char * name )
```

Sobreescribe el nombre del jugador.

Parameters

<i>name</i>	
-------------	--

4.25.2.17 game_data_set_diff()

```
void game_data_set_diff (
    int diff )
```

Setea dificultad.

Parameters

<i>diff</i>	enum DIFFICULTIES
-------------	-------------------

4.25.2.18 game_data_set_goal()

```
void game_data_set_goal (
    unsigned int goal )
```

Setea un goal como completado.

Parameters

<i>goal</i>	0 a MAX_GOALS-1
-------------	-----------------

4.25.2.19 game_data_set_score_max()

```
void game_data_set_score_max (
    unsigned long long score )
```

Carga el score maximo del jugador actual.

Parameters

<i>score</i>	
--------------	--

4.26 src/platform/pc/game_data.h File Reference

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


Enumerations

- enum **DIFFICULTIES** { **DIFFICULTIES_EASY** = 1 , **DIFFICULTIES_NORMAL** , **DIFFICULTIES_HARD** }

Functions

- void **game_data_init** (void)
Inicializa datos internos del juego.
- void **game_data_update** (void)
Actualiza datos internos del juego.
- void **game_data_draw** (void)
Grafica datos del juego (HUD pricipalmente)
- int **game_data_get_lives** (void)
Devuelve vidas.
- void **game_data_subtract_live** (void)
Resta una vida.
- unsigned long long **game_data_get_score** (void)
Devuelve score.
- void **game_data_add_score** (void)
Agrega score por llegar a la meta.
- void **game_data_add_score_bonus** (void)
Agrega score por llegar a la meta con bonus (coins)
- void **game_data_set_score_max** (unsigned long long score)
Carga el score maximo del jugador actual.
- unsigned long long **game_data_get_score_max** (void)
Devuelve el score maximo del jugador actual.
- int **game_data_get_run_number** (void)
Devuelve el numero de run.
- void **game_data_next_run** (void)
Indica que se pase a la siguiente run.
- int **game_data_get_run_time_left** (void)
Devuelve el tiempo restante de la run en segundos.
- void **game_data_add_run_time_goal** (void)
Agrega tiempo a la run por llegar a una meta.
- void **game_data_add_run_time_goal_bonus** (void)
Agrega tiempo (más) a la run por llegar a una meta con coin.
- unsigned long **game_data_get_frames** (void)
Devuelve los frames transcurridos del juego.
- int **game_data_get_timer_in_sec** (void)
Devuelve el tiempo transcurrido en segundos.
- void **game_data_set_diff** (int diff)
Setea dificultad.
- int **game_data_get_diff** (void)
Devuelve dificultad.
- void **game_data_clear_name** (void)
Limpia el nombre del jugador.
- void **game_data_overwrite_name** (char *name)

- Sobreescribe el nombre del jugador.*
- void `game_data_add_name_letter` (char letter)
Agrega una letra la nombre del jugador.
- char * `game_data_get_name` (void)
Devuelve puntero al nombre del jugador.
- bool `game_data_get_goal_state` (unsigned int goal)
Revisa si un punto de llegada es valido o no (vacio o lleno)
- void `game_data_set_goal` (unsigned int goal)
Setea un goal como completado.
- void `game_data_reset_goals` (void)
Habilita todos los goals.
- bool `game_data_get_time_left_flag` (void)
Avisa si se excedio el tiempo de juego.
- bool `game_data_get_game_over_flag` (void)
Devuelve flag de game over.
- bool `game_data_are_goals_full` (void)
Avisa si estan todas las metas completas.
- unsigned long long `game_data_get_old_max_score` (void)
Devuelve el score maximo sin actualizar al terminar el juego.

4.26.1 Detailed Description

Author

your name (`you@domain.com`)

Version

0.1

Date

2022-01-17

Copyright

Copyright (c) 2022

4.26.2 Function Documentation

4.26.2.1 `game_data_add_name_letter()`

```
void game_data_add_name_letter (
    char letter )
```

Agrega una letra la nombre del jugador.

Parameters

<i>letter</i>	Letra
---------------	-------

4.26.2.2 game_data_are_goals_full()

```
bool game_data_are_goals_full (
    void )
```

Avisa si estan todas las metas completas.

Returns

true Si
false No

4.26.2.3 game_data_get_diff()

```
int game_data_get_diff (
    void )
```

Devuelve dificultad.

Returns

int

4.26.2.4 game_data_get_frames()

```
unsigned long game_data_get_frames (
    void )
```

Devuelve los frames transcurridos del juego.

Returns

unsigned long Frames transcurridos

4.26.2.5 game_data_get_game_over_flag()

```
bool game_data_get_game_over_flag (
    void )
```

Devuelve flag de game over.

Returns

true Game over
false No game over

4.26.2.6 game_data_get_goal_state()

```
bool game_data_get_goal_state (
    unsigned int goal )
```

Revisa si un punto de llegada es valido o no (vacío o lleno)

Parameters

<i>goal</i>	0 a MAX_GOALS-1
-------------	-----------------

Returns

true Invalido
false Valido

4.26.2.7 game_data_get_lives()

```
int game_data_get_lives (
    void )
```

Devuelve vidas.

Returns

int vidas

4.26.2.8 game_data_get_name()

```
char * game_data_get_name (  
    void )
```

Devuelve puntero al nombre del jugador.

Returns

char*

4.26.2.9 game_data_get_old_max_score()

```
unsigned long long game_data_get_old_max_score (  
    void )
```

Devuelve el score maximo sin actualizar al terminar el juego.

Returns

unsigned long long

4.26.2.10 game_data_get_run_number()

```
int game_data_get_run_number (  
    void )
```

Devuelve el numero de run.

Returns

int Numero de run

4.26.2.11 game_data_get_run_time_left()

```
int game_data_get_run_time_left (  
    void )
```

Devuelve el tiempo restante de la run en segundos.

Returns

int Tiempo restante

4.26.2.12 game_data_get_score()

```
unsigned long long game_data_get_score (
    void )
```

Devuelve score.

Returns

int

4.26.2.13 game_data_get_score_max()

```
unsigned long long game_data_get_score_max (
    void )
```

Devuelve el score maximo del jugador actual.

Returns

unsigned long long Score maximo

4.26.2.14 game_data_get_time_left_flag()

```
bool game_data_get_time_left_flag (
    void )
```

Avisa si se excedio el tiempo de juego.

Returns

true Excedido

false No excedido

4.26.2.15 game_data_get_timer_in_sec()

```
int game_data_get_timer_in_sec (
    void )
```

Devuelve el tiempo transcurrido en segundos.

Returns

int Segundos transcurridos

4.26.2.16 game_data_overwrite_name()

```
void game_data_overwrite_name (
    char * name )
```

Sobreescribe el nombre del jugador.

Parameters

<i>name</i>	
-------------	--

4.26.2.17 game_data_set_diff()

```
void game_data_set_diff (
    int diff )
```

Setea dificultad.

Parameters

<i>diff</i>	enum DIFFICULTIES
-------------	-------------------

4.26.2.18 game_data_set_goal()

```
void game_data_set_goal (
    unsigned int goal )
```

Setea un goal como completado.

Parameters

<i>goal</i>	0 a MAX_GOALS-1
-------------	-----------------

4.26.2.19 game_data_set_score_max()

```
void game_data_set_score_max (
    unsigned long long score )
```

Carga el score maximo del jugador actual.

Parameters

<i>score</i>	
--------------	--

4.27 game_data.h

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

```

1
2 #ifndef _GAME_DATA_H_
3 #define _GAME_DATA_H_
4
5 /*****
6  * INCLUDE HEADER FILES
7  *****/
8
9 #include <stdbool.h>
10
11 /*****
12  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
13  *****/
14
15 enum DIFFICULTIES
16 {
17     DIFFICULTIES_EASY = 1,
18     DIFFICULTIES_NORMAL,
19     DIFFICULTIES_HARD
20 };
21
22 /*****
23  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
24  *****/
25
26 void game_data_init(void);
27
28 void game_data_update(void);
29
30 void game_data_draw(void);
31
32 int game_data_get_lives(void);
33
34 void game_data_subtract_live(void);
35
36 unsigned long long game_data_get_score(void);
37
38 void game_data_add_score(void);
39
40 void game_data_add_score_bonus(void);
41
42 void game_data_set_score_max(unsigned long long score);
43
44 unsigned long long game_data_get_score_max(void);
45
46 int game_data_get_run_number(void);
47
48 void game_data_next_run(void);
49
50 int game_data_get_run_time_left(void);
51
52 void game_data_add_run_time_goal(void);
53
54 void game_data_add_run_time_goal_bonus(void);
55
56 unsigned long game_data_get_frames(void);
57
58 int game_data_get_timer_in_sec(void);
59
60 void game_data_set_diff(int diff);
61
62 int game_data_get_diff(void);
63 void game_data_clear_name(void);
64
65 void game_data_overwrite_name(char *name);
66
67 void game_data_add_name_letter(char letter);
68
69 char *game_data_get_name(void);
70
71 bool game_data_get_goal_state(unsigned int goal);
72
73 void game_data_set_goal(unsigned int goal);
74
75 void game_data_reset_goals(void);
76
77 bool game_data_get_time_left_flag(void);
78
79 bool game_data_get_game_over_flag(void);
80
81 bool game_data_are_goals_full(void);
82
83 unsigned long long game_data_get_old_max_score(void);
84
85 /*****
86  *****/
87

```



```
242 #endif // _GAME_DATA_H_
```

4.28 src/platform/pc/geometry.c File Reference

```
#include "geometry.h"
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
```

Functions

- int [get_rand_between](#) (int low, int high)
Devuelve un random entre dos numeros dados.
- bool [collide](#) (int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2)
Comprueba colición de hitboxes rectangulares.
- bool [collideShort](#) (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh)
Similar a 'collide', pero pide ancho y alto en vez de segundas coordenadas.
- bool [inside](#) (int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2)
Detecta si un rectángulo está dentro de otro.
- bool [insideShort](#) (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh)
Similar a 'inside', pero pide ancho y alto en vez de segundas coordenadas.
- bool [insideShortScaled](#) (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh, float scale)
Similar a 'insideShort', pero permite setear cuan dentro debe estar una box dentro de la otra para tomarla como tal.
- int [mapInt](#) (int source, int min_in, int max_in, int min_out, int max_out)
Toma un valor comprendido dentro de un rango (in) y lo devuelve (mapea) a otro rango (out)
- [pair_xy_t](#) [getXYFromFrogFrame](#) (int frame)
Devuelve par de coordenadas xy topleft de un frame dado del sprite de la rana.
- [pair_xy_t](#) [getXYFromTurtleFrame](#) (int frame)
Devuelve par de coordenadas xy topleft de un frame dado del sprite de la tortuga.
- [pair_xy_t](#) [getXYFromCarFrame](#) (int frame)
Devuelve par de coordenadas xy topleft de un frame dado del sprite del auto.
- [pair_xy_t](#) [getXYFromCoinFrame](#) (int frame)
Devuelve par de coordenadas xy topleft de un frame dado del sprite de coin.
- [pair_xy_t](#) [getXYFromSplashFrame](#) (int frame)
Devuelve par de coordenadas xy topleft de un frame dado del sprite de splash.
- bool [matchUInt](#) (unsigned int val, const unsigned int *array)
Verifica si un numero coincide con alguno de un array dado (uints)

Variables

- const unsigned int [lanes_logs](#) [LANES_LOG_TOTAL] = {2, 4, 5}
Filas para troncos.
- const unsigned int [lanes_turtles](#) [LANES_TURTLE_TOTAL] = {3, 6}
Filas para tortugas.
- const unsigned int [lanes_cars](#) [LANES_CAR_TOTAL] = {8, 9, 10, 11, 12}
Filas para autos.
- const unsigned int [goal_cols](#) [MAX_GOALS] = {1, 4, 7, 10, 13}
Columnas para puntos de llegada.

4.28.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-10

Copyright

Copyright (c) 2022

4.28.2 Function Documentation

4.28.2.1 collide()

```
bool collide (
    int ax1,
    int ay1,
    int ax2,
    int ay2,
    int bx1,
    int by1,
    int bx2,
    int by2 )
```

Comprueba colición de hitboxes rectangulares.

Parameters

<i>ax1</i>	opleft corner de a (x)
<i>ay1</i>	opleft corner de a (y)
<i>ax2</i>	ottomright corner de a (x)
<i>ay2</i>	ottomright corner de a (y)
<i>bx1</i>	opleft corner de b (x)
<i>by1</i>	opleft corner de b (y)
<i>bx2</i>	ottomright corner de b (x)
<i>by2</i>	ottomright corner de b (y)

Returns

true Colisión
false No colisión

4.28.2.2 collideShort()

```
bool collideShort (
    int ax,
    int ay,
    int aw,
    int ah,
    int bx,
    int by,
    int bw,
    int bh )
```

Similar a 'collide', pero pide ancho y alto en vez de segundas coordenadas.

Parameters

<i>ax</i>	opleft corner x de a
<i>ay</i>	opleft corner y de a
<i>aw</i>	ancho de a
<i>ah</i>	alto de a
<i>bx</i>	opleft corner x de b
<i>by</i>	opleft corner y de b
<i>bw</i>	ancho de b
<i>bh</i>	alto de b

Returns

true Colision
false No colision

4.28.2.3 get_rand_between()

```
int get_rand_between (
    int low,
    int high )
```

Devuelve un randon entre dos numeros dados.

Parameters

<i>low</i>	Valor inferior
<i>high</i>	Valor superior

Returns

int Valor random

4.28.2.4 getXYFromCarFrame()

```
pair_xy_t getXYFromCarFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite del auto.

Parameters

<i>frame</i>	Numero de frame (0 a CAR_TYPE_N - 1)
--------------	--------------------------------------

Returns

pair_xy_t Par de coordenandas

4.28.2.5 getXYFromCoinFrame()

```
pair_xy_t getXYFromCoinFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de coin.

Parameters

<i>frame</i>	Numero de frame (0 a SPRITE_COIN_FRAMES - 1)
--------------	--

Returns

pair_xy_t Par de coordenandas

4.28.2.6 getXYFromFrogFrame()

```
pair_xy_t getXYFromFrogFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la rana.

Parameters

<i>frame</i>	Numero de frame (0 a FROG_FRAMES - 1)
--------------	---------------------------------------

Returns

`pair_xy_t` Par de coordenadas

4.28.2.7 getXYFromSplashFrame()

```
pair_xy_t getXYFromSplashFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de splash.

Parameters

<i>frame</i>	Numero de frame (0 a SPRITE_SPLASH_FRAMES - 1)
--------------	--

Returns

`pair_xy_t` Par de coordenandas

4.28.2.8 getXYFromTurtleFrame()

```
pair_xy_t getXYFromTurtleFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la tortuga.

