

## Zappy SERVER

Generated by Doxygen 1.9.6



<b>1 Class Index</b>	<b>1</b>
1.1 Class List	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 s_app Struct Reference	5
3.2 s_client Struct Reference	5
3.3 s_game Struct Reference	5
3.4 s_gui Struct Reference	6
3.5 s_ia Struct Reference	6
3.6 s_incantation_info Struct Reference	6
3.7 s_inventory Struct Reference	7
3.8 s_list Struct Reference	7
3.9 s_list_node Struct Reference	7
3.10 s_node_data Union Reference	8
3.11 s_parsing Struct Reference	8
3.11.1 Detailed Description	8
3.12 s_server Struct Reference	9
3.13 s_team Struct Reference	9
3.14 s_time_info Struct Reference	9
3.15 s_vector2i Struct Reference	10
3.16 tile_s Struct Reference	10
3.16.1 Detailed Description	10
<b>4 File Documentation</b>	<b>11</b>
4.1 ai.h	11
4.2 command_ai.h	12
4.3 look.h	12
4.4 stuck.h	13
4.5 team.h	13
4.6 app.h	14
4.7 game.h	14
4.8 gui.h	15
4.9 list.h	15
4.10 type.h	15
4.11 map.h	16
4.12 parsing.h	16
4.13 rules.h	17
4.14 client.h	17
4.15 server.h	18
4.16 types.h	19



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">s_app</a>	5
<a href="#">s_client</a>	5
<a href="#">s_game</a>	5
<a href="#">s_gui</a>	6
<a href="#">s_ia</a>	6
<a href="#">s_incantation_info</a>	6
<a href="#">s_inventory</a>	7
<a href="#">s_list</a>	7
<a href="#">s_list_node</a>	7
<a href="#">s_node_data</a>	8
<a href="#">s_parsing</a>	
Struct for parsing the arguments	8
<a href="#">s_server</a>	9
<a href="#">s_team</a>	9
<a href="#">s_time_info</a>	9
<a href="#">s_vector2i</a>	10
<a href="#">tile_s</a>	
Struct for a map's tile	10



## Chapter 2

# File Index

### 2.1 File List

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

/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ <a href="#">parsing.h</a>	16
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ <a href="#">rules.h</a>	17
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ <a href="#">types.h</a>	19
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ <a href="#">ai.h</a>	11
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ <a href="#">stuck.h</a>	13
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ <a href="#">team.h</a>	13
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/cmd/ <a href="#">command_ai.h</a>	12
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/cmd/ <a href="#">look.h</a>	12
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/app/ <a href="#">app.h</a>	14
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/game/ <a href="#">game.h</a>	14
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/gui/ <a href="#">gui.h</a>	15
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/ <a href="#">list.h</a>	15
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/ <a href="#">type.h</a>	15
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/map/ <a href="#">map.h</a>	16
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/ <a href="#">client.h</a>	17
/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/ <a href="#">server.h</a>	18





## Chapter 3

# Class Documentation

### 3.1 s\_app Struct Reference

Collaboration diagram for s\_app:

### 3.2 s\_client Struct Reference

#### Public Attributes

- size\_t **fd**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/client.h

### 3.3 s\_game Struct Reference

Collaboration diagram for s\_game:

#### Public Attributes

- size\_t **height**
- size\_t **width**
- [tile\\_t](#) \*\* **map**
- struct timeval **start**
- struct timeval **start\_food**
- int **freq**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/game/game.h

### 3.4 s\_gui Struct Reference

Collaboration diagram for s\_gui:

#### Public Attributes

- size\_t **fd**
- [list\\_t](#) \* **list\_messages**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/gui/gui.h

### 3.5 s\_ia Struct Reference

Collaboration diagram for s\_ia:

