

SSRoboime

Generated by Doxygen 1.9.8

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 Agent Namespace Reference	9
5.2 AgentPenalty Namespace Reference	9
5.3 BaseAgent Namespace Reference	9
5.4 Booting Namespace Reference	9
5.5 Printing Namespace Reference	9
5.6 RobotPositionManager Namespace Reference	10
5.6.1 Variable Documentation	10
5.6.1.1 root	10
5.7 run_full_team Namespace Reference	10
5.7.1 Variable Documentation	10
5.7.1.1 boot	10
5.7.1.2 p	10
5.7.1.3 players	10
5.8 run_player Namespace Reference	11
5.8.1 Variable Documentation	11
5.8.1.1 boot	11
5.8.1.2 p	11
5.9 ServerComm Namespace Reference	11
6 Class Documentation	13
6.1 Agent.Agent Class Reference	13
6.1.1 Detailed Description	14
6.1.2 Constructor & Destructor Documentation	14
6.1.2.1 __init__()	14
6.1.3 Member Data Documentation	15
6.1.3.1 unum	15
6.2 BaseAgent.BaseAgent Class Reference	15
6.2.1 Detailed Description	18
6.2.2 Constructor & Destructor Documentation	18
6.2.2.1 __init__()	18
6.2.3 Member Function Documentation	18

6.2.3.1 beam()	18
6.2.4 Member Data Documentation	18
6.2.4.1 AGENTS_IN_THE_MATCH	18
6.2.4.2 env	18
6.2.4.3 init_position	19
6.2.4.4 INITIAL_POSITION	19
6.2.4.5 logger	19
6.2.4.6 scom	19
6.2.4.7 unum	19
6.3 Booting.Booting Class Reference	19
6.3.1 Detailed Description	20
6.3.2 Constructor & Destructor Documentation	20
6.3.2.1 __init__()	20
6.3.3 Member Function Documentation	21
6.3.3.1 cpp_builder()	21
6.3.3.2 get_team_params()	21
6.3.3.3 show_spinner()	21
6.3.4 Member Data Documentation	21
6.3.4.1 CONFIG_PATH	21
6.3.4.2 options	22
6.4 Environment Class Reference	22
6.4.1 Detailed Description	24
6.4.2 Member Enumeration Documentation	24
6.4.2.1 PlayMode	24
6.4.2.2 PlayModeGroup	25
6.4.3 Constructor & Destructor Documentation	25
6.4.3.1 Environment()	25
6.4.4 Member Function Documentation	26
6.4.4.1 print_status()	26
6.4.4.2 update_from_server()	26
6.4.5 Member Data Documentation	26
6.4.5.1 goals_conceded	26
6.4.5.2 goals_scored	27
6.4.5.3 is_left	27
6.4.5.4 logger	27
6.4.5.5 time_match	27
6.4.5.6 time_server	27
6.4.5.7 unum	27
6.5 Logger Class Reference	28
6.5.1 Detailed Description	30
6.5.2 Constructor & Destructor Documentation	30
6.5.2.1 Logger() [1/2]	30

6.5.2.2 <code>Logger()</code> [2/2]	30
6.5.2.3 <code>~Logger()</code>	30
6.5.3 Member Function Documentation	30
6.5.3.1 <code>_init_file()</code>	30
6.5.3.2 <code>_worker_loop()</code>	31
6.5.3.3 <code>error()</code> [1/2]	31
6.5.3.4 <code>error()</code> [2/2]	31
6.5.3.5 <code>get()</code>	31
6.5.3.6 <code>info()</code> [1/2]	32
6.5.3.7 <code>info()</code> [2/2]	33
6.5.3.8 <code>log()</code>	33
6.5.3.9 <code>operator=()</code>	33
6.5.3.10 <code>warn()</code> [1/2]	34
6.5.3.11 <code>warn()</code> [2/2]	34
6.5.4 Member Data Documentation	34
6.5.4.1 <code>_current_buffer</code>	34
6.5.4.2 <code>_cv</code>	34
6.5.4.3 <code>_file_stream</code>	34
6.5.4.4 <code>_is_running</code>	35
6.5.4.5 <code>_mutex</code>	35
6.5.4.6 <code>_worker</code>	35
6.5.4.7 <code>_write_buffer</code>	35
6.5.4.8 <code>is_the_first</code>	35
6.6 Environment::Parsing Class Reference	36
6.6.1 Detailed Description	37
6.6.2 Constructor & Destructor Documentation	37
6.6.2.1 <code>Parsing()</code>	37
6.6.3 Member Function Documentation	38
6.6.3.1 <code>advance()</code>	38
6.6.3.2 <code>get_str()</code>	38
6.6.3.3 <code>get_value()</code>	38
6.6.3.4 <code>parse_gamestate()</code>	39
6.6.3.5 <code>parse_time()</code>	39
6.6.3.6 <code>skip_until_char()</code>	39
6.6.4 Member Data Documentation	40
6.6.4.1 <code>buffer</code>	40
6.6.4.2 <code>end</code>	40
6.6.4.3 <code>env</code>	40
6.7 Printing.Printing Class Reference	41
6.7.1 Detailed Description	42
6.7.2 Member Function Documentation	42
6.7.2.1 <code>get_input()</code>	42

6.7.2.2 print_message()	42
6.7.2.3 print_table()	43
6.7.3 Member Data Documentation	43
6.7.3.1 CONSOLE	43
6.7.3.2 IF_IN_DEBUG	43
6.7.3.3 TABLE_COLORS	44
6.8 RobotPositionManager.RobotPositionManager Class Reference	44
6.8.1 Detailed Description	48
6.8.2 Constructor & Destructor Documentation	48
6.8.2.1 __init__()	48
6.8.3 Member Function Documentation	48
6.8.3.1 _canvas_to_field()	48
6.8.3.2 _field_to_canvas()	49
6.8.3.3 apagar_config()	49
6.8.3.4 clear_grid()	49
6.8.3.5 click_on_grid()	49
6.8.3.6 criar_widgets()	50
6.8.3.7 destroy()	50
6.8.3.8 draw_player()	50
6.8.3.9 get_config_positions()	50
6.8.3.10 nova_config()	51
6.8.3.11 on_double_click_in_configs()	51
6.8.3.12 salvar_config()	51
6.8.3.13 save_config_positions()	51
6.8.3.14 update_table_config()	52
6.8.4 Member Data Documentation	52
6.8.4.1 canvas	52
6.8.4.2 canvas_height	52
6.8.4.3 canvas_width	52
6.8.4.4 click_on_grid	52
6.8.4.5 CONFIG_POSITION_PATH	52
6.8.4.6 config_positions	53
6.8.4.7 FIELD_HEIGHT	53
6.8.4.8 FIELD_WIDTH	53
6.8.4.9 GRID_SCALE	53
6.8.4.10 marcadores_jogadores	53
6.8.4.11 MAX_JOGADORES	53
6.8.4.12 nome_de_config_selecionada	53
6.8.4.13 on_double_click_in_configs	53
6.8.4.14 posicoes_atuais	54
6.8.4.15 tv_configs	54
6.8.4.16 X_MAX	54

6.8.4.17 X_MIN	54
6.8.4.18 Y_MAX	54
6.8.4.19 Y_MIN	54
6.9 ServerComm.ServerComm Class Reference	55
6.9.1 Detailed Description	56
6.9.2 Constructor & Destructor Documentation	56
6.9.2.1 __init__()	56
6.9.3 Member Function Documentation	57
6.9.3.1 __receive_async()	57
6.9.3.2 clear_queue()	57
6.9.3.3 close()	57
6.9.3.4 commit()	57
6.9.3.5 commit_beam()	58
6.9.3.6 receive()	58
6.9.3.7 send()	58
6.9.3.8 send_immediate()	58
6.9.4 Member Data Documentation	59
6.9.4.1 buffer	59
6.9.4.2 buffer_size	59
6.9.4.3 env	59
6.9.4.4 message_queue	59
6.9.4.5 socket	59
6.9.4.6 unum	59
7 File Documentation	61
7.1 src/agent/Agent.py File Reference	61
7.1.1 Detailed Description	61
7.2 Agent.py	61
7.3 src/agent/AgentPenalty.py File Reference	62
7.3.1 Detailed Description	62
7.4 AgentPenalty.py	62
7.5 src/agent/BaseAgent.py File Reference	62
7.5.1 Detailed Description	63
7.6 BaseAgent.py	63
7.7 src/communication/ServerComm.py File Reference	64
7.7.1 Detailed Description	64
7.8 ServerComm.py	64
7.9 src/cpp/environment/debug.cc File Reference	67
7.9.1 Function Documentation	68
7.9.1.1 main()	68
7.9.2 Variable Documentation	68
7.9.2.1 example	68