Parameters

<i>frame</i>	Numero de frame (0 a TURTLE_FRAMES - 1)
--------------	---

Returns

`pair_xy_t` Par de coordenadas

4.28.2.9 inside()

```
bool inside (  
    int ax1,
```

```

    int ay1,
    int ax2,
    int ay2,
    int bx1,
    int by1,
    int bx2,
    int by2 )

```

Detecta si un rectángulo está dentro de otro.

Parameters

<i>ax1</i>	opleft corner de big (x)
<i>ay1</i>	opleft corner de big (y)
<i>ax2</i>	bottomright corner de big (x)
<i>ay2</i>	bottomright corner de big (y)
<i>bx1</i>	opleft corner de small (x)
<i>by1</i>	opleft corner de small (y)
<i>bx2</i>	bottomright corner de small (x)
<i>by2</i>	bottomright corner de small (y)

Returns

true Está dentro
 false Está fuera

4.28.2.10 insideShort()

```

bool insideShort (
    int ax,
    int ay,
    int aw,
    int ah,
    int bx,
    int by,
    int bw,
    int bh )

```

Similar a 'inside', pero pide ancho y alto en vez de segundas coordenadas.

Parameters

<i>ax</i>	opleft corner x de big
<i>ay</i>	opleft corner y de big
<i>aw</i>	ancho de big
<i>ah</i>	alto de big
<i>bx</i>	opleft corner x de big
<i>by</i>	opleft corner y de big
<i>bw</i>	ancho de big
<i>bh</i>	alto de big

Returns

true Esta dentro
false Esta fuera

4.28.2.11 insideShortScaled()

```
bool insideShortScaled (
    int ax,
    int ay,
    int aw,
    int ah,
    int bx,
    int by,
    int bw,
    int bh,
    float scale )
```

Similar a 'insideShort', pero permite setear cuan dentro debe estar una box dentro de la otra para tomarla como tal.

Parameters

<i>ax</i>	topleft corner x de big
<i>ay</i>	topleft corner y de big
<i>aw</i>	ancho de big
<i>ah</i>	alto de big
<i>bx</i>	topleft corner x de big
<i>by</i>	topleft corner y de big
<i>bw</i>	ancho de big
<i>bh</i>	alto de big
<i>scale</i>	Factor de insercion. Entre 0.0 (nada metido) y 1.0 (completamente metido). Otro valor devuelve false

Returns

true
false

4.28.2.12 mapInt()

```
int mapInt (
    int source,
    int min_in,
    int max_in,
    int min_out,
    int max_out )
```

Toma un valor comprendido dentro de un rango (in) y lo devuelve (mapea) a otro rango (out)

@source <https://stackoverflow.com/questions/5731863/mapping-a-numeric-range-onto-another>

Parameters

<i>source</i>	Valor a mapear
<i>min_in</i>	Límite inferior del rango de entrada
<i>max_in</i>	Límite superior del rango de entrada
<i>min_out</i>	Límite inferior del rango de salida
<i>max_out</i>	Límite superior del rango de salida

Returns

int Valor mapeado

4.28.2.13 matchUInt()

```
bool matchUInt (
    unsigned int val,
    const unsigned int * array )
```

Verifica si un numero coincide con alguno de un array dado (uints)

Parameters

<i>val</i>	Numero a verificar
<i>array</i>	Array de numeros

Returns

true Existe
false No existe

4.28.3 Variable Documentation**4.28.3.1 goal_cols**

```
const unsigned int goal_cols[MAX_GOALS] = {1, 4, 7, 10, 13}
```

Columnas para puntos de llegada.

Columnas para puntos de llegada, referenciadas a 0.

4.28.3.2 lanes_cars

```
const unsigned int lanes_cars[LANES_CAR_TOTAL] = {8, 9, 10, 11, 12}
```

Filas para autos.

Filas para autos, referenciadas a 0.

4.28.3.3 lanes_logs

```
const unsigned int lanes_logs[LANES_LOG_TOTAL] = {2, 4, 5}
```

Filas para troncos.

Filas para troncos, referenciadas a 0.

4.28.3.4 lanes_turtles

```
const unsigned int lanes_turtles[LANES_TURTLE_TOTAL] = {3, 6}
```

Filas para tortugas.

Filas para tortugas, referenciadas a 0.

4.29 src/platform/pc/geometry.h File Reference

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

Data Structures

- struct [pair_xy_t](#)

Macros

- `#define DISPLAY_W 690`
- `#define DISPLAY_H 644`
- `#define ROWS 14`
- `#define COLS 15`
- `#define CELL_H 46`
- `#define CELL_W 46`
- `#define MAX_LANES (ROWS - 1)`
- `#define LANES_CAR_TOTAL 5`
- `#define LANES_LOG_TOTAL 3`
- `#define LANES_TURTLE_TOTAL 2`
- `#define CELL_TOPLEFT_X 0`
- `#define CELL_TOPLEFT_Y 0`
- `#define CELL_START_X (CELL_TOPLEFT_X + CELL_W * ((COLS - 1) / 2))`
- `#define CELL_START_Y (CELL_TOPLEFT_Y + CELL_H * (ROWS - 1))`
- `#define FROG_W 30`
- `#define FROG_H 30`
- `#define FROG_OFFSET_X (CELL_W / 2 - FROG_W / 2)`
- `#define FROG_OFFSET_Y (CELL_H / 2 - FROG_H / 2)`
- `#define CELL_START_FROG_X (CELL_START_X + FROG_OFFSET_X)`
- `#define CELL_START_FROG_Y (CELL_START_Y + FROG_OFFSET_Y)`
- `#define FROG_FRAMES 8`
- `#define SPRITE_SIZE_FROG_STATIC_H FROG_H`
- `#define SPRITE_SIZE_FROG_STATIC_W FROG_W`
- `#define SPRITE_SIZE_FROG_DYNAMIC_LONG 46`
- `#define SPRITE_SIZE_FROG_DYNAMIC_SHORT FROG_W`
- `#define STEP_FULL_SIZE CELL_H`
- `#define STEP_RATIO (CELL_H / 3)`
- `#define STEP_FRACTION_SIZE (STEP_FULL_SIZE / STEP_RATIO)`
- `#define FROG_MAX_X (DISPLAY_W - (CELL_W - FROG_OFFSET_X))`
- `#define FROG_MAX_Y (DISPLAY_H - (CELL_H - FROG_OFFSET_Y))`
- `#define FROG_MIN_X (CELL_TOPLEFT_X + FROG_OFFSET_X)`
- `#define FROG_MIN_Y (CELL_TOPLEFT_Y + FROG_OFFSET_Y + CELL_H)`
- `#define LOG_W (4 * CELL_W)`
- `#define LOG_H 40`
- `#define LOG_OFFSET_X 0`
- `#define LOG_OFFSET_Y (CELL_H / 2 - LOG_H / 2)`
- `#define CAR_W CELL_W + 26`
- `#define CAR_TRUCK_FIRE_W (3 * CELL_W)`
- `#define CAR_TRUCK_W (4 * CELL_W)`
- `#define CAR_H 40`
- `#define CAR_OFFSET_X 0`
- `#define CAR_OFFSET_Y (CELL_H / 2 - CAR_H / 2)`
- `#define TURTLE_FRAMES 11`
- `#define TURTLE_SIDE CELL_W`
- `#define TURTLE_FRAME_OFFSET_XY (CELL_W / 2 - TURTLE_SIDE / 2)`
- `#define SPRITE_SIZE_HEART 20`
- `#define SPRITE_DEAD_SIZE 35`
- `#define SPRITE_DEAD_OFFSET (CELL_W / 2 - SPRITE_DEAD_SIZE / 2)`
- `#define SPRITE_COIN_FRAMES 6`
- `#define SPRITE_COIN_SIDE 24`
- `#define SPRITE_COIN_OFFSET_XY (CELL_W / 2 - SPRITE_COIN_SIDE / 2)`
- `#define SPRITE_SPLASH_FRAMES 6`
- `#define SPRITE_SPLASH_W 98`

- `#define SPRITE_SPLASH_H 68`
- `#define SPRITE_SPLASH_OFFSET_X (CELL_W / 2 - SPRITE_SPLASH_W / 2)`
- `#define SPRITE_SPLASH_OFFSET_Y (CELL_W / 2 - SPRITE_SPLASH_H / 2)`
- `#define SPRITE_BORDER_START_X 0`
- `#define SPRITE_BORDER_START_Y CELL_H`
- `#define MENU_OPTION_TOPLEFT_X 45`
- `#define MENU_OPTION_TOPLEFT_Y 72`
- `#define MENU_OPTION_DELTA_Y 100`
- `#define MENU_OPTION_W 600`
- `#define MENU_OPTION_H 75`
- `#define CREDITS_SCREEN_LENGTH 2576`
- `#define CREDITS_SCREEN_START 0`
- `#define CREDITS_SCREEN_FINAL (CREDITS_SCREEN_LENGTH - DISPLAY_H)`
- `#define INSERTION_FACTOR (double)0.5`
- `#define GOAL_ROW_OFFSET_Y_FIX 5`
- `#define GOAL_ROW_MARGIN_TO_REACH 5`

Typedefs

- `typedef enum CAR_TYPE CAR_TYPE`

Enumerations

- `enum GOALS {`
`GOAL_LEFT, GOAL_LEFT_MID, GOAL_MID, GOAL_RIGHT_MID,`
`GOAL_RIGHT, MAX_GOALS }`
- `enum DIRECTIONS {`
`DIRECTION_NONE, DIRECTION_UP, DIRECTION_RIGHT, DIRECTION_LEFT,`
`DIRECTION_DOWN }`
- `enum MENU_STATES {`
`MENU_STATE_OPCION_0, MENU_STATE_OPCION_1, MENU_STATE_OPCION_2, MENU_STATE_↵`
`OPCION_3,`
`MENU_STATE_OPCION_4, MENU_STATE_MAX }`
- `enum MENU_WINDOWS {`
`MENU_WINDOW_HOME, MENU_WINDOW_DIFFICULTY, MENU_WINDOW_PAUSE, MENU_↵`
`WINDOW_GAME_OVER,`
`MENU_WINDOW_MAX }`
- `enum CAR_TYPE {`
`CAR_BLUE = 0, CAR_POLICE, CAR_YELLOW, TRUCK_FIRE,`
`TRUCK, CAR_TYPE_N }`

Functions

- `int get_rand_between (int low, int high)`
Devuelve un random entre dos numeros dados.
- `bool collide (int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2)`
Comprueba colición de hitboxes rectangulares.
- `bool collideShort (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh)`
Similar a 'collide', pero pide ancho y alto en vez de segundas coordenadas.
- `bool inside (int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2)`
Detecta si un rectángulo está dentro de otro.
- `bool insideShort (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh)`

Similar a 'inside', pero pide ancho y alto en vez de segundas coordenadas.

- bool `insideShortScaled` (int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh, float scale)

Similar a 'insideShort', pero permite setear cuan dentro debe estar una box dentro de la otra para tomarla como tal.

- int `mapInt` (int source, int min_in, int max_in, int min_out, int max_out)

Toma un valor comprendido dentro de un rango (in) y lo devuelve (mapea) a otro rango (out)

- `pair_xy_t getXFromFrogFrame` (int frame)

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la rana.

- `pair_xy_t getXFromTurtleFrame` (int frame)

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la tortuga.

- `pair_xy_t getXFromCarFrame` (int frame)

Devuelve par de coordenadas xy topleft de un frame dado del sprite del auto.

- `pair_xy_t getXFromCoinFrame` (int frame)

Devuelve par de coordenadas xy topleft de un frame dado del sprite de coin.

- `pair_xy_t getXFromSplashFrame` (int frame)

Devuelve par de coordenadas xy topleft de un frame dado del sprite de splash.

- bool `matchUInt` (unsigned int val, const unsigned int *array)

Verifica si un numero coincide con alguno de un array dado (uints)

Variables

- const unsigned int `lanes_logs` [LANES_LOG_TOTAL]

Filas para troncos, referenciadas a 0.

- const unsigned int `lanes_turtles` [LANES_TURTLE_TOTAL]

Filas para tortugas, referenciadas a 0.

- const unsigned int `lanes_cars` [LANES_CAR_TOTAL]

Filas para autos, referenciadas a 0.

- const unsigned int `goal_cols` [MAX_GOALS]

Columnas para puntos de llegada, referenciadas a 0.

4.29.1 Detailed Description

Author

your name (`you@domain.com`)

Version

0.1

Date

2022-01-10

Copyright

Copyright (c) 2022

4.29.2 Function Documentation

4.29.2.1 collide()

```
bool collide (  
    int ax1,  
    int ay1,  
    int ax2,  
    int ay2,  
    int bx1,  
    int by1,  
    int bx2,  
    int by2 )
```

Comprueba colisión de hitboxes rectangulares.

Parameters

<i>ax1</i>	topleft corner de a (x)
<i>ay1</i>	topleft corner de a (y)
<i>ax2</i>	bottomright corner de a (x)
<i>ay2</i>	bottomright corner de a (y)
<i>bx1</i>	topleft corner de b (x)
<i>by1</i>	topleft corner de b (y)
<i>bx2</i>	bottomright corner de b (x)
<i>by2</i>	bottomright corner de b (y)

Returns

true Colisión

false No colisión

4.29.2.2 collideShort()

```
bool collideShort (  
    int ax,  
    int ay,  
    int aw,  
    int ah,  
    int bx,  
    int by,  
    int bw,  
    int bh )
```

Similar a 'collide', pero pide ancho y alto en vez de segundas coordenadas.

Parameters

<i>ax</i>	toleft corner x de a
<i>ay</i>	toleft corner y de a
<i>aw</i>	ancho de a
<i>ah</i>	alto de a
<i>bx</i>	toleft corner x de b
<i>by</i>	toleft corner y de b
<i>bw</i>	ancho de b
<i>bh</i>	alto de b

Returns

true Collision
false No colision

4.29.2.3 get_rand_between()

```
int get_rand_between (
    int low,
    int high )
```

Devuelve un randon entre dos numeros dados.

Parameters

<i>low</i>	Valor inferior
<i>high</i>	Valor superior

Returns

int Valor random

4.29.2.4 getXYFromCarFrame()

```
pair_xy_t getXYFromCarFrame (
    int frame )
```

Devuelve par de coordenadas xy toleft de un frame dado del sprite del auto.