#### Public Attributes

- size\_t **fd**
- [list\\_t](#) \* **list\_command**
- [list\\_t](#) \* **list\_messages**
- [vector2i\\_t](#) \* **position**
- orientation\_t **direction**
- [inventory\\_t](#) \* **inventory**
- [incantation\\_info\\_t](#) \* **incantation**
- size\_t **level**
- [time\\_info\\_t](#) \* **time**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ai.h

### 3.6 s\_incantation\_info Struct Reference

#### Public Attributes

- bool **status\_incantation**
- size\_t **target\_level**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ai.h

## 3.7 s\_inventory Struct Reference

### Public Attributes

- `size_t food`
- `size_t linemate`
- `size_t deraumere`
- `size_t sibur`
- `size_t mendiane`
- `size_t phiras`
- `size_t thystame`

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

- `/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ai.h`

## 3.8 s\_list Struct Reference

Collaboration diagram for s\_list:

### Public Attributes

- `list_node_t * first`
- `list_node_t * last`
- `size_t len`

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

- `/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/type.h`

## 3.9 s\_list\_node Struct Reference

Collaboration diagram for s\_list\_node:

### Public Attributes

- `node_data_t data`
- `struct s_list_node * next`
- `struct s_list_node * prev`

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

- `/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/type.h`

## 3.10 s\_node\_data Union Reference

Collaboration diagram for s\_node\_data:

### Public Attributes

- [ia\\_t](#) \* **ai**
- [gui\\_t](#) \* **gui**
- [client\\_t](#) \* **client**
- [team\\_t](#) \* **team**
- char \* **message**
- char \* **command**
- [vector2i\\_t](#) \* **coord**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/type.h

## 3.11 s\_parsing Struct Reference

Struct for parsing the arguments.

```
#include <types.h>
```

### Public Attributes

- int **port**
- int **width**
- int **height**
- int **clientsNb**
- int **freq**
- char \*\* **names**

### 3.11.1 Detailed Description

Struct for parsing the arguments.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/types.h

## 3.12 s\_server Struct Reference

### Public Attributes

- fd\_set **read\_fds**
- fd\_set **write\_fds**
- int **fd**
- socklen\_t **addrlen**
- struct sockaddr\_in **addr**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/server.h

## 3.13 s\_team Struct Reference

Collaboration diagram for s\_team:

### Public Attributes

- [list\\_t](#) \* **list\_ai**
- [list\\_t](#) \* **egg\_position**
- char \* **name**
- size\_t **max\_place**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/team.h

## 3.14 s\_time\_info Struct Reference

### Public Attributes

- bool **stuck**
- struct timeval **start\_stuck**
- double **total\_stuck**
- struct timeval **start\_life**
- double **total\_life**

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ai.h

## 3.15 s\_vector2i Struct Reference

### Public Attributes

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

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/type.h

## 3.16 tile\_s Struct Reference

Struct for a map's tile.

```
#include <map.h>
```

### Public Attributes

- size\_t **food**
- size\_t **linemate**
- size\_t **deraumere**
- size\_t **sibur**
- size\_t **mendiane**
- size\_t **phiras**
- size\_t **thystame**

### 3.16.1 Detailed Description

Struct for a map's tile.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/map/map.h

## Chapter 4

# File Documentation

### 4.1 ai.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** AI
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdlib.h>
00012 #include <stdbool.h>
00013
00014 #include "list/list.h"
00015
00016 typedef struct s_app app_t;
00017
00018 typedef enum {
00019     NORTH,
00020     SOUTH,
00021     EAST,
00022     WEST,
00023 } orientation_t;
00024
00025 typedef struct s_inventory {
00026     size_t food;
00027     size_t linemate;
00028     size_t deraumere;
00029     size_t sibur;
00030     size_t mendiane;
00031     size_t phiras;
00032     size_t thystame;
00033 } inventory_t;
00034
00035 typedef struct s_time_info {
00036     bool stuck;
00037     struct timeval start_stuck;
00038     double total_stuck;
00039     struct timeval start_life;
00040     double total_life;
00041 } time_info_t;
00042
00043 typedef struct s_incantation_info {
00044     bool status_incantation;
00045     size_t target_level;
00046 } incantation_info_t;
00047
00048 typedef struct s_ia {
00049     size_t fd;
00050     list_t *list_command;
00051     list_t *list_messages;
00052     vector2i_t *position;
00053     orientation_t direction;
00054     inventory_t *inventory;
00055     incantation_info_t *incantation;
00056     size_t level;
00057     time_info_t *time;
00058 } ia_t;
```