7.9.2.2 example1	69
7.9.2.3 size	69
7.9.2.4 size1	69
7.10 debug.cc	70
7.11 src/cpp/logger/debug.cc File Reference	70
7.11.1 Function Documentation	71
7.11.1.1 main()	71
7.11.1.2 tarefaPesada()	71
7.12 debug.cc	71
7.13 src/cpp/environment/Environment.hpp File Reference	73
7.13.1 Macro Definition Documentation	74
7.13.1.1 False	74
7.13.1.2 True	74
7.14 Environment.hpp	74
7.15 src/cpp/environment/module_main.cpp File Reference	77
7.15.1 Function Documentation	77
7.15.1.1 NB_MODULE()	77
7.16 module_main.cpp	78
7.17 src/cpp/logger/module_main.cpp File Reference	78
7.17.1 Function Documentation	79
7.17.1.1 NB_MODULE()	79
7.18 module_main.cpp	79
7.19 src/cpp/logger/Logger.hpp File Reference	80
7.19.1 Macro Definition Documentation	80
7.19.1.1 False	80
7.19.1.2 True	81
7.20 Logger.hpp	81
7.21 src/run_full_team.py File Reference	83
7.22 run_full_team.py	83
7.23 src/run_player.py File Reference	83
7.24 run_player.py	84
7.25 src/term/Booting.py File Reference	84
7.25.1 Detailed Description	84
7.26 Booting.py	84
7.27 src/term/Printing.py File Reference	87
7.27.1 Detailed Description	87
7.28 Printing.py	88
7.29 src/utls/RobotPositionManager.py File Reference	90
7.29.1 Detailed Description	90
7.30 RobotPositionManager.py	90

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Agent	9
AgentPenalty	9
BaseAgent	9
Booting	9
Printing	9
RobotPositionManager	10
run_full_team	10
run_player	11
ServerComm	11

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Booting.Booting	19
Environment	22
Logger	28
Environment::Parsing	36
Printing.Printing	41
ServerComm.ServerComm	55
tk.Tk	
RobotPositionManager.RobotPositionManager	44
ABC	
BaseAgent.BaseAgent	15
BaseAgent	
Agent.Agent	13

Chapter 3

Class Index

3.1 Class List

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

Agent.Agent	Classe que representará os agentes de campo, possuindo métodos correspondentes	13
BaseAgent.BaseAgent	Classe que agrupará todas as funcionalidades comuns a qualquer agente	15
Booting.Booting	Responsável por inicializar todas as necessidades de execução do time	19
Environment	Responsável por representar o ambiente externo ao robô	22
Logger	Singleton para logging assíncrono	28
Environment::Parsing	Responsável por prover ferramentas de auxílio de parsing	36
Printing.Printing	Responsável pela comunicação usuário - terminal	41
RobotPositionManager.RobotPositionManager	Responsável por permitir ao usuário a criação de diversas formações táticas	44
ServerComm.ServerComm	Responsável pela comunicação com servidor	55

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

src/run_full_team.py	83
src/run_player.py	83
src/agent/Agent.py	
Implementação de Lógica de Agente de Campo	61
src/agent/AgentPenalty.py	
Implementação de Lógica de Goleiro	62
src/agent/BaseAgent.py	
Implementação da classe de jogador base, que deve ser comum a todos os agentes	62
src/communication/ServerComm.py	
Implementação da Comunicação com Servidor	64
src/cpp/environment/debug.cc	67
src/cpp/environment/Environment.hpp	73
src/cpp/environment/module_main.cpp	77
src/cpp/logger/debug.cc	70
src/cpp/logger/Logger.hpp	80
src/cpp/logger/module_main.cpp	78
src/term/Booting.py	
Implementação do Booting do time	84
src/term/Printing.py	
Implementação de Interface no terminal	87
src/utis/RobotPositionManager.py	
Implementação de lógica organizadora de posições iniciais de partida	90

Chapter 5

Namespace Documentation

5.1 Agent Namespace Reference

Classes

- class [Agent](#)

Classe que representará os agentes de campo, possuindo métodos correspondentes.

5.2 AgentPenalty Namespace Reference

5.3 BaseAgent Namespace Reference

Classes

- class [BaseAgent](#)

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

5.4 Booting Namespace Reference

Classes

- class [Booting](#)

Responsável por inicializar todas as necessidades de execução do time.

5.5 Printing Namespace Reference

Classes

- class [Printing](#)

Responsável pela comunicação usuário - terminal.

5.6 RobotPositionManager Namespace Reference

Classes

- class [RobotPositionManager](#)

Responsável por permitir ao usuário a criação de diversas formações táticas.

Variables

- `root` = [RobotPositionManager\(\)](#)

5.6.1 Variable Documentation

5.6.1.1 `root`

```
RobotPositionManager.root = RobotPositionManager\(\)
```

Definition at line 397 of file [RobotPositionManager.py](#).

5.7 `run_full_team` Namespace Reference

Variables

- `boot` = [Booting\(\)](#)
- list `players` = []
- Agent `p`

5.7.1 Variable Documentation

5.7.1.1 `boot`

```
run_full_team.boot = Booting\(\)
```

Definition at line 5 of file [run_full_team.py](#).

5.7.1.2 `p`

```
Agent run_full_team.p
```

Definition at line 13 of file [run_full_team.py](#).

5.7.1.3 `players`

```
list run_full_team.players = []
```

Definition at line 7 of file [run_full_team.py](#).

5.8 run_player Namespace Reference

Variables

- `boot` = `Booting()`
- `p` = `Agent(boot.options)`

5.8.1 Variable Documentation

5.8.1.1 boot

```
run_player.boot = Booting()
```

Definition at line 4 of file [run_player.py](#).

5.8.1.2 p

```
run_player.p = Agent(boot.options)
```

Definition at line 6 of file [run_player.py](#).

5.9 ServerComm Namespace Reference

Classes

- class [ServerComm](#)
Responsável pela comunicação com servidor.

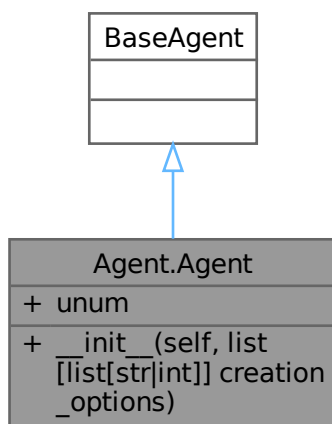
Chapter 6

Class Documentation

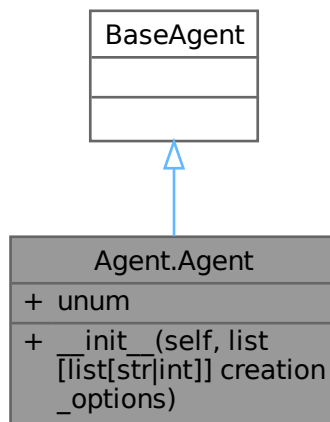
6.1 Agent.Agent Class Reference

Classe que representará os agentes de campo, possuindo métodos correspondentes.

Inheritance diagram for Agent.Agent:



Collaboration diagram for Agent.Agent:



Public Member Functions

- `__init__` (self, list[[list[str|int]]] creation_options)
Construtor da classe agente de campo, inicializando informações gerais.

Public Attributes

- `unum`

6.1.1 Detailed Description

Classe que representará os agentes de campo, possuindo métodos correspondentes.

Definition at line 7 of file [Agent.py](#).

6.1.2 Constructor & Destructor Documentation

6.1.2.1 `__init__()`

```

Agent.Agent.__init__ (
    self,
    list[[list[str | int]] creation_options )

```

Construtor da classe agente de campo, inicializando informações gerais.

Parameters

<code>creation_options</code>	Lista de Parâmetros de Criação de Agente
-------------------------------	--

Parâmetros presentes em `creation_options`:

- IP Server
- Porta de Agente
- Porta de Monitor
- Nome do time
- Número de Uniforme
- Tipo de Robô
- Tiro livre Penâlti
- Proxy
- Modo de Debug

Definition at line 12 of file [Agent.py](#).

6.1.3 Member Data Documentation

6.1.3.1 unum

`Agent.Agent.unum`

Definition at line 29 of file [Agent.py](#).