Parameters

<i>frame</i>	Numero de frame (0 a CAR_TYPE_N - 1)
--------------	--------------------------------------

Returns

`pair_xy_t` Par de coordenadas

4.29.2.5 getXYFromCoinFrame()

```
pair_xy_t getXYFromCoinFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de coin.

Parameters

<i>frame</i>	Numero de frame (0 a SPRITE_COIN_FRAMES - 1)
--------------	--

Returns

`pair_xy_t` Par de coordenadas

4.29.2.6 getXYFromFrogFrame()

```
pair_xy_t getXYFromFrogFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la rana.

Parameters

<i>frame</i>	Numero de frame (0 a FROG_FRAMES - 1)
--------------	---------------------------------------

Returns

`pair_xy_t` Par de coordenadas

4.29.2.7 getXYFromSplashFrame()

```
pair_xy_t getXYFromSplashFrame (  
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de splash.

Parameters

<i>frame</i>	Numero de frame (0 a SPRITE_SPLASH_FRAMES - 1)
--------------	--

Returns

`pair_xy_t` Par de coordenandas

4.29.2.8 `getXYFromTurtleFrame()`

```
pair_xy_t getXYFromTurtleFrame (
    int frame )
```

Devuelve par de coordenadas xy topleft de un frame dado del sprite de la tortuga.

Parameters

<i>frame</i>	Numero de frame (0 a TURTLE_FRAMES - 1)
--------------	---

Returns

`pair_xy_t` Par de coordenadas

4.29.2.9 `inside()`

```
bool inside (
    int ax1,
    int ay1,
    int ax2,
    int ay2,
    int bx1,
    int by1,
    int bx2,
    int by2 )
```

Detecta si un rectángulo está dentro de otro.

Parameters

<i>ax1</i>	topleft corner de big (x)
<i>ay1</i>	topleft corner de big (y)
<i>ax2</i>	bottomright corner de big (x)
<i>ay2</i>	bottomright corner de big (y)
<i>bx1</i>	topleft corner de small (x)
<i>by1</i>	topleft corner de small (y)
<i>bx2</i>	bottomright corner de small (x)
<i>by2</i>	bottomright corner de small (y)

Returns

true Está dentro
false Está fuera

4.29.2.10 insideShort()

```
bool insideShort (
    int ax,
    int ay,
    int aw,
    int ah,
    int bx,
    int by,
    int bw,
    int bh )
```

Similar a 'inside', pero pide ancho y alto en vez de segundas coordenadas.

Parameters

<i>ax</i>	opleft corner x de big
<i>ay</i>	opleft corner y de big
<i>aw</i>	ancho de big
<i>ah</i>	alto de big
<i>bx</i>	opleft corner x de big
<i>by</i>	opleft corner y de big
<i>bw</i>	ancho de big
<i>bh</i>	alto de big

Returns

true Esta dentro
false Esta fuera

4.29.2.11 insideShortScaled()

```
bool insideShortScaled (
    int ax,
    int ay,
    int aw,
    int ah,
    int bx,
    int by,
    int bw,
    int bh,
    float scale )
```

Similar a 'insideShort', pero permite setear cuan dentro debe estar una box dentro de la otra para tomarla como tal.

Parameters

<i>ax</i>	topleft corner x de big
<i>ay</i>	topleft corner y de big
<i>aw</i>	ancho de big
<i>ah</i>	alto de big
<i>bx</i>	topleft corner x de big
<i>by</i>	topleft corner y de big
<i>bw</i>	ancho de big
<i>bh</i>	alto de big
<i>scale</i>	Factor de insercion. Entre 0.0 (nada metido) y 1.0 (completamente metido). Otro valor devuelve false

Returns

true
false

4.29.2.12 mapInt()

```
int mapInt (
    int source,
    int min_in,
    int max_in,
    int min_out,
    int max_out )
```

Toma un valor comprendido dentro de un rango (in) y lo devuelve (mapea) a otro rango (out)

@source <https://stackoverflow.com/questions/5731863/mapping-a-numeric-range-onto-another>

Parameters

<i>source</i>	Valor a mapear
<i>min_in</i>	Límite inferior del rango de entrada
<i>max_in</i>	Límite superior del rango de entrada
<i>min_out</i>	Límite inferior del rango de salida
<i>max_out</i>	Límite superior del rango de salida

Returns

int Valor mapeado

4.29.2.13 matchUInt()

```
bool matchUInt (
    unsigned int val,
    const unsigned int * array )
```

Verifica si un numero coincide con alguno de un array dado (uints)

Parameters

<i>val</i>	Numero a verificar
<i>array</i>	Array de numeros

Returns

true Existe
false No existe

4.29.3 Variable Documentation

4.29.3.1 goal_cols

```
const unsigned int goal_cols[MAX_GOALS] [extern]
```

Columnas para puntos de llegada, referenciadas a 0.

Columnas para puntos de llegada, referenciadas a 0.

4.29.3.2 lanes_cars

```
const unsigned int lanes_cars[LANES_CAR_TOTAL] [extern]
```

Filas para autos, referenciadas a 0.

Filas para autos, referenciadas a 0.

4.29.3.3 lanes_logs

```
const unsigned int lanes_logs[LANES_LOG_TOTAL] [extern]
```

Filas para troncos, referenciadas a 0.

Filas para troncos, referenciadas a 0.

4.29.3.4 lanes_turtles

```
const unsigned int lanes_turtles[LANES_TURTLE_TOTAL] [extern]
```

Filas para tortugas, referenciadas a 0.

Filas para tortugas, referenciadas a 0.

4.30 geometry.h

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

```

1
12 #ifndef _GEOMETRY_H_
13 #define _GEOMETRY_H_
14
15 /*****
16  * INCLUDE HEADER FILES
17  *****/
18
19 #include <time.h>
20 #include <stdbool.h>
21
22 /*****
23  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
24  *****/
25
26 #define DISPLAY_W 690
27 #define DISPLAY_H 644
28 #define ROWS 14
29 #define COLS 15
30 #define CELL_H 46
31 #define CELL_W 46
32
33 #define MAX_LANES (ROWS - 1) //-1 por la primera que es para HUD
34
35 #define LANES_CAR_TOTAL 5
36
37 #define LANES_LOG_TOTAL 3
38
39 #define LANES_TURTLE_TOTAL 2
40
41 // Coordenadas de la celda topleft (en su vértice topleft)
42 #define CELL_TOPLEFT_X 0
43 #define CELL_TOPLEFT_Y 0
44
45 // Coordenadas de inicio ("ultima fila, columna 8"; referidas a topleft)
46 #define CELL_START_X (CELL_TOPLEFT_X + CELL_W * ((COLS - 1) / 2))
47 #define CELL_START_Y (CELL_TOPLEFT_Y + CELL_H * (ROWS - 1))
48
49 #define FROG_W 30
50 #define FROG_H 30
51
52 #define FROG_OFFSET_X (CELL_W / 2 - FROG_W / 2)
53 #define FROG_OFFSET_Y (CELL_H / 2 - FROG_H / 2)
54
55 // Coordenadas inicio rana
56 #define CELL_START_FROG_X (CELL_START_X + FROG_OFFSET_X)
57 #define CELL_START_FROG_Y (CELL_START_Y + FROG_OFFSET_Y)
58
59 // Para los sprites de la rana
60 #define FROG_FRAMES 8
61 #define SPRITE_SIZE_FROG_STATIC_H FROG_H
62 #define SPRITE_SIZE_FROG_STATIC_W FROG_W
63 #define SPRITE_SIZE_FROG_DYNAMIC_LONG 46
64 #define SPRITE_SIZE_FROG_DYNAMIC_SHORT FROG_W
65
66 // Para los pasos de la rana
67 #define STEP_FULL_SIZE CELL_H
68 #define STEP_RATIO (CELL_H / 3)
69 #define STEP_FRACTION_SIZE (STEP_FULL_SIZE / STEP_RATIO)
70
71 // Bordes para la rana en el mapa
72 #define FROG_MAX_X (DISPLAY_W - (CELL_W - FROG_OFFSET_X))
73 #define FROG_MAX_Y (DISPLAY_H - (CELL_H - FROG_OFFSET_Y))
74 #define FROG_MIN_X (CELL_TOPLEFT_X + FROG_OFFSET_X)
75 #define FROG_MIN_Y (CELL_TOPLEFT_Y + FROG_OFFSET_Y + CELL_H)
76
77 // Troncos
78 #define LOG_W (4 * CELL_W)
79 #define LOG_H 40
80 #define LOG_OFFSET_X 0
81 #define LOG_OFFSET_Y (CELL_H / 2 - LOG_H / 2)
82
83 // Autos
84 #define CAR_W CELL_W + 26
85 #define CAR_TRUCK_FIRE_W (3 * CELL_W)
86 #define CAR_TRUCK_W (4 * CELL_W)
87 #define CAR_H 40
88 #define CAR_OFFSET_X 0
89 #define CAR_OFFSET_Y (CELL_H / 2 - CAR_H / 2)
90
91 // Tortugas
92 #define TURTLE_FRAMES 11 // 11 frames distintos tiene la animación completa

```

```

93 #define TURTLE_SIDE CELL_W
94 #define TURTLE_FRAME_OFFSET_XY (CELL_W / 2 - TURTLE_SIDE / 2)
95
96 // Corazon (vidas)
97 #define SPRITE_SIZE_HEART 20 // cuadrado
98
99 #define SPRITE_DEAD_SIZE 35 // cuadrado
100 #define SPRITE_DEAD_OFFSET (CELL_W / 2 - SPRITE_DEAD_SIZE / 2)
101
102 #define SPRITE_COIN_FRAMES 6
103 #define SPRITE_COIN_SIDE 24
104 #define SPRITE_COIN_OFFSET_XY (CELL_W / 2 - SPRITE_COIN_SIDE / 2)
105
106 #define SPRITE_SPLASH_FRAMES 6
107 #define SPRITE_SPLASH_W 98
108 #define SPRITE_SPLASH_H 68
109 #define SPRITE_SPLASH_OFFSET_X (CELL_W / 2 - SPRITE_SPLASH_W / 2)
110 #define SPRITE_SPLASH_OFFSET_Y (CELL_W / 2 - SPRITE_SPLASH_H / 2)
111
112 #define SPRITE_BORDER_START_X 0
113 #define SPRITE_BORDER_START_Y CELL_H
114
115 #define MENU_OPTION_TOPLEFT_X 45
116 #define MENU_OPTION_TOPLEFT_Y 72
117 #define MENU_OPTION_DELTA_Y 100
118 #define MENU_OPTION_W 600
119 #define MENU_OPTION_H 75
120
121 #define CREDITS_SCREEN_LENGTH 2576
122 #define CREDITS_SCREEN_START 0
123 #define CREDITS_SCREEN_FINAL (CREDITS_SCREEN_LENGTH - DISPLAY_H)
124
125 // Factor que determina cuando considerar que un bloque esta dentro de otro (ver 'inside_short_scaled')
126 #define INSERTION_FACTOR (double)0.5
127
128 #define GOAL_ROW_OFFSET_Y_FIX 5 // baja un poco mas en Y
129 #define GOAL_ROW_MARGIN_TO_REACH 5 // holgura para meterse a uno de los goals
130
131 /*****
132  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
133  *****/
134
135 typedef struct
136 {
137     int x;
138     int y;
139 } pair_xy_t;
140
141
142 enum GOALS
143 {
144     GOAL_LEFT,
145     GOAL_LEFT_MID,
146     GOAL_MID,
147     GOAL_RIGHT_MID,
148     GOAL_RIGHT,
149     MAX_GOALS
150 };
151
152 enum DIRECTIONS
153 {
154     DIRECTION_NONE,
155     DIRECTION_UP,
156     DIRECTION_RIGHT,
157     DIRECTION_LEFT,
158     DIRECTION_DOWN
159 };
160
161 enum MENU_STATES
162 {
163     MENU_STATE_OPCION_0,
164     MENU_STATE_OPCION_1,
165     MENU_STATE_OPCION_2,
166     MENU_STATE_OPCION_3,
167     MENU_STATE_OPCION_4,
168     MENU_STATE_MAX
169 };
170
171 enum MENU_WINDOWS
172 {
173     MENU_WINDOW_HOME,
174     MENU_WINDOW_DIFFICULTY,
175     MENU_WINDOW_PAUSE,
176     MENU_WINDOW_GAME_OVER,
177     MENU_WINDOW_MAX
178 };
179

```

```

180 typedef enum CAR_TYPE
181 {
182     CAR_BLUE = 0,
183     CAR_POLICE,
184     CAR_YELLOW,
185     TRUCK_FIRE,
186     TRUCK,
187     CAR_TYPE_N
188 } CAR_TYPE;
189
190 /*****
191  * VARIABLE PROTOTYPES WITH GLOBAL SCOPE
192  *****/
193
194 extern const unsigned int lanes_logs[LANES_LOG_TOTAL];
195
196 extern const unsigned int lanes_turtles[LANES_TURTLE_TOTAL];
197
198 extern const unsigned int lanes_cars[LANES_CAR_TOTAL];
199
200 extern const unsigned int goal_cols[MAX_GOALS];
201
202 /*****
203  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
204  *****/
205
206 int get_rand_between(int low, int high);
207
208 bool collide(int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2);
209
210 bool collideShort(int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh);
211
212 bool inside(int ax1, int ay1, int ax2, int ay2, int bx1, int by1, int bx2, int by2);
213
214 bool insideShort(int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh);
215
216 bool insideShortScaled(int ax, int ay, int aw, int ah, int bx, int by, int bw, int bh, float scale);
217
218 int mapInt(int source, int min_in, int max_in, int min_out, int max_out);
219
220 pair_xy_t getXYFromFrogFrame(int frame);
221
222 pair_xy_t getXYFromTurtleFrame(int frame);
223
224 pair_xy_t getXYFromCarFrame(int frame);
225
226 pair_xy_t getXYFromCoinFrame(int frame);
227
228 pair_xy_t getXYFromSplashFrame(int frame);
229
230 bool matchUInt(unsigned int val, const unsigned int *array);
231
232 /*****
233  *****/
234
235 #endif // _GEOMETRY_H_

```

4.31 src/platform/pc/input.c File Reference

```

#include "../input.h"
#include "allegro_stuff.h"
#include "entities.h"
#include "game_data.h"

```

Functions

- void **iniciarEntradas** (void)
Inicializa las entradas de la plataforma.
- event_t **leerEntradas** (void)
Devuelve una entrada válida.

4.31.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-22

Copyright

Copyright (c) 2022

4.31.2 Function Documentation

4.31.2.1 leerEntradas()

```
event_t leerEntradas (  
    void )
```

Devuelve una entrada válida.

Returns

event_t enum eventos_tecla

4.32 src/platform/rpi/input.c File Reference

Archivo para manejo del joystick en RPI.