```
00059
00068 ia_t *create_ia(app_t *app, int fd, team_t *team);
00069
00077 void add_ia(app_t *app, size_t fd, char *line);
00078
00086 ia_t *find_ia(app_t *app, size_t fd);
00087
00093 void check_die(app_t *app);
```

## 4.2 command\_ai.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** AI Command
00006 */
00007
00008 #pragma once
00009
00010 #include "ai/ai.h"
00011
00012 #define BEGIN_INCANTATION 1
00013 #define END_INCANTATION 2
00014
00022 void command_ai_handler(app_t *app, ia_t *ai, char *line);
00023
00033 bool move_command(app_t *app, ia_t *ai, char *line);
00034
00044 bool object_info_command(app_t *app, ia_t *ai, char *line);
00045
00055 bool other_command(app_t *app, ia_t *ai, char *line);
00056
00063 void eject_cmd(app_t *app, ia_t *ai);
00064
00071 void eject_player(app_t *app, ia_t *ai);
00072
00079 void eject_egg(app_t *app, ia_t *ai);
00080
00086 void dead_response(app_t *app);
00087
00094 void fork_cmd(app_t *app, ia_t *ai);
00095
00102 void connect_nbr_cmd(app_t *app, ia_t *ai);
00103
00110 void take_cmd(app_t *app, ia_t *ai);
00111
00118 void set_cmd(app_t *app, ia_t *ai);
00119
00129 bool broadcast_command(app_t *app, ia_t *ai, char *line);
00130
00140 bool incantation_command(app_t *app, ia_t *ai, char *line);
00141
00152 bool check_incantation(app_t *app, ia_t *ai, int status);
00153
00161 void update_status(app_t *app, ia_t *ai, int update_status);
00162
00169 void level_up(app_t *app, ia_t *ai);
00170
00177 void add_command_to_list(ia_t *ai, char *line);
00178
00184 void treat_command(app_t *app);
00185
00191 void destroy_command_list(list_t *command_list);
```

## 4.3 look.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** AI Command
00006 */
00007
00008 #pragma once
00009
00010 #include "ai/ai.h"
00011
00012 #define PLAYER_STRING " player"
```



```

00013 #define EGG_STRING " egg"
00014 #define FOOD_STRING " food"
00015 #define LINEMATE_STRING " linemate"
00016 #define DERAUMERE_STRING " deraumere"
00017 #define SIBUR_STRING " sibur"
00018 #define MENDIANE_STRING " mendiane"
00019 #define PHIRAS_STRING " phiras"
00020 #define THYSTAME_STRING " thystame"
00021
00028 void look_cmd(app_t *app, ia_t *ai);
00029
00038 void check_player(vector2i_t *pos, app_t *app, char **reply, ia_t *ai);
00039
00047 void check_egg(vector2i_t *pos, app_t *app, char **reply);
00048
00056 void check_resources(vector2i_t *pos, app_t *app, char **reply);
00057
00066 void look_north(app_t *app, ia_t *ai, int index_line, char **reply);
00067
00076 void look_east(app_t *app, ia_t *ai, int index_line, char **reply);
00077
00086 void look_south(app_t *app, ia_t *ai, int index_line, char **reply);
00087
00096 void look_west(app_t *app, ia_t *ai, int index_line, char **reply);

```

## 4.4 stuck.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Game struct
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdlib.h>
00012 #include <stdbool.h>
00013 #include <sys/time.h>
00014
00015 #include "list/list.h"
00016
00017 typedef struct s_app app_t;
00018
00025 double time_elapsed(struct timeval *time);
00026
00033 void set_time_stuck(ia_t *ai, double total_stuck);
00034
00040 void treat_stuck(app_t *app);