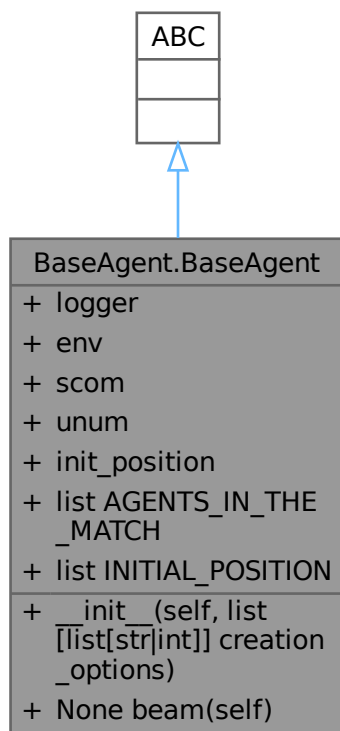
The documentation for this class was generated from the following file:

- [src/agent/Agent.py](#)

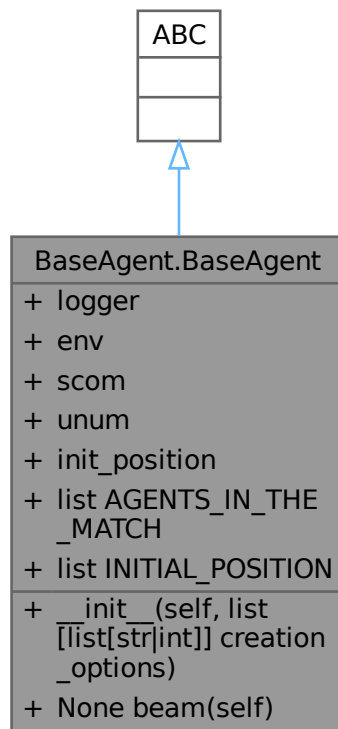
6.2 BaseAgent.BaseAgent Class Reference

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

Inheritance diagram for BaseAgent.BaseAgent:



Collaboration diagram for BaseAgent.BaseAgent:



Public Member Functions

- `__init__` (self, list[list[str|int]] creation_options)
Construtor da classe base de agente, chamando todos os construtores de outras classes mínimas para cada agente.
- `None beam` (self)
Responsável por gerenciar o teletransporte dos jogadores.

Public Attributes

- `logger`
- `env`
- `scom`
- `unum`
- `init_position`

Static Public Attributes

- list `AGENTS_IN_THE_MATCH` = []
- list `INITIAL_POSITION` = []

6.2.1 Detailed Description

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

Definition at line 10 of file [BaseAgent.py](#).

6.2.2 Constructor & Destructor Documentation

6.2.2.1 `__init__()`

```
BaseAgent.BaseAgent.__init__ (
    self,
    list[list[str | int]] creation_options )
```

Construtor da classe base de agente, chamando todos os construtores de outras classes mínimas para cada agente.

Parameters

<code>creation_options</code>	Lista de Parâmetros de Criação de Agente
-------------------------------	--

Definition at line 18 of file [BaseAgent.py](#).

6.2.3 Member Function Documentation

6.2.3.1 `beam()`

```
None BaseAgent.BaseAgent.beam (
    self )
```

Responsável por gerenciar o teletransporte dos jogadores.

Definition at line 55 of file [BaseAgent.py](#).

6.2.4 Member Data Documentation

6.2.4.1 `AGENTS_IN_THE_MATCH`

```
list BaseAgent.BaseAgent.AGENTS_IN_THE_MATCH = [] [static]
```

Definition at line 15 of file [BaseAgent.py](#).

6.2.4.2 `env`

```
BaseAgent.BaseAgent.env
```

Definition at line 30 of file [BaseAgent.py](#).

6.2.4.3 init_position

`BaseAgent.BaseAgent.init_position`

Definition at line 53 of file [BaseAgent.py](#).

6.2.4.4 INITIAL_POSITION

`list BaseAgent.BaseAgent.INITIAL_POSITION = [] [static]`

Definition at line 16 of file [BaseAgent.py](#).

6.2.4.5 logger

`BaseAgent.BaseAgent.logger`

Definition at line 29 of file [BaseAgent.py](#).

6.2.4.6 scom

`BaseAgent.BaseAgent.scom`

Definition at line 31 of file [BaseAgent.py](#).

6.2.4.7 unum

`BaseAgent.BaseAgent.unum`

Definition at line 40 of file [BaseAgent.py](#).

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

- [src/agent/BaseAgent.py](#)

6.3 Booting.Booting Class Reference

Responsável por inicializar todas as necessidades de execução do time.

Collaboration diagram for Booting.Booting:

Booting.Booting
+ options
+ str CONFIG_PATH
+ __init__(self)
+ list[list[str int]] get_team_params()
+ None show_spinner(list [bool] running_flag)
+ None cpp_builder()

Public Member Functions

- [__init__](#) (self)

Responsável por chamar as inicializações mínimas.

Static Public Member Functions

- list[list[str|int]] [get_team_params](#) ()

Verifica existência de arquivo de parâmetros de time, caso não exista, usará o default.

- None [show_spinner](#) (list[bool] running_flag)

Por motivos estéticos, mostrará um spinner enquanto há o carregamento de módulos C++.

- None [cpp_builder](#) ()

Responsável por buildar os arquivos .cpp presentes na pasta cpp.

Public Attributes

- [options](#)

Static Public Attributes

- str [CONFIG_PATH](#) = Path(__file__).resolve().parent / "config_team_params.txt"

6.3.1 Detailed Description

Responsável por inicializar todas as necessidades de execução do time.

Assume as seguintes responsabilidades:

- Estabelece um arquivo de configurações default caso já não exista um.

Definition at line 16 of file [Booting.py](#).

6.3.2 Constructor & Destructor Documentation

6.3.2.1 __init__()

```
Booting.Booting.__init__ (  
    self )
```

Responsável por chamar as inicializações mínimas.

Definition at line 26 of file [Booting.py](#).

6.3.3 Member Function Documentation

6.3.3.1 `cpp_builder()`

```
None Booting.Bootng.cpp_builder ( ) [static]
```

Responsável por buildar os arquivos .cpp presentes na pasta cpp.

Returns

Funcionalidades C++ em condições de interoperabilidade.

Definition at line 110 of file [Bootng.py](#).

6.3.3.2 `get_team_params()`

```
list[list[str | int]] Booting.Bootng.get_team_params ( ) [static]
```

Verifica existência de arquivo de parâmetros de time, caso não exista, usará o default.

Faremos em tupla para permitir uso mínimo de memória.

Returns

Definition at line 44 of file [Bootng.py](#).

6.3.3.3 `show_spinner()`

```
None Booting.Bootng.show_spinner (
    list[bool] running_flag ) [static]
```

Por motivos estéticos, mostrará um spinner enquanto há o carregamento de módulos C++.

Definition at line 93 of file [Bootng.py](#).

6.3.4 Member Data Documentation

6.3.4.1 `CONFIG_PATH`

```
str Booting.Bootng.CONFIG_PATH = Path(__file__).resolve().parent / "config_team_params.txt"
[static]
```

Definition at line 24 of file [Bootng.py](#).

6.3.4.2 options

`Booting.Booting.options`

Definition at line 31 of file [Booting.py](#).

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

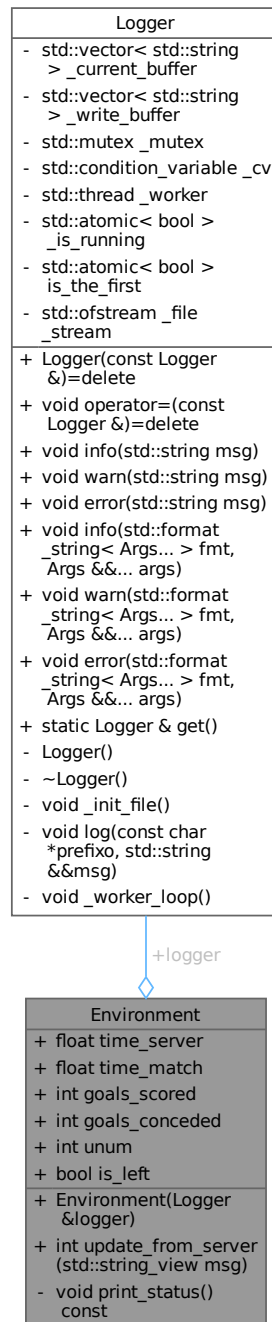
- `src/term/Booting.py`

6.4 Environment Class Reference

Responsável por representar o ambiente externo ao robô

```
#include <Environment.hpp>
```

Collaboration diagram for Environment:



Classes

- class [Parsing](#)

Responsável por prover ferramentas de auxílio de parsing.