```
#include "../..input.h"  
#include "joydrv.h"  
#include <stdint.h>
```

Functions

- void **iniciarEntradas** ()
Inicializa las entradas de la plataforma.
- event_t **leerEntradas** ()
Devuelve una entrada válida.

4.32.1 Detailed Description

Archivo para manejo del joystick en RPI.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.32.2 Function Documentation

4.32.2.1 leerEntradas()

```
event_t leerEntradas (  
    void )
```

Devuelve una entrada válida.

Returns

event_t enum eventos_tecla

4.33 src/platform/pc/menu.c File Reference

```
#include "../menu.h"  
#include "../sound.h"  
#include "allegro_stuff.h"  
#include "geometry.h"  
#include "game_data.h"
```

Data Structures

- struct [window_t](#)
- struct [menu_t](#)

Macros

- #define **STATS_X_COORD** 20
- #define **STATS_Y_COORD_START** (DISPLAY_H / 2 + 50)

Functions

- void **iniciarMenu** (void)
Inicia el menu.
- void **destruirMenu** (void)
Destruye del menu.
- void **setMenu** (int *a, unsigned int size)
Selecciona un menu.
- void **setOpcion** (int opc)
Selecciona una opcion del menu.
- int **getOpcion** (void)
Devuelve la opcion actual del menu.
- void **subirOpcion** (void)
Selecciona la opcion superior a la actual.
- void **bajarOpcion** (void)
Selecciona la opcion inferior a la actual.
- void **moverOpcionActual** (void)

4.33.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-22

Copyright

Copyright (c) 2022

4.33.2 Function Documentation

4.33.2.1 getOpcion()

```
int getOpcion (  
    void )
```

Devuelve la opcion actual del menu.

Returns

int Opcion seleccionada actualmente

4.33.2.2 setMenu()

```
void setMenu (  
    int * a,  
    unsigned int size )
```

Selecciona un menu.

Parameters

<i>a</i>	Puntero a textos del menu
<i>size</i>	Opciones del menu

4.33.2.3 setOpcion()

```
void setOpcion (
    int opc )
```

Selecciona una opcion del menu.

Parameters

<i>opc</i>	Opcion a seleccionar
------------	----------------------

4.34 src/platform/rpi/menu.c File Reference

Archivo para manejo de los menús en RPI.

```
#include "../menu.h"
#include "../display.h"
#include "../sound.h"
#include <stdlib.h>
```

Functions

- void [setMenu](#) (int *a, unsigned int size)
Selecciona un menu.
- void [setOpcion](#) (int opc)
Selecciona una opcion del menu.
- int [getOpcion](#) ()
Devuelve la opcion actual del menu.
- void [subirOpcion](#) ()
Selecciona la opcion superior a la actual.
- void [bajarOpcion](#) ()
Selecciona la opcion inferior a la actual.
- void [iniciarMenu](#) ()
Inicia el menu.
- void [destruirMenu](#) ()
Destruye del menu.

4.34.1 Detailed Description

Archivo para manejo de los menús en RPI.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.34.2 Function Documentation

4.34.2.1 `getOpcion()`

```
int getOpcion (  
    void )
```

Devuelve la opcion actual del menu.

Returns

int Opcion seleccionada actualmente

4.34.2.2 `setMenu()`

```
void setMenu (  
    int * a,  
    unsigned int size )
```

Selecciona un menu.

Parameters

<i>a</i>	Puntero a textos del menu
<i>size</i>	Opciones del menu

4.34.2.3 `setOpcion()`

```
void setOpcion (  
    int opc )
```

Selecciona una opcion del menu.

Parameters

<i>opc</i>	Opcion a seleccionar
------------	----------------------

4.35 src/platform/pc/nombre.c File Reference

```
#include "../.. /nombre.h"
#include "allegro_stuff.h"
#include "game_data.h"
#include <string.h>
```

Macros

- `#define NAME_TOPLEFT_X 55`
- `#define NAME_TOPLEFT_Y 312`

Functions

- void **nuevoNombre** (void)
Se ejecuta una vez al ingresar a poner un nuevo nombre.
- void **subirLetra** (void)
Selecciona la siguiente letra superior.
- void **bajarLetra** (void)
Selecciona la letra inferior.
- void **siguienteLetra** (void)
Confirma la letra y pasa a seleccionar la siguiente.
- void **agregarLetra** (void)
Confirma la letra.
- void **subirNombre** (void)
- char * **devolverNombre** (void)
Devuelve puntero al nombre.

4.35.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-22

Copyright

Copyright (c) 2022

4.35.2 Function Documentation

4.35.2.1 devolverNombre()

```
char * devolverNombre (  
    void )
```

Devuelve puntero al nombre.

Returns

char* Puntero al nombre

4.36 src/platform/rpi/nombre.c File Reference

Archivo para manejo de información en el ingreso del nombre.

```
#include "../..//nombre.h"  
#include "mensajes.h"  
#include "../..//display.h"
```

Functions

- void **nuevoNombre** ()
Se ejecuta una vez al ingresar a poner un nuevo nombre.
- void **subirLetra** ()
Selecciona la siguiente letra superior.
- void **bajarLetra** ()
Selecciona la letra inferior.
- void **siguienteLetra** ()
Confirma la letra y pasa a seleccionar la siguiente.
- void **agregarLetra** (void)
Confirma la letra.
- char * **devolverNombre** (void)
Devuelve puntero al nombre.

Variables

- matriz_t **disp_matriz**

4.36.1 Detailed Description

Archivo para manejo de información en el ingreso del nombre.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.36.2 Function Documentation

4.36.2.1 devolverNombre()

```
char * devolverNombre (  
    void )
```

Devuelve puntero al nombre.

Returns

char* Puntero al nombre

4.37 src/platform/pc/sound.c File Reference

```
#include "../..//sound.h"  
#include "allegro_stuff.h"
```

Functions

- bool [iniciarSonido](#) (void)
Inicializa el sonido de la plataforma.
- void **destruirSonido** (void)
Desinicializa el sonido de la plataforma.
- void **pausarMusica** (void)
Pausa la musica actual.
- void [reproducirMusica](#) (int musica)
Pone a reproducir una musica dada.
- void [reproducirEfecto](#) (int sonido)
Pone a reproducir un efecto dado.

4.37.1 Detailed Description

Author

your name (you@domain.com)

Version

0.1

Date

2022-01-27

Copyright

Copyright (c) 2022

4.37.2 Function Documentation

4.37.2.1 iniciarSonido()

```
bool iniciarSonido (
    void )
```

Inicializa el sonido de la plataforma.

Returns

true Exito
false Error

4.37.2.2 reproducirEfecto()

```
void reproducirEfecto (
    int efecto )
```

Pone a reproducir un efecto dado.

Parameters

<i>int</i>	num efectos
------------	-------------

4.37.2.3 reproducirMusica()

```
void reproducirMusica (
    int musica )
```

Pone a reproducir una musica dada.

Parameters

<i>int</i>	enum musica
------------	-------------

4.38 src/platform/rpi/sound.c File Reference

Archivo para manejo del sonido en RPI.

```
#include "../..//sound.h"
#include "simpleSDL2audio/audio.h"
#include <stdio.h>
```

Macros

- #define **MUSICA_DIR** "../res/sounds/streams"
- #define **EFFECTOS_DIR** "../res/sounds/samples"

Functions

- bool **iniciarSonido** (void)
Inicializa el sonido de la plataforma.
- void **destruirSonido** (void)
Desinicializa el sonido de la plataforma.
- void **pausarMusica** (void)
Pausa la musica actual.
- void **reproducirMusica** (int m)
Pone a reproducir una musica dada.
- void **reproducirEfecto** (int e)
Pone a reproducir un efecto dado.

4.38.1 Detailed Description

Archivo para manejo del sonido en RPI.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.38.2 Function Documentation

4.38.2.1 iniciarSonido()

```
bool iniciarSonido (
    void )
```

Inicializa el sonido de la plataforma.

Returns

true Exit
false Error

4.38.2.2 reproducirEfecto()

```
void reproducirEfecto (
    int efecto )
```

Pone a reproducir un efecto dado.

Parameters

<i>int</i>	num efectos
------------	-------------

4.38.2.3 reproducirMusica()

```
void reproducirMusica (
    int musica )
```

Pone a reproducir una musica dada.

Parameters

<i>int</i>	enum musica
------------	-------------

4.39 src/platform/rpi/bitmap.h File Reference

Encabezado del archivo para manejo de matrices 16x16.

```
#include <stdint.h>
```

Macros

- `#define CANT_FILAS 16`
- `#define CANT_COLUMNAS 16`

Typedefs

- `typedef uint16_t matriz_t[CANT_FILAS]`

Functions

- void `printMatriz` (`matriz_t A`)
Imprime una matriz en consola (para debug)
- void `limpiarMatriz` (`matriz_t A`)
Borra el contenido de una matriz.
- void `copiarMatriz` (`matriz_t destino, const matriz_t desde`)
Copia el contenido de una matriz en otra.
- void `matrizAnd` (`matriz_t A, matriz_t B`)
Dadas dos matrices A y B, se hará la operación "A &= B".
- void `matrizOr` (`matriz_t A, matriz_t B`)
Dadas dos matrices A y B, se hará la operación "A |= B".
- void `matrizNot` (`matriz_t A`)
Dadas una matriz A, se hará la operación "A = ~A".
- void `matrizXor` (`matriz_t A, matriz_t B`)
Dadas dos matrices A y B, se hará la operación "A ^= B".

4.39.1 Detailed Description

Encabezado del archivo para manejo de matrices 16x16.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.39.2 Function Documentation

4.39.2.1 `copiarMatriz()`

```
void copiarMatriz (  
    matriz_t destino,  
    const matriz_t desde )
```

Copia el contenido de una matriz en otra.

Parameters

<i>destino</i>	
<i>desde</i>	

4.39.2.2 limpiarMatriz()

```
void limpiarMatriz (
    matriz_t A )
```

Borra el contenido de una matriz.

Parameters

<i>A</i>	
----------	--

4.39.2.3 matrizAnd()

```
void matrizAnd (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación "A &= B".

Parameters

<i>A</i>	
<i>B</i>	

4.39.2.4 matrizNot()

```
void matrizNot (
    matriz_t A )
```

Dadas una matriz A, se hará la operación "A = ~A".

Parameters

<i>A</i>	
----------	--

4.39.2.5 matrizOr()

```
void matrizOr (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación " $A \mid B$ ".

Parameters

A	
B	

4.39.2.6 matrizXor()

```
void matrizXor (
    matriz_t A,
    matriz_t B )
```

Dadas dos matrices A y B, se hará la operación " $A \wedge B$ ".

Parameters

A	
B	

4.39.2.7 printMatriz()

```
void printMatriz (
    matriz_t A )
```

Imprime una matriz en consola (para debug)

Parameters

A	
---	--

4.40 bitmap.h

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

```
1
8 #ifndef _BITMAP_H_
9 #define _BITMAP_H_
10
```

```

11 /*****
12  * INCLUDE HEADER FILES
13  *****/
14
15 #include <stdint.h>
16
17 /*****
18  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
19  *****/
20
21 #define CANT_FILAS 16
22 #define CANT_COLUMNAS 16
23
24 /*****
25  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
26  *****/
27
28 typedef uint16_t matriz_t[CANT_FILAS]; //se define el tipo de dato para trabajar en el display, cada
    elemento del array es una fila
29
30 /*****
31  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
32  *****/
33
39 void printMatriz(matriz_t A);
40
46 void limpiarMatriz(matriz_t A);
47
54 void copiarMatriz(matriz_t destino, const matriz_t desde);
55
62 void matrizAnd(matriz_t A, matriz_t B);
63
70 void matrizOr(matriz_t A, matriz_t B);
71
77 void matrizNot(matriz_t A);
78
85 void matrizXor(matriz_t A, matriz_t B);
86
87 /*****
88  *****/
89
90 #endif // _BITMAP_H_

```

4.41 disdrv.h

```

1
12 #ifndef DISDRV_H
13 #define DISDRV_H
14
15 /*****
16  * INCLUDE HEADER FILES
17  *****/
18
19 #include <stdint.h>
20
21 /*****
22  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
23  *****/
24
25 #define DISP_CANT_X_DOTS 16
26 #define DISP_CANT_Y_DOTS 16
27
28 #define DISP_MIN 0
29 #define DISP_MAX_X (DISP_MIN + DISP_CANT_X_DOTS - 1) // = 15
30 #define DISP_MAX_Y (DISP_MIN + DISP_CANT_Y_DOTS - 1) // = 15
31
32 /*****
33  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
34  *****/
35
36 typedef struct
37 {
38     uint8_t x; // coordenada x del display
39     uint8_t y; // coordenada y del display
40 } dcoord_t;
41
42 typedef enum
43 {
44     D_OFF,
45     D_ON
46 } dlevel_t; // Valores posibles para cada LED
47
48 /*****

```

```

49  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
50  *****/
51
52 // Display Services
53
54 void disp_init(void);
55
56 void disp_clear(void);
57
58 void disp_write(dcoord_t coord, dlevel_t val);
59
60 void disp_update(void);
61
62 /*****
63  *****/
64 #endif // DISDRV_H

```

4.42 joydrv.h

```

1
2 #ifndef JOYDRV_H
3 #define JOYDRV_H
4
5 /*****
6  * INCLUDE HEADER FILES
7  *****/
8
9 #include <stdint.h>
10
11 /*****
12  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
13  *****/
14
15 // Las coordenadas del joystick varían entre -128 y 127 para cada coordenada
16 #define JOY_MAX_POS 127
17 #define JOY_MAX_NEG -128
18
19 /*****
20  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
21  *****/
22
23 typedef struct
24 {
25     int8_t x;
26     int8_t y;
27 } jcoord_t;
28
29 typedef enum
30 {
31     J_NOPRESS,
32     J_PRESS
33 } jswitch_t;
34
35 /*****
36  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
37  *****/
38
39 void joy_init(void);
40
41 int joy_update(void); // void
42
43 jcoord_t joy_get_coord(void);
44
45 jswitch_t joy_get_switch(void);
46
47 /*****
48  *****/
49 #endif // JOYDRV_H

```

4.43 src/platform/rpi/mensajes.c File Reference

Permite codificar strings en formato renglon para mostrar en display.