```

## 4.5 team.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Team struct
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdlib.h>
00012 #include <stdbool.h>
00013
00014 #include "list/list.h"
00015
00016 typedef struct s_app app_t;
00017
00018 typedef struct s_team {
00019     list_t *list_ai;
00020     list_t *egg_position;
00021     char *name;
00022     size_t max_place;
00023 } team_t;
00024
00033 team_t *create_team(app_t *app, char *name, size_t max_place);
00034

```

```
00042 void add_team(app_t *app, char *team_name, size_t max_place);
00043
00049 void display_egg_position(app_t *app);
00050
00058 void add_egg(list_t *eggs, int random_x, int random_y);
00059
00067 team_t *find_team(app_t *app, size_t fd);
00068
00074 void destroy_team(list_t *teams_list);
```

## 4.6 app.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** App
00006 */
00007
00008 #pragma once
00009
00010 #include <time.h>
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include <stdarg.h>
00014 #include <string.h>
00015 #include <signal.h>
00016
00017 #include "ai/ai.h"
00018 #include "ai/team.h"
00019 #include "gui/gui.h"
00020 #include "parsing.h"
00021 #include "ai/stuck.h"
00022 #include "list/list.h"
00023 #include "game/game.h"
00024 #include "server/server.h"
00025
00026 typedef struct s_app {
00027     list_t *gui_list;
00028     list_t *teams_list;
00029     list_t *clients_list;
00030     server_t *server;
00031     game_t *game;
00032 } app_t;
00033
00040 app_t *create_app(parsing_t *parsing);
00041
00047 void destroy_app(app_t *app);
```

## 4.7 game.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Game struct
00006 */
00007
00008 #pragma once
00009
00010 #include "map/map.h"
00011
00012 typedef struct s_game {
00013     size_t height;
00014     size_t width;
00015     tile_t **map;
00016     struct timeval start;
00017     struct timeval start_food;
00018     int freq;
00019 } game_t;
00020
00029 game_t *create_game(int height, int width, int freq);
00030
00036 void spawn_ressources(app_t *app);
00037
00045 bool check_win(app_t *app);
00046
00052 void destroy_game(game_t *game);
```

## 4.8 gui.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Gui struct
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdbool.h>
00012
00013 #include "list/list.h"
00014
00015 typedef struct s_gui {
00016     size_t fd;
00017     list_t *list_messages;
00018 } gui_t;
00019
00026 gui_t *create_gui(int fd);
00027
00035 void add_gui(app_t *app, size_t fd, char *line);
00036
00044 gui_t *find_gui(app_t *app, size_t fd);
00045
00051 void destroy_gui(list_t *gui_list);

```

## 4.9 list.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Function about list
00006 */
00007
00008 #pragma once
00009
00010 #include <stdbool.h>
00011
00012 #include "type.h"
00013
00019 list_t *list_new(void);
00020
00026 void list_free(list_t *list);
00027
00034 void list_delete(list_t *list, list_node_t *node);
00035
00041 void list_remove_front(list_t *list);
00042
00048 void list_remove_back(list_t *list);
00049
00057 bool list_add_back(list_t *list, node_data_t data);
00058
00066 bool list_add_front(list_t *list, node_data_t data);

```

## 4.10 type.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Type of List
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011
00012 typedef struct s_ia ia_t;
00013 typedef struct s_gui gui_t;
00014 typedef struct s_team team_t;
00015 typedef struct s_client client_t;
00016
00017 typedef struct s_vector2i {
00018     int x;
00019     int y;