Public Types

- enum class [PlayMode](#) : int {
[OUR_KICKOFF](#) = 0 , [OUR_KICK_IN](#) = 1 , [OUR_CORNER_KICK](#) = 2 , [OUR_GOAL_KICK](#) = 3 ,
[OUR_FREE_KICK](#) = 4 , [OUR_PASS](#) = 5 , [OUR_DIR_FREE_KICK](#) = 6 , [OUR_GOAL](#) = 7 ,
[OUR_OFFSIDE](#) = 8 , [THEIR_KICKOFF](#) = 9 , [THEIR_KICK_IN](#) = 10 , [THEIR_CORNER_KICK](#) = 11 ,
[THEIR_GOAL_KICK](#) = 12 , [THEIR_FREE_KICK](#) = 13 , [THEIR_PASS](#) = 14 , [THEIR_DIR_FREE_KICK](#) = 15 ,
[THEIR_GOAL](#) = 16 , [THEIR_OFFSIDE](#) = 17 , [BEFORE_KICKOFF](#) = 18 , [GAME_OVER](#) = 19 ,
[PLAY_ON](#) = 20 }
- enum class [PlayModeGroup](#) : int {
[OUR_KICK](#) = 0 , [THEIR_KICK](#) = 1 , [ACTIVE_BEAM](#) = 2 , [PASSIVE_BEAM](#) = 3 ,
[OTHER](#) = 4 }

Public Member Functions

- [Environment](#) ([Logger](#) &[logger](#))
Construtor da Classe.
- int [update_from_server](#) (std::string_view msg)
Interpretará as mensagens do servidor.

Public Attributes

- [Logger](#) & [logger](#)
- float [time_server](#)
Instante de Tempo do Servidor, útil apenas para sincronização entre agentes.
- float [time_match](#)
Instante de Tempo de Partida.
- int [goals_scored](#)
Nossos Gols, pode ser útil para mudarmos de tática conforme o jogo avança.
- int [goals_conceded](#)
Gols adversários, pode ser útil para mudarmos de tática conforme o jogo avança.
- int [unum](#)
Número do Jogador.
- bool [is_left](#)

Private Member Functions

- void [print_status](#) () const
Apresentará os dados lidos do servidor.

6.4.1 Detailed Description

Responsável por representar o ambiente externo ao robô

Agrupará todos os métodos de interpretação do mundo. Focaremos em performance e eficiência no uso da memória.

Definition at line 17 of file [Environment.hpp](#).

6.4.2 Member Enumeration Documentation

6.4.2.1 PlayMode

```
enum class Environment::PlayMode : int [strong]
```


Enumerator

OUR_KICKOFF	
OUR_KICK_IN	
OUR_CORNER_KICK	
OUR_GOAL_KICK	
OUR_FREE_KICK	
OUR_PASS	
OUR_DIR_FREE_KICK	
OUR_GOAL	
OUR_OFFSIDE	
THEIR_KICKOFF	
THEIR_KICK_IN	
THEIR_CORNER_KICK	
THEIR_GOAL_KICK	
THEIR_FREE_KICK	
THEIR_PASS	
THEIR_DIR_FREE_KICK	
THEIR_GOAL	
THEIR_OFFSIDE	
BEFORE_KICKOFF	
GAME_OVER	
PLAY_ON	

Definition at line 37 of file [Environment.hpp](#).

6.4.2.2 PlayModeGroup

```
enum class Environment::PlayModeGroup : int [strong]
```

Enumerator

OUR_KICK	
THEIR_KICK	
ACTIVE_BEAM	
PASSIVE_BEAM	
OTHER	

Definition at line 65 of file [Environment.hpp](#).

6.4.3 Constructor & Destructor Documentation

6.4.3.1 Environment()

```
Environment::Environment (
    Logger & logger ) [inline]
```

Constructor da Classe.

Definition at line 25 of file [Environment.hpp](#).

6.4.4 Member Function Documentation

6.4.4.1 print_status()

```
void Environment::print_status ( ) const [inline], [private]
```

Apresentará os dados lidos do servidor.

Definition at line 283 of file [Environment.hpp](#).

6.4.4.2 update_from_server()

```
int Environment::update_from_server (
    std::string_view msg ) [inline]
```

Interpretará as mensagens do servidor.

Parameters

<i>msg</i>	Mensagem bruta enviada pelo servidor.
------------	---------------------------------------

Returns

Atualização de todas as variáveis de ambiente.

< Vamos extrair uma tag

< Há apenas 'time'

< Pode ser 'GS' ou 'GYR'

< Tag Desconhecida

< Tag Superior Desconhecida

Definition at line 230 of file [Environment.hpp](#).

6.4.5 Member Data Documentation

6.4.5.1 goals_conceded

```
int Environment::goals_conceded
```

Gols adversários, pode ser útil para mudarmos de tática conforme o jogo avança.

Definition at line 33 of file [Environment.hpp](#).

6.4.5.2 goals_scored

```
int Environment::goals_scored
```

Nossos Gols, pode ser útil para mudarmos de tática conforme o jogo avança.

Definition at line 32 of file [Environment.hpp](#).

6.4.5.3 is_left

```
bool Environment::is_left
```

Definition at line 35 of file [Environment.hpp](#).

6.4.5.4 logger

```
Logger& Environment::logger
```

Definition at line 20 of file [Environment.hpp](#).

6.4.5.5 time_match

```
float Environment::time_match
```

Instante de Tempo de Partida.

Definition at line 31 of file [Environment.hpp](#).

6.4.5.6 time_server

```
float Environment::time_server
```

Instante de Tempo do Servidor, útil apenas para sincronização entre agentes.

Definition at line 30 of file [Environment.hpp](#).

6.4.5.7 unum

```
int Environment::unum
```

Número do Jogador.

Definition at line 34 of file [Environment.hpp](#).

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

- [src/cpp/environment/Environment.hpp](#)

6.5 Logger Class Reference

Singleton para logging assíncrono.

```
#include <Logger.hpp>
```

Collaboration diagram for Logger:

Logger
<ul style="list-style-type: none"> - std::vector< std::string > _current_buffer - std::vector< std::string > _write_buffer - std::mutex _mutex - std::condition_variable _cv - std::thread _worker - std::atomic< bool > _is_running - std::atomic< bool > _is_the_first - std::ofstream _file_stream
<ul style="list-style-type: none"> + Logger(const Logger &)=delete + void operator=(const Logger &)=delete + void info(std::string msg) + void warn(std::string msg) + void error(std::string msg) + void info(std::format_string< Args... > fmt, Args &&... args) + void warn(std::format_string< Args... > fmt, Args &&... args) + void error(std::format_string< Args... > fmt, Args &&... args) + static Logger & get() - Logger() - ~Logger() - void _init_file() - void log(const char *prefixo, std::string &&msg) - void _worker_loop()

Public Member Functions

- `Logger` (const `Logger` &)=delete
- void `operator=` (const `Logger` &)=delete
- void `info` (std::string msg)
Adiciona log nível INFO.
- void `warn` (std::string msg)
Adiciona log nível WARN.
- void `error` (std::string msg)
Adiciona log nível ERROR.
- template<typename... Args>
void `info` (std::format_string< Args... > fmt, Args &&... args)
Log INFO usando C++20 std::format (Alta Performance).
- template<typename... Args>
void `warn` (std::format_string< Args... > fmt, Args &&... args)
Log WARN usando C++20 std::format.
- template<typename... Args>
void `error` (std::format_string< Args... > fmt, Args &&... args)
Log ERROR usando C++20 std::format.

Static Public Member Functions

- static `Logger` & `get` ()
Acesso à instância única.

Private Member Functions

- `Logger` ()
Construtor privado: Inicializa arquivo e thread.
- `~Logger` ()
Destrutor: Sinaliza parada e espera thread terminar.
- void `_init_file` ()
Responsável por criar ambiente de .log.
- void `log` (const char *prefixo, std::string &&msg)
Responsável por providenciar genérica chamada de impressão em .log.
- void `_worker_loop` ()
Loop da thread de background, responsável por escrever no arquivo .log da melhor forma possível.

Private Attributes

- std::vector< std::string > `_current_buffer`
- std::vector< std::string > `_write_buffer`
- std::mutex `_mutex`
- std::condition_variable `_cv`
- std::thread `_worker`
- std::atomic< bool > `_is_running`
- std::atomic< bool > `is_the_first` = `True`
- std::ofstream `_file_stream`

6.5.1 Detailed Description

Singleton para logging assíncrono.

Focada em performance utiliza uma lógica de fila de mensagens.

Definition at line 25 of file [Logger.hpp](#).

6.5.2 Constructor & Destructor Documentation

6.5.2.1 `Logger()` [1/2]

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

6.5.2.2 `Logger()` [2/2]

```
Logger::Logger ( ) [inline], [private]
```

Construtor privado: Inicializa arquivo e thread.