```

#include "mensajes.h"
#include <stdio.h>

```

```
#include <string.h>
#include <stdlib.h>
```

Macros

- `#define INDEX_ESPACIO 26`
- `#define INDEX_CERO 27`
- `#define INDEX_FULL 37`
- `#define CANT_SIMBOLOS 38`
- `#define PEDIR_FULL -1`

Functions

- void `printRenglon (renglon_t r)`
imprime dos renglon en consola (para debuggear)
- void `borrarRenglon (renglon_t r)`
elimina el contenido del Renglon
- void `renglonShiftDer (renglon_t r, uint16_t s)`
Desplaza a la derecha el contenido de un Renglon.
- void `renglonShiftIzq (renglon_t r, uint16_t s)`
Desplaza a la izquierda el contenido de un Renglon.
- void `renglonOr (renglon_t r, renglon_t s)`
Se ejecuta la operación "r |= s".
- void `renglonAnd (renglon_t r, renglon_t s)`
Se ejecuta la operación "r &= s".
- void `renglonNot (renglon_t r)`
Se invierte el contenido del Renglon (se obtiene el complemento)
- void `copiarRenglon (renglon_t r1, const renglon_t r2)`
copia el contenido de r2 en r1
- void `copiarMatrizRenglon (matriz_t m, const renglon_t r, int pos)`
copia un Renglon sobre una Matriz a partir de una fila especificada
- bool `renglonIzquierdoLibre (mensaje_t *msj)`
indica si la parte izquierda del renglón de un mensaje está vacía
- void `charARenglon (char c, renglon_t r)`
Convierte un caracter a Renglon, ubicándolo en el extremo izquierdo del Renglon.
- void `charAMatriz (char c, matriz_t m, const int coord[])`
Escribe un caracter en una matriz, a partir de las posiciones (x,y) (Extremo superior izdo)
- void `uintARenglon (uint16_t n, renglon_t r)`
Convierte un entero no signado a Renglon, ubicándolo en el extremo izquierdo del renglon.
- void `reemplazarLetra (renglon_t r, char c, int j)`
(re)escribe sobre el Renglon un caracter dado a partir de la columna j
- `mensaje_t mensaje` (char *msj, int pos, bool repetir)
constructor de la variable mensaje_t
- void `moverMensaje (mensaje_t *msj)`
desplaza el contenido del Renglon doble hacia la izquierda
- void `concatenarLetraMensaje (char c, mensaje_t *msj)`
agrega una letra al string del mensaje y también al renglon
- void `reemplazarUltLetraMensaje (char c, mensaje_t *msj)`
reemplaza la última letra en el string del mensaje y también del renglon

4.43.1 Detailed Description

Permite codificar strings en formato renglon para mostrar en display.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.43.2 Function Documentation

4.43.2.1 borrarRenglon()

```
void borrarRenglon (
    renglon_t r )
```

elimina el contenido del Renglon

Parameters

<i>r</i>	Renglon a limpiar
----------	-------------------

4.43.2.2 charAMatriz()

```
void charAMatriz (
    char c,
    matriz_t m,
    const int coord[] )
```

Escribe un caracter en una matriz, a partir de las posiciones (x,y) (Extremo superior izdo)

Parameters

<i>c</i>	caracter
<i>m</i>	Matriz
<i>x</i>	
<i>y</i>	

4.43.2.3 charARenglon()

```
void charARenglon (
    char c,
    renglon_t r )
```

Convierte un caracter a Renglon, ubicándolo en el extremo izquierdo del Renglon.

Parameters

<i>c</i>	caracter
<i>r</i>	Renglon

4.43.2.4 concatenarLetraMensaje()

```
void concatenarLetraMensaje (
    char c,
    mensaje_t * msj )
```

agrega una letra al string del mensaje y también al renglon

Parameters

<i>c</i>	
<i>msj</i>	

4.43.2.5 copiarMatrizRenglon()

```
void copiarMatrizRenglon (
    matriz_t m,
    const renglon_t r,
    int pos )
```

copia un Renglon sobre una Matriz a partir de una fila especificada

Parameters

<i>m</i>	Matriz destino
<i>r</i>	Renglon origen
<i>pos</i>	Fila de inicio

4.43.2.6 copiarRenglon()

```
void copiarRenglon (
    renglon_t r1,
    const renglon_t r2 )
```

copia el contenido de r2 en r1

Parameters

<i>r1</i>	Renglon destino
<i>r2</i>	Renglon origen

4.43.2.7 mensaje()

```
mensaje_t mensaje (
    char * msj,
    int pos,
    bool repetir )
```

constructor de la variable mensaje_t

Parameters

<i>msj</i>	String que se desea convertir a mensaje
<i>pos</i>	fila sobre la que se deberá mostrar en display
<i>repetir</i>	si se repetirá una vez terminado de mostrar

Returns

mensaje_t

4.43.2.8 moverMensaje()

```
void moverMensaje (
    mensaje_t * msj )
```

desplaza el contenido del Renglon doble hacia la izquierda

Parameters

<i>msj</i>	puntero a la variable mensaje_t
------------	------------------------------------

4.43.2.9 printRenglon()

```
void printRenglon (
    renglon_t r )
```

imprime dos renglon en consola (para debuggear)

Parameters

<i>r</i>	renglon a imprimir
----------	--------------------

4.43.2.10 reemplazarLetra()

```
void reemplazarLetra (
    renglon_t r,
    char c,
    int j )
```

(re)escribe sobre el Renglon un caracter dado a partir de la columna j

Parameters

<i>r</i>	Renglon
<i>c</i>	caracter
<i>j</i>	columna sobre la que se quiere escribir

4.43.2.11 reemplazarUltLetraMensaje()

```
void reemplazarUltLetraMensaje (
    char c,
    mensaje_t * msj )
```

reemplaza la última letra en el string del mensaje y también del renglon

Parameters

<i>c</i>	
<i>msj</i>	

4.43.2.12 renglonAnd()

```
void renglonAnd (
```

```
    renglon_t r,  
    renglon_t s )
```

Se ejecuta la operación "r &= s".

Parameters

<i>r</i>	primer operando AND
<i>s</i>	segundo operando AND

4.43.2.13 renglonIzquierdoLibre()

```
bool renglonIzquierdoLibre (  
    mensaje_t * msj )
```

indica si la parte izquierda del renglón de un mensaje está vacía

Parameters

<i>r</i>	Renglon a chequear
----------	--------------------

Returns

true
false

4.43.2.14 renglonNot()

```
void renglonNot (  
    renglon_t r )
```

Se invierte el contenido del Renglon (se obtiene el complemento)

Parameters

<i>r</i>	Renglon a invertir
----------	--------------------

4.43.2.15 renglonOr()

```
void renglonOr (  
    renglon_t r,  
    renglon_t s )
```

Se ejecuta la operación " $r \mid= s$ ".

Parameters

<i>r</i>	primer operando OR
<i>s</i>	segundo operando OR

4.43.2.16 renglonShiftDer()

```
void renglonShiftDer (
    renglon_t r,
    uint16_t s )
```

Desplaza a la derecha el contenido de un Renglon.

Parameters

<i>r</i>	Renglon a desplazar
<i>s</i>	cantidad de espacios

4.43.2.17 renglonShiftIzq()

```
void renglonShiftIzq (
    renglon_t r,
    uint16_t s )
```

Desplaza a la izquierda el contenido de un Renglon.

Parameters

<i>r</i>	Renglon a desplazar
<i>s</i>	cantidad de espacios

4.43.2.18 uintARenglon()

```
void uintARenglon (
    uint16_t n,
    renglon_t r )
```

Convierte un entero no signado a Renglon, ubicándolo en el extremo izquierdo del renglon.

Parameters

<i>n</i>	entero no signado de 16 bits
<i>r</i>	Renglon

4.44 src/platform/rpi/mensajes.h File Reference

Encabezado de mensajes, con definiciones sobre tipos de datos y funciones.

```
#include "bitmap.h"
#include <stdint.h>
#include <stdbool.h>
```

Data Structures

- union [renglon_t](#)
- struct [Mensaje](#)

Macros

- #define **TAM_REGLON** 5
- #define **POS_MSJ1** 2
- #define **POS_MSJ2** 9
- #define **POS_MSJ3** 5
- #define **L_MAX** 64

Typedefs

- typedef struct [Mensaje](#) **mensaje_t**

Functions

- void [printRenglon](#) ([renglon_t](#) r)
imprime dos renglon en consola (para debuggear)
- void [borrarRenglon](#) ([renglon_t](#) r)
elimina el contenido del Renglon
- void [renglonShiftDer](#) ([renglon_t](#) r, uint16_t s)
Desplaza a la derecha el contenido de un Renglon.
- void [renglonShiftIzq](#) ([renglon_t](#) r, uint16_t s)
Desplaza a la izquierda el contenido de un Renglon.
- void [renglonOr](#) ([renglon_t](#) r, [renglon_t](#) s)
Se ejecuta la operación "r |= s".
- void [renglonAnd](#) ([renglon_t](#) r, [renglon_t](#) s)
Se ejecuta la operación "r &= s".
- void [renglonNot](#) ([renglon_t](#) r)

- Se invierte el contenido del Renglon (se obtiene el complemento)*
- void `copiarRenglon` (`renglon_t` r1, const `renglon_t` r2)
 - copia el contenido de r2 en r1*
- void `copiarMatrizRenglon` (`matriz_t` m, const `renglon_t` r, int pos)
 - copia un Renglon sobre una Matriz a partir de una fila especificada*
- bool `renglonIzquierdoLibre` (`mensaje_t` *msj)
 - indica si la parte izquierda del renglón de un mensaje está vacía*
- void `charARenglon` (char c, `renglon_t` r)
 - Convierte un caracter a Renglon, ubicándolo en el extremo izquierdo del Renglon.*
- void `charAMatriz` (char c, `matriz_t` m, const int coord[])
 - Escribe un caracter en una matriz, a partir de las posiciones (x,y) (Extremo superior izdo)*
- void `uintARenglon` (`uint16_t` n, `renglon_t` r)
 - Convierte un entero no signado a Renglon, ubicándolo en el extremo izquierdo del renglon.*
- void `reemplazarLetra` (`renglon_t` r, char c, int j)
 - (re)escribe sobre el Renglon un caracter dado a partir de la columna j*
- `mensaje_t` `mensaje` (char *msj, int pos, bool repetir)
 - constructor de la variable mensaje_t*
- void `moverMensaje` (`mensaje_t` *msj)
 - desplaza el contenido del Renglon doble hacia la izquierda*
- void `concatenarLetraMensaje` (char c, `mensaje_t` *msj)
 - agrega una letra al string del mensaje y también al renglon*
- void `reemplazarUltLetraMensaje` (char c, `mensaje_t` *msj)
 - reemplaza la última letra en el string del mensaje y también del renglon*

4.44.1 Detailed Description

Encabezado de mensajes, con definiciones sobre tipos de datos y funciones.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.44.2 Function Documentation

4.44.2.1 borrarRenglon()

```
void borrarRenglon (
    renglon_t r )
```

elimina el contenido del Renglon

Parameters

<i>r</i>	Renglon a limpiar
----------	-------------------

4.44.2.2 charAMatriz()

```
void charAMatriz (
    char c,
    matriz_t m,
    const int coord[] )
```

Escribe un caracter en una matriz, a partir de las posiciones (x,y) (Extremo superior izdo)

Parameters

<i>c</i>	caracter
<i>m</i>	Matriz
<i>x</i>	
<i>y</i>	

4.44.2.3 charARenglon()

```
void charARenglon (
    char c,
    renglon_t r )
```

Convierte un caracter a Renglon, ubicándolo en el extremo izquierdo del Renglon.

Parameters

<i>c</i>	caracter
<i>r</i>	Renglon

4.44.2.4 concatenarLetraMensaje()

```
void concatenarLetraMensaje (
    char c,
    mensaje_t * msj )
```

agrega una letra al string del mensaje y también al renglon

Parameters

<i>c</i>	
<i>msj</i>	

4.44.2.5 copiarMatrizRenglon()

```
void copiarMatrizRenglon (
    matriz_t m,
    const renglon_t r,
    int pos )
```

copia un Renglon sobre una Matriz a partir de una fila especificada

Parameters

<i>m</i>	Matriz destino
<i>r</i>	Renglon origen
<i>pos</i>	Fila de inicio

4.44.2.6 copiarRenglon()

```
void copiarRenglon (
    renglon_t r1,
    const renglon_t r2 )
```

copia el contenido de r2 en r1

Parameters

<i>r1</i>	Renglon destino
<i>r2</i>	Renglon origen

4.44.2.7 mensaje()

```
mensaje_t mensaje (
    char * msj,
    int pos,
    bool repetir )
```

constructor de la variable mensaje_t

Parameters

<i>msj</i>	String que se desea convertir a mensaje
<i>pos</i>	fila sobre la que se deberá mostrar en display
<i>repetir</i>	si se repetirá una vez terminado de mostrar

Returns

mensaje_t

4.44.2.8 moverMensaje()

```
void moverMensaje (
    mensaje_t * msj )
```

desplaza el contenido del Renglon doble hacia la izquierda

Parameters

<i>msj</i>	puntero a la variable mensaje_t
------------	------------------------------------

4.44.2.9 printRenglon()

```
void printRenglon (
    renglon_t r )
```

imprime dos renglon en consola (para debuggear)

Parameters

<i>r</i>	renglon a imprimir
----------	--------------------

4.44.2.10 reemplazarLetra()

```
void reemplazarLetra (
    renglon_t r,
    char c,
    int j )
```

(re)escribe sobre el Renglon un caracter dado a partir de la columna j

Parameters

<i>r</i>	Renglon
<i>c</i>	caracter
<i>j</i>	columna sobre la que se quiere escribir

4.44.2.11 reemplazarUltLetraMensaje()

```
void reemplazarUltLetraMensaje (
    char c,
    mensaje_t * msj )
```

reemplaza la última letra en el string del mensaje y también del renglon

Parameters

<i>c</i>	
<i>msj</i>	

4.44.2.12 renglonAnd()

```
void renglonAnd (
    renglon_t r,
    renglon_t s )
```

Se ejecuta la operación "*r* &= *s*".

Parameters

<i>r</i>	primer operando AND
<i>s</i>	segundo operando AND

4.44.2.13 renglonIzquierdoLibre()

```
bool renglonIzquierdoLibre (
    mensaje_t * msj )
```

indica si la parte izquierda del renglón de un mensaje está vacía

Parameters

<i>r</i>	Renglon a chequear
----------	--------------------

Returns

true
false

4.44.2.14 renglonNot()