```

```

00020 } vector2i_t;
00021
00022 typedef union s_node_data {
00023     ia_t *ai;
00024     gui_t *gui;
00025     client_t *client;
00026     team_t *team;
00027     char *message;
00028     char *command;
00029     vector2i_t *coord;
00030 } node_data_t;
00031
00032 typedef struct s_list_node {
00033     node_data_t data;
00034     struct s_list_node *next;
00035     struct s_list_node *prev;
00036 } list_node_t;
00037
00038 typedef struct s_list {
00039     list_node_t *first;
00040     list_node_t *last;
00041     size_t len;
00042 } list_t;

```

## 4.11 map.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** map
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdbool.h>
00012
00013 #define FOOD_DENSITY 0.5
00014 #define LINEMATE_DENSITY 0.3
00015 #define DERAUMERE_DENSITY 0.15
00016 #define SIBUR_DENSITY 0.1
00017 #define MENDIANE_DENSITY 0.1
00018 #define PHIRAS_DENSITY 0.08
00019 #define THYSTAME_DENSITY 0.05
00020
00024 enum entity_type_e {
00025     EGG,
00026     FOOD,
00027     LINEMATE,
00028     DERAUMERE,
00029     SIBUR,
00030     MENDIANE,
00031     PHIRAS,
00032     THYSTAME,
00033     NONE
00034 };
00035
00039 typedef struct tile_s {
00040     size_t food;
00041     size_t linemate;
00042     size_t deraumere;
00043     size_t sibur;
00044     size_t mendiane;
00045     size_t phiras;
00046     size_t thystame;
00047 } tile_t;
00048
00056 tile_t **create_map(int width, int height);
00057
00064 void free_map(tile_t **map, int height);
00065
00073 void display_map(tile_t **map, int height, int width);

```

## 4.12 parsing.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy

```

```

00004 ** File description:
00005 ** Header file for parsing the arguments passed to the server.
00006 */
00007
00008 #pragma once
00009
00010 #define HELP_FLAG_LABEL "--help"
00011 #define HELP_FLAG "-h"
00012 #define PORT_FLAG "-p"
00013 #define WIDTH_FLAG "-x"
00014 #define HEIGHT_FLAG "-y"
00015 #define TEAMS_NAMES_FLAG "-n"
00016 #define CLIENTS_FLAG "-c"
00017 #define FREQUENCY_FLAG "-f"
00018
00019 #define NB_ARGS_MIN 13
00020 #define NB_ARGS_HELP 2
00021
00022 #include "types.h"
00023
00024 #include <stdbool.h>
00025
00033 parsing_t *parse_arg(int ac, char **av);
00034
00044 int handle_help(int ac, char **av);
00045
00053 int parse_positive_int_arg(char *arg);
00054
00067 int parse_client(char **arg, int *pos, parsing_t *parsing);
00068
00081 int parse_frequency(char **arg, int *pos, parsing_t *parsing);
00082
00095 int parse_height(char **arg, int *pos, parsing_t *parsing);
00096
00109 int parse_port(char **arg, int *pos, parsing_t *parsing);
00110
00123 int parse_width(char **arg, int *pos, parsing_t *parsing);
00124
00137 int parse_names(char **arg, int *pos, parsing_t *parsing);
00138
00144 void destroy_parsing(parsing_t *parsing);

```

## 4.13 rules.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Rules for the server
00006 */
00007
00008 #pragma once
00009
00010 enum RETURN_CODES {
00011     CODE_ERROR_MISSING_ARG = -1,
00012     CODE_ERROR_INVALID_ARG = -2,
00013     CODE_ERROR_WRONG_FLAG = -3,
00014     CODE_ERROR_INVALID_NUMBER = -4,
00015     CODE_ERROR_MALLOC_FAILED = -5,
00016     CODE_HELP_SUCCESS = 1,
00017     CODE_SUCCESS = 0,
00018     CODE_FAILURE = 84
00019 };

```

## 4.14 client.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Client
00006 */
00007
00008 #pragma once
00009
00010 #include <stddef.h>
00011 #include <stdlib.h>
00012 #include <stdbool.h>
00013