Reservará 1000 slots para evitarmos realocações

Definition at line 103 of file [Logger.hpp](#).

6.5.2.3 `~Logger()`

```
Logger::~~Logger ( ) [inline], [private]
```

Destrutor: Sinaliza parada e espera thread terminar.

< Informa a thread da condição de encerramento

Definition at line 112 of file [Logger.hpp](#).

6.5.3 Member Function Documentation

6.5.3.1 `_init_file()`

```
void Logger::_init_file ( ) [inline], [private]
```

Responsável por criar ambiente de .log.

Possui uma lógica para garantir que logs sejam únicos.

Definition at line 125 of file [Logger.hpp](#).

6.5.3.2 `_worker_loop()`

```
void Logger::_worker_loop ( ) [inline], [private]
```

Loop da thread de background, responsável por escrever no arquivo .log da melhor forma possível.

Função de alto nível < Espera até ter dados ou ser instruído a encerrar

< Agora escrevemos no disco SEM bloquear quem quer adicionar logs

Definition at line 172 of file [Logger.hpp](#).

6.5.3.3 `error()` [1/2]

```
template<typename... Args>
void Logger::error (
    std::format_string< Args... > fmt,
    Args &&... args ) [inline]
```

Log ERROR usando C++20 `std::format`.

Definition at line 83 of file [Logger.hpp](#).

6.5.3.4 `error()` [2/2]

```
void Logger::error (
    std::string msg ) [inline]
```

Adiciona log nível ERROR.

Parameters

<i>msg</i>	Mensagem a ser imprimida.
------------	---------------------------

Recebe por valor para permitir `std::move` (otimização de r-values).

Definition at line 57 of file [Logger.hpp](#).

6.5.3.5 `get()`

```
static Logger & Logger::get ( ) [inline], [static]
```

Acesso à instância única.

Definition at line 30 of file [Logger.hpp](#).

6.5.3.6 info() [1/2]

```
template<typename... Args>  
void Logger::info (  
    std::format_string< Args... > fmt,  
    Args &&... args ) [inline]
```

Log INFO usando C++20 std::format (Alta Performance).

Parameters

<i>fmt</i>	A string de formatação (ex: "Valor: {}"). Deve ser uma string literal (constante).
<i>args</i>	Os argumentos a serem formatados.

Definition at line 65 of file [Logger.hpp](#).

6.5.3.7 info() [2/2]

```
void Logger::info (
    std::string msg ) [inline]
```

Adiciona log nível INFO.

Parameters

<i>msg</i>	Mensagem a ser imprimida.
------------	---------------------------

Recebe por valor para permitir std::move (otimização de r-values).

Definition at line 41 of file [Logger.hpp](#).

6.5.3.8 log()

```
void Logger::log (
    const char * prefixo,
    std::string && msg ) [inline], [private]
```

Responsável por providenciar genérica chamada de impressão em .log.

Parameters

<i>prefixo</i>	Cabeçalho que será colocada antes da mensagem.
<i>msg</i>	Mensagem principal. Usa lock apenas para empurrar no vetor (operação de nanossegundos).

< Esse lock_guard trava enquanto estiver nesse escopo

Definition at line 150 of file [Logger.hpp](#).

6.5.3.9 operator=()

```
void Logger::operator= (
    const Logger & ) [delete]
```

6.5.3.10 warn() [1/2]

```
template<typename... Args>
void Logger::warn (
    std::format_string< Args... > fmt,
    Args &&... args ) [inline]
```

Log WARN usando C++20 std::format.

Definition at line 75 of file [Logger.hpp](#).

6.5.3.11 warn() [2/2]

```
void Logger::warn (
    std::string msg ) [inline]
```

Adiciona log nível WARN.

Parameters

<i>msg</i>	Mensagem a ser imprimida.
------------	---------------------------

Recebe por valor para permitir std::move (otimização de r-values).

Definition at line 49 of file [Logger.hpp](#).

6.5.4 Member Data Documentation

6.5.4.1 _current_buffer

```
std::vector<std::string> Logger::_current_buffer [private]
```

Definition at line 89 of file [Logger.hpp](#).

6.5.4.2 _cv

```
std::condition_variable Logger::_cv [private]
```

Definition at line 93 of file [Logger.hpp](#).

6.5.4.3 _file_stream

```
std::ofstream Logger::_file_stream [private]
```

Definition at line 97 of file [Logger.hpp](#).

6.5.4.4 `_is_running`

```
std::atomic<bool> Logger::_is_running [private]
```

Definition at line 95 of file [Logger.hpp](#).

6.5.4.5 `_mutex`

```
std::mutex Logger::_mutex [private]
```

Definition at line 92 of file [Logger.hpp](#).

6.5.4.6 `_worker`

```
std::thread Logger::_worker [private]
```

Definition at line 94 of file [Logger.hpp](#).

6.5.4.7 `_write_buffer`

```
std::vector<std::string> Logger::_write_buffer [private]
```

Definition at line 90 of file [Logger.hpp](#).

6.5.4.8 `is_the_first`

```
std::atomic<bool> Logger::is_the_first = True [private]
```

Definition at line 96 of file [Logger.hpp](#).

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

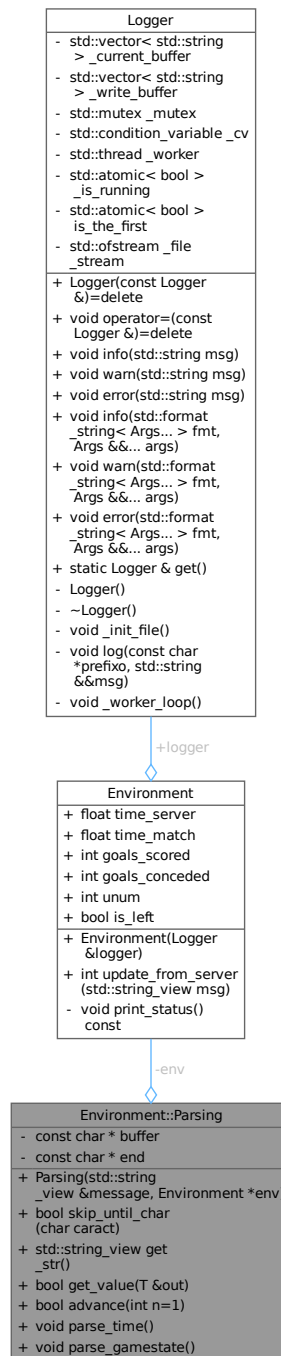
- [src/cpp/logger/Logger.hpp](#)

6.6 Environment::Parsing Class Reference

Responsável por prover ferramentas de auxílio de parsing.

```
#include <Environment.hpp>
```

Collaboration diagram for Environment::Parsing:



Public Member Functions

- [Parsing](#) (std::string_view &message, [Environment](#) *env)
Construtor do [Parsing](#) dedica à interpretação.
- bool [skip_until_char](#) (char caract)
Avançará até encontrar um determinado caractere de parada, pulando-o em seguida.
- std::string_view [get_str](#) ()
Ignorando eventuais ' ', '(' e ')', obterá a próxima string, encerrando apenas a encontrar ' '. Pulando este último caractere.
- template<typename T >
bool [get_value](#) (T &out)
Fará a conversão de caracteres em inteiro ou float, dependendo do tipo de referência dado.
- bool [advance](#) (int n=1)
Avançará o cursor uma determinada quantidade.
- void [parse_time](#) ()
Responsável pela interpretação da mensagem de 'time'.
- void [parse_gamestate](#) ()
Responsável pela interpretação da mensagem de 'GS'.

Private Attributes

- const char * [buffer](#) = nullptr
Permitirá-nos saber o ponto da mensagem que estamos.
- const char * [end](#) = nullptr
Permitirá-nos saber o ponto final.
- [Environment](#) * [env](#) = nullptr
Permitirá-nos modificar atributos.

6.6.1 Detailed Description

Responsável por prover ferramentas de auxílio de parsing.

Centralizará todas as funções inerentes ao parsing das mensagens.

Definition at line 86 of file [Environment.hpp](#).

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Parsing()

```
Environment::Parsing::Parsing (
    std::string_view & message,
    Environment * env ) [inline]
```

Construtor do [Parsing](#) dedica à interpretação.

Parameters

<i>msg</i>	Mensagem bruta enviada pelo servidor.
------------	---------------------------------------

Returns

Atualização de todas as variáveis de ambiente.

Definition at line 100 of file [Environment.hpp](#).

6.6.3 Member Function Documentation

6.6.3.1 advance()

```
bool Environment::Parsing::advance (
    int n = 1 ) [inline]
```

Avançará o cursor uma determinada quantidade.