```
void renglonNot (  
    renglon_t r )
```

Se invierte el contenido del Renglon (se obtiene el complemento)

Parameters

<i>r</i>	Renglon a invertir
----------	--------------------

4.44.2.15 renglonOr()

```
void renglonOr (  
    renglon_t r,  
    renglon_t s )
```

Se ejecuta la operación " $r \mid= s$ ".

Parameters

<i>r</i>	primer operando OR
<i>s</i>	segundo operando OR

4.44.2.16 renglonShiftDer()

```
void renglonShiftDer (  
    renglon_t r,  
    uint16_t s )
```

Desplaza a la derecha el contenido de un Renglon.

Parameters

<i>r</i>	Renglon a desplazar
<i>s</i>	cantidad de espacios

4.44.2.17 renglonShiftIzq()

```
void renglonShiftIzq (
    renglon_t r,
    uint16_t s )
```

Desplaza a la izquierda el contenido de un Renglon.

Parameters

<i>r</i>	Renglon a desplazar
<i>s</i>	cantidad de espacios

4.44.2.18 uintARenglon()

```
void uintARenglon (
    uint16_t n,
    renglon_t r )
```

Convierte un entero no signado a Renglon, ubicándolo en el extremo izquierdo del renglon.

Parameters

<i>n</i>	entero no signado de 16 bits
<i>r</i>	Renglon

4.45 mensajes.h

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

```
1
2
3 #ifndef _MENSAJES_H_
4 #define _MENSAJES_H_
5
6 /*****
7  * INCLUDE HEADER FILES
8  *****/
9
10 #include "bitmap.h"
11
12 #include <stdint.h>
13 #include <stdbool.h>
14
15 /*****
16  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
17  *****/
18
19 #define TAM_REGLON 5
20
21 #define POS_MSJ1 2
22 #define POS_MSJ2 9
23 #define POS_MSJ3 5
24 #define L_MAX 64
```

```

30
31 /*****
32  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
33  *****/
34 typedef union
35 {
36     uint32_t completo;
37     struct
38     {
39         uint16_t mitad_der;
40         uint16_t mitad_izq;
41     };
42 } renglon_t[5];
43
44 typedef struct Mensaje
45 {
46     char msj[L_MAX];
47     int posicion;
48     int index;
49     int longitud;
50     int j;
51     bool habilitacion;
52     bool mover_texto;
53     bool repetir_msj;
54     renglon_t renglon;
55 } mensaje_t;
56
57 /*****
58  * VARIABLE PROTOTYPES WITH GLOBAL SCOPE
59  *****/
60
61 // +ej: extern unsigned int anio_actual;+
62
63 /*****
64  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
65  *****/
66
67 void printRenglon(renglon_t r);
68
69 void borrarRenglon(renglon_t r);
70
71 void renglonShiftDer(renglon_t r, uint16_t s);
72
73 void renglonShiftIzq(renglon_t r, uint16_t s);
74
75 void renglonOr(renglon_t r, renglon_t s);
76
77 void renglonAnd(renglon_t r, renglon_t s);
78
79 void renglonNot(renglon_t r);
80
81 void copiarRenglon(renglon_t r1, const renglon_t r2);
82
83 void copiarMatrizRenglon(matriz_t m, const renglon_t r, int pos);
84
85 bool renglonIzquierdoLibre(mensaje_t *msj);
86
87 void charARenglon(char c, renglon_t r);
88
89 void charAMatriz(char c, matriz_t m, const int coord[]);
90
91 void uintARenglon(uint16_t n, renglon_t r); // copia un número a renglon hasta que se acabe el número o
92     el renglon (lo 1 q ocurra)
93
94 void reemplazarLetra(renglon_t r, char c, int j);
95
96 mensaje_t mensaje(char *msj, int pos, bool repetir);
97
98 void moverMensaje(mensaje_t *msj);
99
100 void concatenarLetraMensaje(char c, mensaje_t *msj);
101
102 void reemplazarUltLetraMensaje(char c, mensaje_t *msj);
103
104 /*****
105  *****/
106
107 #endif // _MENSAJES_H_

```

4.46 audio.h

```

1 /*
2  * Simple-SDL2-Audio

```

```

3  *
4  * Copyright 2016 Jake Besworth
5  *
6  * Licensed under the Apache License, Version 2.0 (the "License");
7  * you may not use this file except in compliance with the License.
8  * You may obtain a copy of the License at
9  *
10 *     http://www.apache.org/licenses/LICENSE-2.0
11 *
12 * Unless required by applicable law or agreed to in writing, software
13 * distributed under the License is distributed on an "AS IS" BASIS,
14 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 * See the License for the specific language governing permissions and
16 * limitations under the License.
17 *
18 */
19
20 /*
21 * audio.h
22 *
23 * All audio related functions go here
24 *
25 */
26 #ifndef SIMPLE_AUDIO_
27 #define SIMPLE_AUDIO_
28
29 #ifdef __cplusplus
30 extern "C"
31 {
32 #endif
33
34 #include <SDL2/SDL.h>
35
36 /*
37 * Queue structure for all loaded sounds
38 *
39 */
40 typedef struct sound
41 {
42     uint32_t length;
43     uint32_t lengthTrue;
44     uint8_t * bufferTrue;
45     uint8_t * buffer;
46     uint8_t loop;
47     uint8_t fade;
48     uint8_t free;
49     uint8_t volume;
50
51     SDL_AudioSpec audio;
52
53     struct sound * next;
54 } Audio;
55
56 /*
57 * Create a Audio object
58 *
59 * @param filename      Filename for the WAVE file to load
60 * @param loop          0 ends after playing once (sound), 1 repeats and fades when other music added
61 *                      (music)
62 * @param volume        Volume, read playSound()
63 *
64 * @return returns a new Audio or NULL on failure, you must call freeAudio() on return Audio
65 */
66 Audio * createAudio(const char * filename, uint8_t loop, int volume);
67
68 /*
69 * Frees as many chained Audios as given
70 *
71 * @param audio        Chain of sounds to free
72 *
73 */
74 void freeAudio(Audio * audio);
75
76 /*
77 * Play a wave file currently must be S16LE format 2 channel stereo
78 *
79 * @param filename      Filename to open, use getAbsolutePath
80 * @param volume        Volume 0 - 128. SDL_MIX_MAXVOLUME constant for max volume
81 *
82 */
83 void playSound(const char * filename, int volume);
84
85 /*
86 * Plays a new music, only 1 at a time plays
87 *
88 * @param filename      Filename of the WAVE file to load

```



```

89  * @param volume      Volume read playSound for moree
90  *
91  */
92 void playMusic(const char * filename, int volume);
93
94 /*
95  * Plays a sound from a createAudio object (clones), only 1 at a time plays
96  * Advantage to this method is no more disk reads, only once, data is stored and constantly reused
97  *
98  * @param audio      Audio object to clone and use
99  * @param volume      Volume read playSound for moree
100  *
101  */
102 void playSoundFromMemory(Audio * audio, int volume);
103
104 /*
105  * Plays a music from a createAudio object (clones), only 1 at a time plays
106  * Advantage to this method is no more disk reads, only once, data is stored and constantly reused
107  *
108  * @param audio      Audio object to clone and use
109  * @param volume      Volume read playSound for moree
110  *
111  */
112 void playMusicFromMemory(Audio * audio, int volume);
113
114 /*
115  * Free all audio related variables
116  * Note, this needs to be run even if initAudio fails, because it frees the global audio device
117  *
118  */
119 void endAudio(void);
120
121 /*
122  * Initialize Audio Variable
123  *
124  */
125 void initAudio(void);
126
127 /*
128  * Pause audio from playing
129  *
130  */
131 void pauseAudio(void);
132
133 /*
134  * Unpause audio from playing
135  *
136  */
137 void unpauseAudio(void);
138
139 #ifdef __cplusplus
140 }
141 #endif
142
143 #endif

```

4.47 src/queue.c File Reference

Source del modulo queue. Funciones para el manejo de la cola de eventos.

```

#include "queue.h"
#include <stdlib.h>
#include <stdio.h>

```

Data Structures

- struct [nodeT](#)

Typedefs

- typedef struct [nodeT](#) **node_t**

Functions

- void **queueInsertar** (event_t nuevo)
Agrega un evento a la cola.
- bool **queueVacía** (void)
Chequea si la cola está vacía.
- event_t **queueSiguienteEvento** (void)
Devuelve el siguiente evento de la cola.
- void **destruirQueue** (void)
Destruye la cola de eventos.

4.47.1 Detailed Description

Source del modulo queue. Funciones para el manejo de la cola de eventos.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.47.2 Function Documentation

4.47.2.1 queueSiguienteEvento()

```
event_t queueSiguienteEvento (  
    void )
```

Devuelve el siguiente evento de la cola.

Returns

event_t

4.47.2.2 queueVacía()

```
bool queueVacía (  
    void )
```

Chequea si la cola está vacía.

Returns

true Vacía
false No vacía

4.48 src/queue.h File Reference

Header del modulo queue. Prototipos de funciones de interaccion con la cola de eventos.

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

Typedefs

- typedef int **event_t**

Enumerations

- enum **eventos** { **NADA** = -1 , **SALIR** = 0 , **GAME_OVER** , **FORCE_SALIR** }
- enum **eventos_tecla** {
 NO_MOVER = -1 , **ESC** = 59 , **BORRAR** = 63 , **ENTER** = 67 ,
 IZDA = 82 , **DCHA** , **ARRIBA** , **ABAJO** }

Functions

- void **queueInsertar** (event_t)
 Agrega un evento a la cola.
- bool **queueVacía** (void)
 Chequea si la cola está vacía.
- event_t **queueSiguienteEvento** (void)
 Devuelve el siguiente evento de la cola.
- void **destruirQueue** (void)
 Destruye la cola de eventos.

4.48.1 Detailed Description

Header del modulo queue. Prototipos de funciones de interaccion con la cola de eventos.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.48.2 Function Documentation

4.48.2.1 queueSiguienteEvento()

```
event_t queueSiguienteEvento (
    void )
```

Devuelve el siguiente evento de la cola.

Returns

event_t

4.48.2.2 queueVacía()

```
bool queueVacía (
    void )
```

Chequea si la cola está vacía.

Returns

true Vacía

false No vacía

4.49 queue.h

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

```
1
12 // https://stackoverflow.com/questions/3536153/c-dynamically-growing-array
13
14 #ifndef _QUEUE_H_
15 #define _QUEUE_H_
16
17 /*****
18  * INCLUDE HEADER FILES
19  *****/
20
21 #include <stdbool.h>
22
23 /*****
24  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
25  *****/
26
27 // Tipo de dato para eventos
28 typedef int event_t;
29
30 // Eventos posibles principales
31 enum eventos
32 {
33     NADA = -1,
34     SALIR = 0,
35     GAME_OVER,
36     FORCE_SALIR
37 };
38
39 // Eventos posibles de interacción en el juego
40 enum eventos_tecla
41 {
42     NO_MOVER = -1,
43     ESC = 59,
44     BORRAR = 63,
45     ENTER = 67,
46     IZDA = 82,
47     DCHA,
```

```

48  ARRIBA,
49  ABAJO
50 };
51
52 /*****
53  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
54  *****/
55
60 void queueInsertar(event_t);
61
68 bool queueVacía(void);
69
75 event_t queueSiguienteEvento(void);
76
81 void destruirQueue(void);
82
83 /*****
84  *****/
85
86 #endif // _QUEUE_H_

```

4.50 src/ranking.c File Reference

Source del modulo ranking. Funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente.

```
#include "ranking.h"
```

Macros

- `#define MAX_LEN 100`

Functions

- void **iniciarRanking** (void)
Inicializa el sistema de ranking.
- void **actualizarRanking** (char *name, unsigned long long score)
Actualiza el ranking de un jugador dado.
- void **desiniciarRanking** (void)
Desinicializa el sistema de ranking, actualizando el archivo correspondiente.
- bool **verificarJugadorRanking** (char *name)
Verifica si el jugador existe en el ranking.
- unsigned long long **getJugadorRankingPuntos** (char *name)
Devuelve el puntaje de un jugador dado.
- uint **getRankingLineas** (void)
Devuelve cantidad de renglones del ranking.
- char ** **getRankingNombres** (void)
Devuelve array de nombres de jugadores.
- unsigned long long * **getRankingPuntos** (void)
Devuelve array de puntos de jugadores.

Variables

- FILE * **handlerRanking** = NULL
- FILE * **handlerTemp** = NULL

4.50.1 Detailed Description

Source del modulo ranking. Funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente,.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.50.2 Function Documentation

4.50.2.1 actualizarRanking()

```
void actualizarRanking (
    char * name,
    unsigned long long score )
```

Actualiza el ranking de un jugador dado.

Parameters

<i>name</i>	Nombre del jugador
<i>score</i>	Puntos del jugador

4.50.2.2 getJugadorRankingPuntos()

```
unsigned long long getJugadorRankingPuntos (
    char * name )
```

Devuelve el puntaje de un jugador dado.

Parameters

<i>name</i>	Nombre del jugador
-------------	--------------------

Returns

unsigned long long Score

4.50.2.3 getRankingLineas()

```
uint getRankingLineas (
    void )
```

Devuelve cantidad de renglones del ranking.

Returns

int Renglones

4.50.2.4 getRankingNombres()

```
char ** getRankingNombres (
    void )
```

Devuelve array de nombres de jugadores.

Returns

char**

4.50.2.5 getRankingPuntos()

```
unsigned long long * getRankingPuntos (
    void )
```

Devuelve array de puntos de jugadores.

Returns

unsigned long long*

4.50.2.6 verificarJugadorRanking()

```
bool verificarJugadorRanking (
    char * name )
```

Verifica si el jugador existe en el ranking.

Parameters

<i>name</i>	Nombre del jugador
-------------	--------------------

Returns

true Existe
false No existe

4.51 src/ranking.h File Reference

Header del modulo ranking. Prototipos de funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente,.