```

```

00014 #include "app/app.h"
00015
00016 typedef struct s_client {
00017     size_t fd;
00018 } client_t;
00019
00026 client_t *create_client(int fd);
00027
00036 bool its_client(app_t *app, size_t fd);
00037
00045 list_node_t *find_client(list_t *clients_list, size_t fd);
00046
00052 void destroy_client(list_t *client_list);

```

## 4.15 server.h

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy Server
00004 ** File description:
00005 ** Server struct
00006 */
00007
00008 #pragma once
00009
00010 #include <stdio.h>
00011 #include <stddef.h>
00012 #include <stdlib.h>
00013 #include <unistd.h>
00014 #include <stdbool.h>
00015 #include <sys/select.h>
00016 #include <netinet/in.h>
00017
00018 #include "list/list.h"
00019
00020 #define LISTEN_NUMBER 100
00021 #define WELCOME_MESSAGE "WELCOME\n"
00022 #define WELCOME_MESSAGE_LEN 8
00023
00024 typedef struct s_app app_t;
00025
00026 typedef struct s_server {
00027     fd_set read_fds;
00028     fd_set write_fds;
00029     int fd;
00030     socklen_t addrlen;
00031     struct sockaddr_in addr;
00032 } server_t;
00033
00039 void destroy_server(server_t *server);
00040
00047 server_t *create_server(size_t port);
00048
00055 bool server_run(app_t *app);
00056
00064 bool server_connection_handler(app_t *app, size_t fd);
00065
00073 bool server_data_handler(app_t *app, size_t fd);
00074
00081 char *read_line(int fd);
00082
00089 void server_quit_handler(app_t *app, size_t fd);
00090
00099 bool write_message(list_t *list_messages, size_t fd);
00100
00107 void add_message(list_t *list, char *message);
00108
00116 char *format_string(const char *format, ...);
00117
00123 void server_reset_fd(app_t *app);
00124
00132 void handle_request(app_t *app, size_t fd, char *line);
00133
00141 char *append_char(char *line, char current_char);
00142
00149 void handle_client_read(app_t *app, int fd);
00150
00157 void handle_client_write(app_t *app, int fd);
00158
00166 vector2i_t *create_vector2i(int x, int y);
00167
00174 void concatenate_strings(char **str1, char *str2);
00175
00181 void destroy_message_list(list_t *message_list);

```

## 4.16 types.h

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Types for the server
00006 */
00007
00008 #pragma once
00009
00014 typedef struct s_parsing {
00015     int port;
00016     int width;
00017     int height;
00018     int clientsNb;
00019     int freq;
00020     char **names;
00021 } parsing_t;
```





# Index

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/ai.h](#),  
[11](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/cmd/command\\_ai.h](#),  
[12](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/cmd/look.h](#),  
[12](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/stuck.h](#),  
[13](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/ai/team.h](#),  
[13](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/app/app.h](#),  
[14](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/game/game.h](#),  
[14](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/gui/gui.h](#),  
[15](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/list.h](#),  
[15](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/list/type.h](#),  
[15](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/map/map.h](#),  
[16](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/parsing.h](#),  
[16](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/rules.h](#),  
[17](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/client.h](#),  
[17](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/server/server.h](#),  
[18](#)

[/home/tjerome-rocher/Desktop/Tek2/Zappy/server/include/types.h](#),  
[19](#)

  

[s\\_app](#), [5](#)

[s\\_client](#), [5](#)

[s\\_game](#), [5](#)

[s\\_gui](#), [6](#)

[s\\_ia](#), [6](#)

[s\\_incantation\\_info](#), [6](#)

[s\\_inventory](#), [7](#)

[s\\_list](#), [7](#)

[s\\_list\\_node](#), [7](#)

[s\\_node\\_data](#), [8](#)

[s\\_parsing](#), [8](#)

[s\\_server](#), [9](#)

[s\\_team](#), [9](#)

[s\\_time\\_info](#), [9](#)

[s\\_vector2i](#), [10](#)

  

[tile\\_s](#), [10](#)