Parameters

<i>n</i>	quantidade de avanços desejados
----------	---------------------------------

Returns

False, se o avanço não foi permitido. True, caso contrário.

Definition at line 157 of file [Environment.hpp](#).

6.6.3.2 get_str()

```
std::string_view Environment::Parsing::get_str ( ) [inline]
```

Ignorando eventuais ' ', '(' e ')', obterá a próxima string, encerrando apenas a encontrar ' '. Pulando este último caractere.

Returns

String_view da string.

Definition at line 129 of file [Environment.hpp](#).

6.6.3.3 get_value()

```
template<typename T >
bool Environment::Parsing::get_value (
    T & out ) [inline]
```

Fará a conversão de caracteres em inteiro ou float, dependendo do tipo de referência dado.

Iniciará a leitura a partir do ponto que buffer se encontra. Encerrará ao encontrar ' ' ou ')', pulando este.

Parameters

out	<i>Variável</i>	que receberá o valor
-----	-----------------	----------------------

Returns

True, se não houve erro. False, caso contrário.

Definition at line 145 of file [Environment.hpp](#).

6.6.3.4 parse_gamestate()

```
void Environment::Parsing::parse_gamestate ( ) [inline]
```

Responsável pela interpretação da mensagem de 'GS'.

< Obteremos as subtags

< Poderá ser 'sl', 'sr'

< Há apenas 'pm'

< Há 'time' e 'team'

< Há apenas o 'u'

Definition at line 181 of file [Environment.hpp](#).

6.6.3.5 parse_time()

```
void Environment::Parsing::parse_time ( ) [inline]
```

Responsável pela interpretação da mensagem de 'time'.

< Vamos ter fé que nunca será diferente.

< Sairemos da tag 'time'

Definition at line 165 of file [Environment.hpp](#).

6.6.3.6 skip_until_char()

```
bool Environment::Parsing::skip_until_char (
    char caract ) [inline]
```

Avançará até encontrar um determinado caractere de parada, pulando-o em seguida.

Parameters

<i>caract</i>	Caractere de Parada.
---------------	----------------------

Returns

True, caso encontre corretamente. False, caso chegue ao final da mensagem.

Definition at line 115 of file [Environment.hpp](#).

6.6.4 Member Data Documentation

6.6.4.1 buffer

```
const char* Environment::Parsing::buffer = nullptr [private]
```

Permitirá-nos saber o ponto da mensagem que estamos.

Definition at line 88 of file [Environment.hpp](#).

6.6.4.2 end

```
const char* Environment::Parsing::end = nullptr [private]
```

Permitirá-nos saber o ponto final.

Definition at line 89 of file [Environment.hpp](#).

6.6.4.3 env

```
Environment\* Environment::Parsing::env = nullptr [private]
```

Permitirá-nos modificar atributos.

Definition at line 90 of file [Environment.hpp](#).

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

- [src/cpp/environment/Environment.hpp](#)

6.7 Printing.Printing Class Reference

Responsável pela comunicação usuário - terminal.

Collaboration diagram for Printing.Printing:

Printing.Printing
+ bool IF_IN_DEBUG
+ dict TABLE_COLORS
+ CONSOLE
+ None print_message (str message, str role=None)
+ None ConsoleRenderable print_table(list[str] columns, list[list] dados, str header_style="bold", dict[int, str] row_style=None, int width=None, dict[str, str] column_styles=None, dict[str, str] column_justify=None, dict [str, int] column_widths=None, bool renderable=False)
+ get_input(int bytes _to_be_read, Callable return_type=str)

Static Public Member Functions

- None [print_message](#) (str message, str role=None)
Apresentará uma mensagem estilizada de forma específica.
- None|ConsoleRenderable [print_table](#) (list[str] columns, list[list] dados, str header_style="bold", dict[int, str] row_style=None, int width=None, dict[str, str] column_styles=None, dict[str, str] column_justify=None, dict[str, int] column_widths=None, bool renderable=False)
Apresentará uma tabela completamente personalizada.
- [get_input](#) (int bytes_to_be_read, Callable return_type=str)
Função complexa que fará leitura de entrada do usuário.

Static Public Attributes

- bool [IF_IN_DEBUG](#) = True
- dict [TABLE_COLORS](#)
- [CONSOLE](#) = Console()

6.7.1 Detailed Description

Responsável pela comunicação usuário - terminal.

Definition at line 13 of file [Printing.py](#).

6.7.2 Member Function Documentation

6.7.2.1 `get_input()`

```
Printing.Printing.get_input (
    int bytes_to_be_read,
    Callable return_type = str ) [static]
```

Função complexa que fará leitura de entrada do usuário.

Tome cuidado com a execução dessa função, pois ela é poderosa

Parameters

<i>return_type</i>	Tipo de entrada a ser retornado
<i>bytes_to_be_read</i>	Quantidade de Bytes que serão lidos

Returns

Entrada do usuário

Definition at line 116 of file [Printing.py](#).

6.7.2.2 `print_message()`

```
None Printing.Printing.print_message (
    str message,
    str role = None ) [static]
```

Apresentará uma mensagem estilizada de forma específica.

Parameters

<i>message</i>	Mensagem a ser apresentada
<i>role</i>	String indicando qual o motivo da mensagem

Há uma quantidade específica de roles possíveis:

- info
- warning
- error

Caso nenhuma dessas seja inserida, há a possibilidade de inserir o comando ASCII de uma vez.

Definition at line 26 of file [Printing.py](#).

6.7.2.3 print_table()

```
None | ConsoleRenderable Printing.Printing.print_table (
    list[str] columns,
    list[list] dados,
    str header_style = "bold",
    dict[int, str] row_style = None,
    int width = None,
    dict[str, str] column_styles = None,
    dict[str, str] column_justify = None,
    dict[str, int] column_widths = None,
    bool renderable = False ) [static]
```

Apresentará uma tabela completamente personalizada.

Parameters

<i>columns</i>	Lista dos nomes das colunas
<i>data</i>	Lista de listas com os valores de linhas

Assume os seguintes parâmetros de personalização: columns: Lista de nomes das colunas data: Lista de listas com dados das linhas header_style: Estilo do cabeçalho row_styles: Estilos alternados para linhas width: Largura fixa da tabela column_styles: {nome_coluna: estilo} column_justify: {nome_coluna: "left"/"center"/"right"} column_widths: {nome_coluna: largura}

Definition at line 61 of file [Printing.py](#).

6.7.3 Member Data Documentation

6.7.3.1 CONSOLE

```
Printing.Printing.CONSOLE = Console() [static]
```

Definition at line 23 of file [Printing.py](#).

6.7.3.2 IF_IN_DEBUG

```
bool Printing.Printing.IF_IN_DEBUG = True [static]
```

Definition at line 17 of file [Printing.py](#).

6.7.3.3 TABLE_COLORS

`dict Printing.Printing.TABLE_COLORS [static]`

Initial value:

```
= {  
    "info": "\033[1;36m",  
    "warning": "\033[1;33m",  
    "error": "\033[1;31m"  
}
```

Definition at line 18 of file [Printing.py](#).

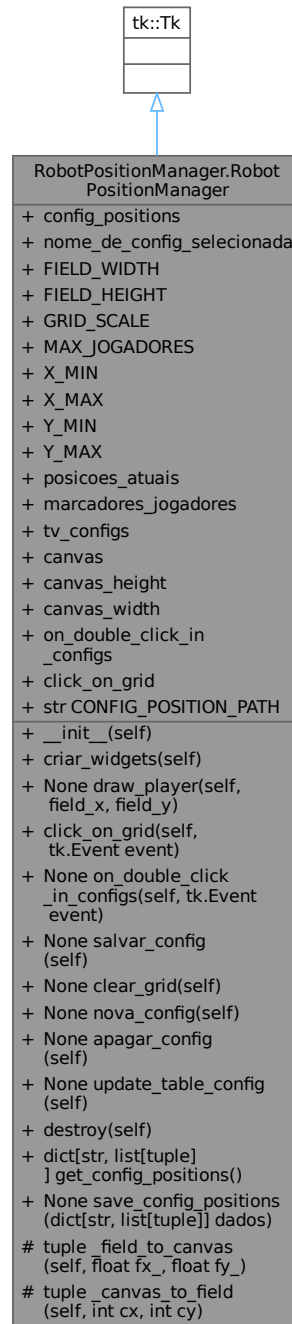
The documentation for this class was generated from the following file:

- [src/term/Printing.py](#)

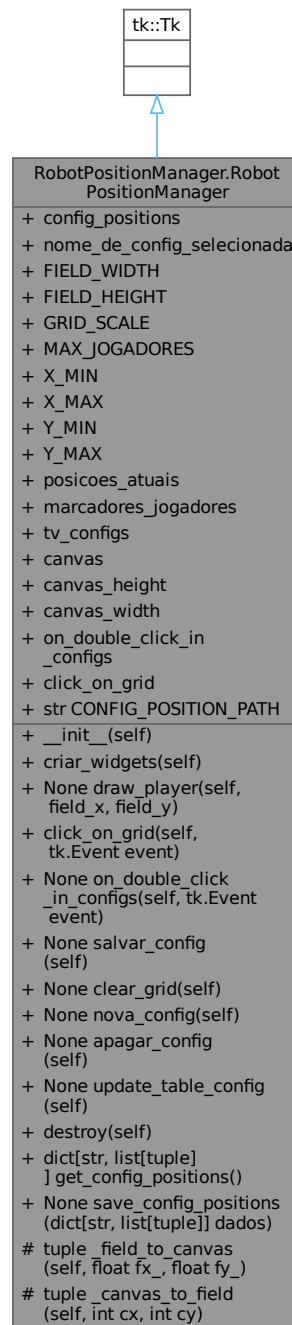
6.8 RobotPositionManager.RobotPositionManager Class Reference

Responsável por permitir ao usuário a criação de diversas formações táticas.