```
#include <stdio.h>
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include <stdlib.h>
```

Macros

- `#define DEFAULT_PLAYER_NAME "PLAYER"`

Functions

- void **iniciarRanking** (void)
Inicializa el sistema de ranking.
- void **actualizarRanking** (char *name, unsigned long long score)
Actualiza el ranking de un jugador dado.
- void **desiniciarRanking** (void)
Desinicializa el sistema de ranking, actualizando el archivo correspondiente.
- bool **verificarJugadorRanking** (char *name)
Verifica si el jugador existe en el ranking.
- unsigned long long **getJugadorRankingPuntos** (char *name)
Devuelve el puntaje de un jugador dado.
- uint **getRankingLineas** (void)
Devuelve cantidad de renglones del ranking.
- char ** **getRankingNombres** (void)
Devuelve array de nombres de jugadores.
- unsigned long long * **getRankingPuntos** (void)
Devuelve array de puntos de jugadores.

4.51.1 Detailed Description

Header del modulo ranking. Prototipos de funciones de interaccion con el ranking de jugadores. Permite trabajar con el txt correspondiente fácilmente,.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.51.2 Function Documentation

4.51.2.1 actualizarRanking()

```
void actualizarRanking (
    char * name,
    unsigned long long score )
```

Actualiza el ranking de un jugador dado.

Parameters

<i>name</i>	Nombre del jugador
<i>score</i>	Puntos del jugador

4.51.2.2 getJugadorRankingPuntos()

```
unsigned long long getJugadorRankingPuntos (
    char * name )
```

Devuelve el puntaje de un jugador dado.

Parameters

<i>name</i>	Nombre del jugador
-------------	--------------------

Returns

unsigned long long Score

4.51.2.3 getRankingLineas()

```
uint getRankingLineas (
    void )
```

Devuelve cantidad de renglones del ranking.

Returns

int Renglones

4.51.2.4 getRankingNombres()

```
char ** getRankingNombres (
    void )
```

Devuelve array de nombres de jugadores.

Returns

char**

4.51.2.5 getRankingPuntos()

```
unsigned long long * getRankingPuntos (
    void )
```

Devuelve array de puntos de jugadores.

Returns

unsigned long long*

4.51.2.6 verificarJugadorRanking()

```
bool verificarJugadorRanking (
    char * name )
```

Verifica si el jugador existe en el ranking.

Parameters

<i>name</i>	Nombre del jugador
-------------	--------------------

Returns

true Existe
false No existe

4.52 ranking.h

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

```
1
13 #ifndef _RANKING_H_
14 #define _RANKING_H_
```

```

15
16 /*****
17  * INCLUDE HEADER FILES
18  *****/
19
20 #include <stdio.h>
21 #include <stdbool.h>
22 #include <stdint.h>
23 #include <string.h>
24 #include <stdlib.h>
25
26 /*****
27  * CONSTANT AND MACRO DEFINITIONS USING #DEFINE
28  *****/
29
30 #define DEFAULT_PLAYER_NAME "PLAYER"
31
32 /*****
33  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
34  *****/
35
40 void iniciarRanking(void);
41
48 void actualizarRanking(char *name, unsigned long long score);
49
54 void desiniciarRanking(void);
55
63 bool verificarJugadorRanking(char *name);
64
71 unsigned long long getJugadorRankingPuntos(char *name);
72
78 uint getRankingLineas(void);
79
85 char **getRankingNombres(void);
86
92 unsigned long long *getRankingPuntos(void);
93
94 /*****
95  *****/
96
97 #endif // _RANKING_H_

```

4.53 src/sound.h File Reference

Header del modulo sound Vinculo entre la fsm y las plataformas en lo que respecta al sonido.

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

Enumerations

- enum **musica** {
MUSICA_MENU_PPAL , **MUSICA_RANKING** , **MUSICA_CREDITOS** , **MUSICA_JUGANDO** ,
MUSICA_MENU_PAUSA , **MUSICA_GAME_OVER** , **SIZEOF_MUSICA** }
- enum **efectos** {
EFFECTO_SELECCION , **EFFECTO_SALTO** , **EFFECTO_IMPACTO** , **EFFECTO_AHOGADO** ,
EFFECTO_POCO_TIEMPO , **EFFECTO_META** , **EFFECTO_NIVEL_COMPLETO** , **EFFECTO_NUEVO_MAX_SCORE** ,
EFFECTO_MENU_ENTER , **EFFECTO_SALIENDO** , **EFFECTO_SIN_TIEMPO** , **SIZEOF_EFECTOS** }

Functions

- bool **iniciarSonido** (void)
Inicializa el sonido de la plataforma.
- void **destruirSonido** (void)
Desinicializa el sonido de la plataforma.

- void **pausarMusica** (void)
Pausa la musica actual.
- void **reproducirMusica** (int musica)
Pone a reproducir una musica dada.
- void **reproducirEfecto** (int efecto)
Pone a reproducir un efecto dado.

4.53.1 Detailed Description

Header del modulo sound Vinculo entre la fsm y las plataformas en lo que respecta al sonido.

Authors

AGRIPPINO, ALVAREZ, CASTRO, HEIR

Copyright

Copyright (c) 2022 ~ Ingeniería Electrónica ~ ITBA

4.53.2 Function Documentation

4.53.2.1 iniciarSonido()

```
bool iniciarSonido (  
    void )
```

Inicializa el sonido de la plataforma.

Returns

true Exit
false Error

4.53.2.2 reproducirEfecto()

```
void reproducirEfecto (  
    int efecto )
```

Pone a reproducir un efecto dado.

Parameters

<i>int</i>	num efectos
------------	-------------

4.53.2.3 reproducirMusica()

```
void reproducirMusica (
    int musica )
```

Pone a reproducir una musica dada.

Parameters

<i>int</i>	enum musica
------------	-------------

4.54 sound.h

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

```
1
13 #ifndef _SOUND_H_
14 #define _SOUND_H_
15
16 /*****
17  * INCLUDE HEADER FILES
18  *****/
19
20 #include <stdbool.h>
21
22 /*****
23  * ENUMERATIONS AND STRUCTURES AND TYPEDEFS
24  *****/
25
26 // Musicas a usar
27 enum musica
28 {
29     MUSICA_MENU_PPAL,
30     MUSICA_RANKING,
31     MUSICA_CREDITOS,
32     MUSICA_JUGANDO,
33     MUSICA_MENU_PAUSA,
34     MUSICA_GAME_OVER,
35     SIZEOF_MUSICA
36 };
37
38 // Efectos a usar
39 enum efectos
40 {
41     EFECTO_SELECCION,
42     EFECTO_SALTO,
43     EFECTO_IMPACTO,
44     EFECTO_AHOGADO,
45     EFECTO_POCO_TIEMPO,
46     EFECTO_META,
47     EFECTO_NIVEL_COMPLETO,
48     EFECTO_NUEVO_MAX_SCORE,
49     EFECTO_MENU_ENTER,
50     EFECTO_SALIENDO,
51     EFECTO_SIN_TIEMPO,
52     SIZEOF_EFECTOS
53 };
54
55 /*****
56  * FUNCTION PROTOTYPES WITH GLOBAL SCOPE
57  *****/
58
59 bool iniciarSonido(void);
60
61 void destruirSonido(void);
62
63 void pausarMusica(void);
64
65 void reproducirMusica(int musica);
66
```

```
91 void reproducirEfecto(int efecto);
92
93 /*****
94  *****/
95
96 #endif // _SOUND_H_
```

Index

- actualizarRanking
 - ranking.c, [146](#)
 - ranking.h, [149](#)
- ALGIF_ANIMATION, [5](#)
- ALGIF_BITMAP, [5](#)
- ALGIF_FRAME, [6](#)
- ALGIF_PALETTE, [6](#)
- ALGIF_RGB, [6](#)
- allegro_draw_hitbox
 - allegro_stuff.c, [42](#)
 - allegro_stuff.h, [50](#)
- allegro_draw_menu_background
 - allegro_stuff.c, [42](#)
 - allegro_stuff.h, [51](#)
- allegro_get_last_key
 - allegro_stuff.c, [43](#)
 - allegro_stuff.h, [51](#)
- allegro_get_next_event
 - allegro_stuff.c, [43](#)
 - allegro_stuff.h, [51](#)
- allegro_get_var_done
 - allegro_stuff.c, [43](#)
 - allegro_stuff.h, [51](#)
- allegro_get_var_event
 - allegro_stuff.c, [43](#)
 - allegro_stuff.h, [52](#)
- allegro_get_var_font
 - allegro_stuff.c, [44](#)
 - allegro_stuff.h, [52](#)
- allegro_get_var_font_h
 - allegro_stuff.c, [44](#)
 - allegro_stuff.h, [52](#)
- allegro_get_var_font_w
 - allegro_stuff.c, [44](#)
 - allegro_stuff.h, [52](#)
- allegro_get_var_redraw
 - allegro_stuff.c, [44](#)
 - allegro_stuff.h, [53](#)
- allegro_is_event_queueVacía
 - allegro_stuff.c, [45](#)
 - allegro_stuff.h, [53](#)
- allegro_set_last_key
 - allegro_stuff.c, [45](#)
 - allegro_stuff.h, [53](#)
- allegro_set_rick_flag
 - allegro_stuff.c, [45](#)
 - allegro_stuff.h, [54](#)
- allegro_set_var_done
 - allegro_stuff.c, [46](#)
 - allegro_stuff.h, [54](#)
- allegro_set_var_event
 - allegro_stuff.c, [46](#)
 - allegro_stuff.h, [54](#)
- allegro_set_var_redraw
 - allegro_stuff.c, [46](#)
 - allegro_stuff.h, [54](#)
- allegro_wait_for_event
 - allegro_stuff.c, [46](#)
 - allegro_stuff.h, [54](#)
- must_init, [47](#)
- allegro_stuff.h
 - allegro_draw_hitbox, [50](#)
 - allegro_draw_menu_background, [51](#)
 - allegro_get_last_key, [51](#)
 - allegro_get_next_event, [51](#)
 - allegro_get_var_done, [51](#)
 - allegro_get_var_event, [52](#)
 - allegro_get_var_font, [52](#)
 - allegro_get_var_font_h, [52](#)
 - allegro_get_var_font_w, [52](#)
 - allegro_get_var_redraw, [53](#)
 - allegro_is_event_queueVacía, [53](#)
 - allegro_set_last_key, [53](#)
 - allegro_set_rick_flag, [54](#)
 - allegro_set_var_done, [54](#)
 - allegro_set_var_event, [54](#)
 - allegro_set_var_redraw, [54](#)
 - allegro_wait_for_event, [55](#)
 - must_init, [55](#)
- allegro_t, [7](#)
- allegro_wait_for_event
 - allegro_stuff.c, [46](#)
 - allegro_stuff.h, [55](#)

- bitmap.c
 - copiarMatriz, [36](#)
 - limpiarMatriz, [36](#)
 - matrizAnd, [37](#)
 - matrizNot, [37](#)
 - matrizOr, [37](#)
 - matrizXor, [38](#)
 - printMatriz, [38](#)
- bitmap.h
 - copiarMatriz, [119](#)
 - limpiarMatriz, [120](#)
 - matrizAnd, [120](#)
 - matrizNot, [120](#)
 - matrizOr, [120](#)
 - matrizXor, [121](#)
 - printMatriz, [121](#)
- borrarRenglon
 - mensajes.c, [125](#)
 - mensajes.h, [132](#)
- car_t, [7](#)
- cargarRanking
 - display.c, [59](#)
 - display.h, [16](#)
- charAMatriz
 - mensajes.c, [125](#)
 - mensajes.h, [133](#)
- charARenglon
 - mensajes.c, [125](#)
 - mensajes.h, [133](#)
- coin_t, [7](#)
- collide
 - geometry.c, [86](#)
 - geometry.h, [97](#)
- collideShort
 - geometry.c, [87](#)
 - geometry.h, [97](#)
- concatenarLetraMensaje
 - mensajes.c, [126](#)
 - mensajes.h, [133](#)
- copiarMatriz
 - bitmap.c, [36](#)
 - bitmap.h, [119](#)
- copiarMatrizRenglon
 - mensajes.c, [126](#)
 - mensajes.h, [134](#)
- copiarRenglon
 - mensajes.c, [126](#)
 - mensajes.h, [134](#)
- data_t, [8](#)
- dcoord_t, [8](#)
- dejarTexto
 - display.c, [60](#)
 - display.h, [16](#)
- devolverNombre
 - nombre.c, [114](#), [115](#)
 - nombre.h, [32](#)
- display.c
 - cargarRanking, [59](#)
 - dejarTexto, [60](#)
 - iniciarDisplay, [60](#)
 - mostrarCreditos, [60](#)
 - mostrarTexto, [60](#)
- display.h
 - cargarRanking, [16](#)
 - dejarTexto, [16](#)
 - iniciarDisplay, [16](#)
 - mostrarCreditos, [17](#)
 - mostrarTexto, [17](#)
- en_game_over
 - fsm.c, [20](#)
- en_game_over_esperando_opcion
 - fsm.c, [21](#)
- en_menu_ppal
 - fsm.c, [21](#)
- en_pausa
 - fsm.c, [21](#)
- en_pausa_esperando_opcion
 - fsm.c, [21](#)
- entities.h
 - entities_move_frog, [61](#)
- entities_move_frog
 - entities.h, [61](#)
- fixHighCpuUsage
 - fsm.c, [20](#)
 - fsm.h, [24](#)
- frog_t, [8](#)
- fsm
 - fsm.c, [20](#)
 - fsm.h, [24](#)
- fsm.c
 - en_game_over, [20](#)
 - en_game_over_esperando_opcion, [21](#)
 - en_menu_ppal, [21](#)
 - en_pausa, [21](#)
 - en_pausa_esperando_opcion, [21](#)
 - fixHighCpuUsage, [20](#)
 - fsm, [20](#)
 - inicializarFsm, [20](#)
 - jugando, [22](#)
 - menu_ppal_esperando_opcion, [22](#)
 - poniendo_nombre, [22](#)
 - seleccionando_dificultad, [22](#)
 - viendo_creditos, [23](#)
 - viendo_ranking, [23](#)
- fsm.h
 - fixHighCpuUsage, [24](#)
 - fsm, [24](#)
 - inicializarFsm, [24](#)
- game.c
 - getMaxPuntos, [63](#), [67](#)
 - getNivel, [64](#), [67](#)
 - getNombre, [64](#), [68](#)
 - getPuntos, [64](#), [68](#)

- setDificultad, [64](#), [68](#)
- setMaxPuntos, [65](#), [69](#)
- setNombre, [65](#), [69](#)
- tiempoRefrescoEntidades, [65](#), [69](#)
- game.h
 - getMaxPuntos, [26](#)
 - getNivel, [27](#)
 - getNombre, [27](#)
 - getPuntos, [27](#)
 - setDificultad, [27](#)
 - setMaxPuntos, [28](#)
 - setNombre, [28](#)
 - tiempoRefrescoEntidades, [28](#)
- game_data.c
 - game_data_add_name_letter, [71](#)
 - game_data_are_goals_full, [72](#)
 - game_data_get_diff, [72](#)
 - game_data_get_frames, [72](#)
 - game_data_get_game_over_flag, [72](#)
 - game_data_get_goal_state, [73](#)
 - game_data_get_lives, [73](#)
 - game_data_get_name, [73](#)
 - game_data_get_old_max_score, [74](#)
 - game_data_get_run_number, [74](#)
 - game_data_get_run_time_left, [74](#)
 - game_data_get_score, [74](#)
 - game_data_get_score_max, [75](#)
 - game_data_get_time_left_flag, [75](#)
 - game_data_get_timer_in_sec, [75](#)
 - game_data_overwrite_name, [75](#)
 - game_data_set_diff, [76](#)
 - game_data_set_goal, [76](#)
 - game_data_set_score_max, [76](#)
- game_data.h
 - game_data_add_name_letter, [78](#)
 - game_data_are_goals_full, [79](#)
 - game_data_get_diff, [79](#)
 - game_data_get_frames, [79](#)
 - game_data_get_game_over_flag, [79](#)
 - game_data_get_goal_state, [80](#)
 - game_data_get_lives, [80](#)
 - game_data_get_name, [80](#)
 - game_data_get_old_max_score, [81](#)
 - game_data_get_run_number, [81](#)
 - game_data_get_run_time_left, [81](#)
 - game_data_get_score, [81](#)
 - game_data_get_score_max, [82](#)
 - game_data_get_time_left_flag, [82](#)
 - game_data_get_timer_in_sec, [82](#)
 - game_data_overwrite_name, [82](#)
 - game_data_set_diff, [83](#)
 - game_data_set_goal, [83](#)
 - game_data_set_score_max, [83](#)
- game_data_add_name_letter
 - game_data.c, [71](#)
 - game_data.h, [78](#)
- game_data_are_goals_full
 - game_data.c, [72](#)
- game_data.h
 - game_data.h, [79](#)
- game_data_get_diff
 - game_data.c, [72](#)
 - game_data.h, [79](#)
- game_data_get_frames
 - game_data.c, [72](#)
 - game_data.h, [79](#)
- game_data_get_game_over_flag
 - game_data.c, [72](#)
 - game_data.h, [79](#)
- game_data_get_goal_state
 - game_data.c, [73](#)
 - game_data.h, [80](#)
- game_data_get_lives
 - game_data.c, [73](#)
 - game_data.h, [80](#)
- game_data_get_name
 - game_data.c, [73](#)
 - game_data.h, [80](#)
- game_data_get_old_max_score
 - game_data.c, [74](#)
 - game_data.h, [81](#)
- game_data_get_run_number
 - game_data.c, [74](#)
 - game_data.h, [81](#)
- game_data_get_run_time_left
 - game_data.c, [74](#)
 - game_data.h, [81](#)
- game_data_get_score
 - game_data.c, [74](#)
 - game_data.h, [81](#)
- game_data_get_score_max
 - game_data.c, [75](#)
 - game_data.h, [82](#)
- game_data_get_time_left_flag
 - game_data.c, [75](#)
 - game_data.h, [82](#)
- game_data_get_timer_in_sec
 - game_data.c, [75](#)
 - game_data.h, [82](#)
- game_data_overwrite_name
 - game_data.c, [75](#)
 - game_data.h, [82](#)
- game_data_set_diff
 - game_data.c, [76](#)
 - game_data.h, [83](#)
- game_data_set_goal
 - game_data.c, [76](#)
 - game_data.h, [83](#)
- game_data_set_score_max
 - game_data.c, [76](#)
 - game_data.h, [83](#)
- geometry.c
 - collide, [86](#)
 - collideShort, [87](#)
 - get_rand_between, [87](#)
 - getXYFromCarFrame, [88](#)
 - getXYFromCoinFrame, [88](#)

- getXYFromFrogFrame, 88
- getXYFromSplashFrame, 89
- getXYFromTurtleFrame, 89
- goal_cols, 92
- inside, 89
- insideShort, 90
- insideShortScaled, 91
- lanes_cars, 92
- lanes_logs, 93
- lanes_turtles, 93
- mapInt, 91
- matchUInt, 92
- geometry.h
 - collide, 97
 - collideShort, 97
 - get_rand_between, 98
 - getXYFromCarFrame, 98
 - getXYFromCoinFrame, 99
 - getXYFromFrogFrame, 99
 - getXYFromSplashFrame, 99
 - getXYFromTurtleFrame, 100
 - goal_cols, 103
 - inside, 100
 - insideShort, 101
 - insideShortScaled, 101
 - lanes_cars, 103
 - lanes_logs, 103
 - lanes_turtles, 103
 - mapInt, 102
 - matchUInt, 102
- get_rand_between
 - geometry.c, 87
 - geometry.h, 98
- getJugadorRankingPuntos
 - ranking.c, 146
 - ranking.h, 149
- getMaxPuntos
 - game.c, 63, 67
 - game.h, 26
- getNivel
 - game.c, 64, 67
 - game.h, 27
- getNombre
 - game.c, 64, 68
 - game.h, 27
- getOpcion
 - menu.c, 109, 112
- getPuntos
 - game.c, 64, 68
 - game.h, 27
- getRankingLineas
 - ranking.c, 146
 - ranking.h, 149
- getRankingNombres
 - ranking.c, 147
 - ranking.h, 149
- getRankingPuntos
 - ranking.c, 147
- ranking.h, 150
- getXYFromCarFrame
 - geometry.c, 88
 - geometry.h, 98
- getXYFromCoinFrame
 - geometry.c, 88
 - geometry.h, 99
- getXYFromFrogFrame
 - geometry.c, 88
 - geometry.h, 99
- getXYFromSplashFrame
 - geometry.c, 89
 - geometry.h, 99
- getXYFromTurtleFrame
 - geometry.c, 89
 - geometry.h, 100
- goal_cols
 - geometry.c, 92
 - geometry.h, 103
- inicializarFsm
 - fsm.c, 20
 - fsm.h, 24
- iniciarDisplay
 - display.c, 60
 - display.h, 16
- iniciarSonido
 - sound.c, 116, 118
 - sound.h, 152
- input.c
 - leerEntradas, 107, 108
- input.h
 - leerEntradas, 30
- inside
 - geometry.c, 89
 - geometry.h, 100
- insideShort
 - geometry.c, 90
 - geometry.h, 101
- insideShortScaled
 - geometry.c, 91
 - geometry.h, 101
- jcoord_t, 9
- jugando
 - fsm.c, 22
- lanes_cars
 - geometry.c, 92
 - geometry.h, 103
- lanes_logs
 - geometry.c, 93
 - geometry.h, 103
- lanes_turtles
 - geometry.c, 93
 - geometry.h, 103
- leerEntradas
 - input.c, 107, 108
 - input.h, 30

- limpiarMatriz
 - bitmap.c, 36
 - bitmap.h, 120
- log_t, 9
- mapInt
 - geometry.c, 91
 - geometry.h, 102
- matchUInt
 - geometry.c, 92
 - geometry.h, 102
- matrizAnd
 - bitmap.c, 37
 - bitmap.h, 120
- matrizNot
 - bitmap.c, 37
 - bitmap.h, 120
- matrizOr
 - bitmap.c, 37
 - bitmap.h, 120
- matrizXor
 - bitmap.c, 38
 - bitmap.h, 121
- Mensaje, 9
- mensaje
 - mensajes.c, 127
 - mensajes.h, 134
- mensajes.c
 - borrarRenglon, 125
 - charAMatriz, 125
 - charARenglon, 125
 - concatenarLetraMensaje, 126
 - copiarMatrizRenglon, 126
 - copiarRenglon, 126
 - mensaje, 127
 - moverMensaje, 127
 - printRenglon, 127
 - reemplazarLetra, 128
 - reemplazarUltLetraMensaje, 128
 - renglonAnd, 128
 - renglonIzquierdoLibre, 129
 - renglonNot, 129
 - renglonOr, 129
 - renglonShiftDer, 130
 - renglonShiftIzq, 130
 - uintARenglon, 130
- mensajes.h
 - borrarRenglon, 132
 - charAMatriz, 133
 - charARenglon, 133
 - concatenarLetraMensaje, 133
 - copiarMatrizRenglon, 134
 - copiarRenglon, 134
 - mensaje, 134
 - moverMensaje, 135
 - printRenglon, 135
 - reemplazarLetra, 135
 - reemplazarUltLetraMensaje, 136
 - renglonAnd, 136
 - renglonIzquierdoLibre, 136
 - renglonNot, 137
 - renglonOr, 137
 - renglonShiftDer, 137
 - renglonShiftIzq, 138
 - uintARenglon, 138
- menu.c
 - getOpcion, 109, 112
 - setMenu, 109, 112
 - setOpcion, 111, 112
- menu_ppal_esperando_opcion
 - fsm.c, 22
- menu_t, 10
- mostrarCreditos
 - display.c, 60
 - display.h, 17
- mostrarTexto
 - display.c, 60
 - display.h, 17
- moverMensaje
 - mensajes.c, 127
 - mensajes.h, 135
- must_init
 - allegro_stuff.c, 47
 - allegro_stuff.h, 55
- nodeT, 10
- nombre.c
 - devolverNombre, 114, 115
- nombre.h
 - devolverNombre, 32
- pair_xy_t, 10
- poniendo_nombre
 - fsm.c, 22
- printMatriz
 - bitmap.c, 38
 - bitmap.h, 121
- printRenglon
 - mensajes.c, 127
 - mensajes.h, 135
- privateAudioDevice, 10
- queue.c
 - queueSiguienteEvento, 142
 - queueVacia, 142
- queue.h
 - queueSiguienteEvento, 143
 - queueVacia, 144
- queueSiguienteEvento
 - queue.c, 142
 - queue.h, 143
- queueVacia
 - queue.c, 142
 - queue.h, 144
- ranking.c
 - actualizarRanking, 146
 - getJugadorRankingPuntos, 146

- getRankingLineas, [146](#)
 - getRankingNombres, [147](#)
 - getRankingPuntos, [147](#)
 - verificarJugadorRanking, [147](#)
- ranking.h
 - actualizarRanking, [149](#)
 - getJugadorRankingPuntos, [149](#)
 - getRankingLineas, [149](#)
 - getRankingNombres, [149](#)
 - getRankingPuntos, [150](#)
 - verificarJugadorRanking, [150](#)
- reemplazarLetra
 - mensajes.c, [128](#)
 - mensajes.h, [135](#)
- reemplazarUltLetraMensaje
 - mensajes.c, [128](#)
 - mensajes.h, [136](#)
- renglon_t, [11](#)
- renglonAnd
 - mensajes.c, [128](#)
 - mensajes.h, [136](#)
- renglonIzquierdoLibre
 - mensajes.c, [129](#)
 - mensajes.h, [136](#)
- renglonNot
 - mensajes.c, [129](#)
 - mensajes.h, [137](#)
- renglonOr
 - mensajes.c, [129](#)
 - mensajes.h, [137](#)
- renglonShiftDer
 - mensajes.c, [130](#)
 - mensajes.h, [137](#)
- renglonShiftIzq
 - mensajes.c, [130](#)
 - mensajes.h, [138](#)
- reproducirEfecto
 - sound.c, [116](#), [118](#)
 - sound.h, [152](#)
- reproducirMusica
 - sound.c, [117](#), [118](#)
 - sound.h, [153](#)
- seleccionando_dificultad
 - fsm.c, [22](#)
- setDificultad
 - game.c, [64](#), [68](#)
 - game.h, [27](#)
- setMaxPuntos
 - game.c, [65](#), [69](#)
 - game.h, [28](#)
- setMenu
 - menu.c, [109](#), [112](#)
- setNombre
 - game.c, [65](#), [69](#)
 - game.h, [28](#)
- setOpcion
 - menu.c, [111](#), [112](#)
- sound, [11](#)
- sound.c
 - iniciarSonido, [116](#), [118](#)
 - reproducirEfecto, [116](#), [118](#)
 - reproducirMusica, [117](#), [118](#)
- sound.h
 - iniciarSonido, [152](#)
 - reproducirEfecto, [152](#)
 - reproducirMusica, [153](#)
- sounds_t, [11](#)
- sprites_menu_t, [12](#)
- sprites_t, [12](#)
- src/display.h, [15](#), [17](#)
- src/fsm.c, [18](#)
- src/fsm.h, [23](#), [25](#)
- src/game.h, [25](#), [29](#)
- src/input.h, [29](#), [30](#)
- src/main.c, [31](#)
- src/menu.h, [31](#)
- src/nombre.h, [32](#), [33](#)
- src/platform/pc/algif5/algif.h, [33](#), [34](#)
- src/platform/pc/allegro_stuff.c, [38](#)
- src/platform/pc/allegro_stuff.h, [47](#), [55](#)
- src/platform/pc/display.c, [58](#)
- src/platform/pc/entities.h, [61](#), [62](#)
- src/platform/pc/game.c, [62](#)
- src/platform/pc/game_data.c, [69](#)
- src/platform/pc/game_data.h, [76](#), [83](#)
- src/platform/pc/geometry.c, [85](#)
- src/platform/pc/geometry.h, [93](#), [104](#)
- src/platform/pc/input.c, [106](#)
- src/platform/pc/menu.c, [108](#)
- src/platform/pc/nombre.c, [113](#)
- src/platform/pc/sound.c, [115](#)
- src/platform/rpi/bitmap.c, [35](#)
- src/platform/rpi/bitmap.h, [118](#), [121](#)
- src/platform/rpi/disdrv.h, [122](#)
- src/platform/rpi/game.c, [66](#)
- src/platform/rpi/input.c, [107](#)
- src/platform/rpi/joydrv.h, [123](#)
- src/platform/rpi/mensajes.c, [123](#)
- src/platform/rpi/mensajes.h, [131](#), [138](#)
- src/platform/rpi/menu.c, [111](#)
- src/platform/rpi/nombre.c, [114](#)
- src/platform/rpi/simpleSDL2audio/audio.h, [139](#)
- src/platform/rpi/sound.c, [117](#)
- src/queue.c, [141](#)
- src/queue.h, [143](#), [144](#)
- src/ranking.c, [145](#)
- src/ranking.h, [148](#), [150](#)
- src/sound.h, [151](#), [153](#)
- state_diagram_edge, [13](#)
- tiempoRefrescoEntidades
 - game.c, [65](#), [69](#)
 - game.h, [28](#)
- turtle_pack_t, [14](#)
- uintARenglon
 - mensajes.c, [130](#)

mensajes.h, [138](#)

verificarJugadorRanking
 ranking.c, [147](#)
 ranking.h, [150](#)

viendo_creditos
 fsm.c, [23](#)

viendo_ranking
 fsm.c, [23](#)

window_t, [14](#)