Inheritance diagram for RobotPositionManager.RobotPositionManager:



Collaboration diagram for RobotPositionManager.RobotPositionManager:



Public Member Functions

- `__init__` (self)
Construtor da Classe, inicializa variáveis importantes, como o próprio dicionário.
- `criar_widgets` (self)
Disporá os widgets da interface de forma inteligente, provendo informações úteis.
- `None draw_player` (self, field_x, field_y)

- *Desenharemos um jogador na posição especificada.*
• [click_on_grid](#) (self, tk.Event event)
Responsável por identificar onde o usuário clicou e adicionar essa posição na lista.
- None [on_double_click_in_configs](#) (self, tk.Event event)
Responsável por plotar a configuração de jogadores selecionada.
- None [salvar_config](#) (self)
Salvará uma configuração selecionada.
- None [clear_grid](#) (self)
Responsável por limpar as posições e a grade.
- None [nova_config](#) (self)
Prepará uma nova configuração para ser criada.
- None [apagar_config](#) (self)
Apagará uma configuração selecionada.
- None [update_table_config](#) (self)
Responsável por atualizar e preencher tabela de configurações de posição.
- [destroy](#) (self)

Static Public Member Functions

- dict[str, list[tuple]] [get_config_positions](#) ()
Verificará existência do arquivo binário correspondente ao dicionário.
- None [save_config_positions](#) (dict[str, list[tuple]] dados)
Responsável por salvar uma estrutura de dados em arquivo binário.

Public Attributes

- [config_positions](#)
- [nome_de_config_selecionada](#)
- [FIELD_WIDTH](#)
- [FIELD_HEIGHT](#)
- [GRID_SCALE](#)
- [MAX_JOGADORES](#)
- [X_MIN](#)
- [X_MAX](#)
- [Y_MIN](#)
- [Y_MAX](#)
- [posicoes_atuais](#)
- [marcadores_jogadores](#)
- [tv_configs](#)
- [canvas](#)
- [canvas_height](#)
- [canvas_width](#)
- [on_double_click_in_configs](#)
- [click_on_grid](#)

Static Public Attributes

- str [CONFIG_POSITION_PATH](#) = Path(__file__).resolve().parents[1] / "agent" / "tactical_formation.pkl"

Protected Member Functions

- tuple `_field_to_canvas` (self, float fx_, float fy_)
Responsável por converter coordenadas do campo para pixels no canvas.
- tuple `_canvas_to_field` (self, int cx, int cy)
Converterá o pixel clicado para o quadrado correspondente.

6.8.1 Detailed Description

Responsável por permitir ao usuário a criação de diversas formações táticas.

Focada em diversão e customização, gerencia um binário que é a representação de dicionário de listas que contém as 11 posições. Por ter esse objetivo, não faz sentido que haja essa função na lógica geral dos agentes.

Definition at line 11 of file [RobotPositionManager.py](#).

6.8.2 Constructor & Destructor Documentation

6.8.2.1 `__init__()`

```
RobotPositionManager.RobotPositionManager.__init__ (
    self )
```

Construtor da Classe, inicializa variáveis importantes, como o próprio dicionário.

Definition at line 23 of file [RobotPositionManager.py](#).

6.8.3 Member Function Documentation

6.8.3.1 `_canvas_to_field()`

```
tuple RobotPositionManager.RobotPositionManager._canvas_to_field (
    self,
    int cx,
    int cy ) [protected]
```

Converterá o pixel clicado para o quadrado correspondente.

Parameters

<code>cx</code>	Posição X do pixel
<code>cy</code>	Posição Y do pixel

Returns

tupla de posições reais

Definition at line 102 of file [RobotPositionManager.py](#).

6.8.3.2 _field_to_canvas()

```
tuple RobotPositionManager.RobotPositionManager._field_to_canvas (
    self,
    float fx_,
    float fy_ ) [protected]
```

Responsável por converter coordenadas do campo para pixels no canvas.

Parameters

fx_{\leftrightarrow} _↔	Coordenada real em x
fy_{\leftrightarrow} _↔	Coordenada real em y

Returns

Coordenadas corrigidas para o grid

Definition at line 90 of file [RobotPositionManager.py](#).

6.8.3.3 apagar_config()

```
None RobotPositionManager.RobotPositionManager.apagar_config (
    self )
```

Apagará uma configuração selecionada.

Definition at line 355 of file [RobotPositionManager.py](#).

6.8.3.4 clear_grid()

```
None RobotPositionManager.RobotPositionManager.clear_grid (
    self )
```

Responsável por limpar as posições e a grade.

Definition at line 267 of file [RobotPositionManager.py](#).

6.8.3.5 click_on_grid()

```
RobotPositionManager.RobotPositionManager.click_on_grid (
    self,
    tk.Event event )
```

Responsável por identificar onde o usuário clicou e adicionar essa posição na lista.

Parameters

<i>event</i>	Argumento default do bind
--------------	---------------------------

Definition at line 192 of file [RobotPositionManager.py](#).

6.8.3.6 criar_widgets()

```
RobotPositionManager.RobotPositionManager.criar_widgets (
    self )
```

Disporá os widgets da interface de forma inteligente, provendo informações úteis.

Definition at line 127 of file [RobotPositionManager.py](#).

6.8.3.7 destroy()

```
RobotPositionManager.RobotPositionManager.destroy (
    self )
```

Definition at line 390 of file [RobotPositionManager.py](#).

6.8.3.8 draw_player()

```
None RobotPositionManager.RobotPositionManager.draw_player (
    self,
    field_x,
    field_y )
```

Desenharemos um jogador na posição especificada.

Parameters

<i>field_x</i>	Posição real em X
<i>field_y</i>	Posição real em Y

Definition at line 174 of file [RobotPositionManager.py](#).

6.8.3.9 get_config_positions()

```
dict[str, list[tuple]] RobotPositionManager.RobotPositionManager.get_config_positions ( )
[static]
```

Verificará existência do arquivo binário correspondente ao dicionário.

Returns

Caso exista, o retornará restaurado. Caso não, retornará um dicionário vazio.

Definition at line 62 of file [RobotPositionManager.py](#).

6.8.3.10 nova_config()

```
None RobotPositionManager.RobotPositionManager.nova_config (
    self )
```

Prepará uma nova configuração para ser criada.

Definition at line 332 of file [RobotPositionManager.py](#).

6.8.3.11 on_double_click_in_configs()

```
None RobotPositionManager.RobotPositionManager.on_double_click_in_configs (
    self,
    tk.Event event )
```

Responsável por plotar a configuração de jogadores selecionada.

Parameters

<i>event</i>	Argumento Default de bind
--------------	---------------------------

Definition at line 219 of file [RobotPositionManager.py](#).

6.8.3.12 salvar_config()

```
None RobotPositionManager.RobotPositionManager.salvar_config (
    self )
```

Salvará uma configuração selecionada.

Definition at line 239 of file [RobotPositionManager.py](#).

6.8.3.13 save_config_positions()

```
None RobotPositionManager.RobotPositionManager.save_config_positions (
    dict[str, list[tuple]] dados ) [static]
```

Responsável por salvar uma estrutura de dados em arquivo binário.

Parameters

<i>dados</i>	Estrutura de dados a ser salva
--------------	--------------------------------

Definition at line 77 of file [RobotPositionManager.py](#).

6.8.3.14 update_table_config()

```
None RobotPositionManager.RobotPositionManager.update_table_config (
    self )
```

Responsável por atualizar e preencher tabela de configurações de posição.

Definition at line 379 of file [RobotPositionManager.py](#).

6.8.4 Member Data Documentation

6.8.4.1 canvas

```
RobotPositionManager.RobotPositionManager.canvas
```

Definition at line 52 of file [RobotPositionManager.py](#).

6.8.4.2 canvas_height

```
RobotPositionManager.RobotPositionManager.canvas_height
```

Definition at line 53 of file [RobotPositionManager.py](#).

6.8.4.3 canvas_width

```
RobotPositionManager.RobotPositionManager.canvas_width
```

Definition at line 54 of file [RobotPositionManager.py](#).

6.8.4.4 click_on_grid

```
RobotPositionManager.RobotPositionManager.click_on_grid
```

Definition at line 170 of file [RobotPositionManager.py](#).

6.8.4.5 CONFIG_POSITION_PATH

```
str RobotPositionManager.RobotPositionManager.CONFIG_POSITION_PATH = Path(__file__).resolve().parents[1]
/ "agent" / "tactical_formation.pkl" [static]
```

Definition at line 20 of file [RobotPositionManager.py](#).

6.8.4.6 config_positions

RobotPositionManager.RobotPositionManager.config_positions

Definition at line 33 of file [RobotPositionManager.py](#).

6.8.4.7 FIELD_HEIGHT

RobotPositionManager.RobotPositionManager.FIELD_HEIGHT

Definition at line 38 of file [RobotPositionManager.py](#).

6.8.4.8 FIELD_WIDTH

RobotPositionManager.RobotPositionManager.FIELD_WIDTH

Definition at line 37 of file [RobotPositionManager.py](#).

6.8.4.9 GRID_SCALE

RobotPositionManager.RobotPositionManager.GRID_SCALE

Definition at line 39 of file [RobotPositionManager.py](#).

6.8.4.10 marcadores_jogadores

RobotPositionManager.RobotPositionManager.marcadores_jogadores

Definition at line 48 of file [RobotPositionManager.py](#).

6.8.4.11 MAX_JOGADORES

RobotPositionManager.RobotPositionManager.MAX_JOGADORES

Definition at line 40 of file [RobotPositionManager.py](#).

6.8.4.12 nome_de_config_selecionada

RobotPositionManager.RobotPositionManager.nome_de_config_selecionada

Definition at line 34 of file [RobotPositionManager.py](#).

6.8.4.13 on_double_click_in_configs

RobotPositionManager.RobotPositionManager.on_double_click_in_configs

Definition at line 146 of file [RobotPositionManager.py](#).

6.8.4.14 posicoes_atuais

`RobotPositionManager.RobotPositionManager.posicoes_atuais`

Definition at line 47 of file [RobotPositionManager.py](#).

6.8.4.15 tv_configs

`RobotPositionManager.RobotPositionManager.tv_configs`

Definition at line 51 of file [RobotPositionManager.py](#).

6.8.4.16 X_MAX

`RobotPositionManager.RobotPositionManager.X_MAX`

Definition at line 42 of file [RobotPositionManager.py](#).

6.8.4.17 X_MIN

`RobotPositionManager.RobotPositionManager.X_MIN`

Definition at line 41 of file [RobotPositionManager.py](#).

6.8.4.18 Y_MAX

`RobotPositionManager.RobotPositionManager.Y_MAX`

Definition at line 44 of file [RobotPositionManager.py](#).

6.8.4.19 Y_MIN

`RobotPositionManager.RobotPositionManager.Y_MIN`

Definition at line 43 of file [RobotPositionManager.py](#).

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

- [src/utis/RobotPositionManager.py](#)

6.9 ServerComm.ServerComm Class Reference

Responsável pela comunicação com servidor.

Collaboration diagram for ServerComm.ServerComm:

ServerComm.ServerComm
+ buffer_size + buffer + socket + message_queue + unum + env
+ <code>__init__</code> (self, list [list[str]] creation_options, environment, list other_players) + None <code>send_immediate</code> (self, bytes message) + None <code>receive</code> (self) + None <code>commit</code> (self, bytes message) + None <code>close</code> (self) + None <code>send</code> (self) + None <code>clear_queue</code> (self) + <code>commit_beam</code> (self, list vector_position2d, float rotation) - None <code>__receive_async</code> (self, list other_players)

Public Member Functions

- `__init__`(self, list[list[str]] creation_options, environment, list other_players)
Construtor da classe, inicializando buffers e a conexão de cada agente com servidor.
- None `send_immediate`(self, bytes message)
Envia uma mensagem instantânea ao servidor, verificando se a conexão continua ativa.
- None `receive`(self)
Receberá informações diretamente do servidor, fazendo todas as verificações necessárias.
- None `commit`(self, bytes message)
Responsável por adicionar uma nova mensagem à fila de mensagens.
- None `close`(self)
Responsável por fazer o encerramento dos canais de comunicação.

- None [send](#) (self)
Enviar  ao servidor todas as mensagens commitadas.
- None [clear_queue](#) (self)
Limpar  a fila de commits.
- [commit_beam](#) (self, list vector_position2d, float rotation)
Comando de beam oficial do agente.

Public Attributes

- [buffer_size](#)
- [buffer](#)
- [socket](#)
- [message_queue](#)
- [unum](#)
- [env](#)

Private Member Functions

- None [__receive_async](#) (self, list other_players)
Respons vel por esperar resposta do servidor de forma ass ncrona, sem impedir fluxo de execu  o.

6.9.1 Detailed Description

Respons vel pela comunica  o com servidor.

Definition at line 10 of file [ServerComm.py](#).

6.9.2 Constructor & Destructor Documentation

6.9.2.1 `__init__()`

```
ServerComm.ServerComm.__init__ (
    self,
    list[list[str]] creation_options,
    environment,
    list other_players )
```

Construtor da classe, inicializando buffers e a conex o de cada agente com servidor.

Parameters

<i>creation_options</i>	Lista de par�metros de cria��o, self ainda n�o foi inclu�do na lista.
<i>environment</i>	
<i>other_players</i>	

Definition at line 15 of file [ServerComm.py](#).

6.9.3 Member Function Documentation

6.9.3.1 __receive_async()

```
None ServerComm.ServerComm.__receive_async (
    self,
    list other_players ) [private]
```

Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de execução.

Essa função foi criada com o único propósito de impedir que a espera por resposta do servidor interrompa o fluxo de execução. Não deve ser executada posteriormente.

Parameters

<i>other_players</i>	Lista de jogadores de mesmo time presentes na partida
----------------------	---

Definition at line 144 of file [ServerComm.py](#).

6.9.3.2 clear_queue()

```
None ServerComm.ServerComm.clear_queue (
    self )
```

Limpará a fila de commits.

Definition at line 216 of file [ServerComm.py](#).

6.9.3.3 close()

```
None ServerComm.ServerComm.close (
    self )
```

Responsável por fazer o encerramento dos canais de comunicação.

Definition at line 189 of file [ServerComm.py](#).

6.9.3.4 commit()

```
None ServerComm.ServerComm.commit (
    self,
    bytes message )
```

Responsável por adicionar uma nova mensagem à fila de mensagens.

Parameters

<i>message</i>	String em bytes a ser adicionada à fila
----------------	---

Definition at line 181 of file [ServerComm.py](#).

6.9.3.5 commit_beam()

```
ServerComm.ServerComm.commit_beam (
    self,
    list vector_position2d,
    float rotation )
```

Comando de beam oficial do agente.

Parameters

<i>vector_position2d</i>	Sequência de dois valores, x e y finais do agente
<i>rotation</i>	Valor de rotação a ser dado ao robô

Definition at line 223 of file [ServerComm.py](#).

6.9.3.6 receive()

```
None ServerComm.ServerComm.receive (
    self )
```

Receberá informações diretamente do servidor, fazendo todas as verificações necessárias.

Definition at line 94 of file [ServerComm.py](#).

6.9.3.7 send()

```
None ServerComm.ServerComm.send (
    self )
```

Enviará ao servidor todas as mensagens commitadas.

Definition at line 196 of file [ServerComm.py](#).

6.9.3.8 send_immediate()

```
None ServerComm.ServerComm.send_immediate (
    self,
    bytes message )
```

Envia uma mensagem instantânea ao servidor, verificando se a conexão continua ativa.

Parameters

<i>message</i>	String em forma de bytes para ser transmitida
----------------	---

Coloca-se na frente uma informação de tamanho da mensagem dentro de 4 bytes.

Definition at line 79 of file [ServerComm.py](#).

6.9.4 Member Data Documentation

6.9.4.1 buffer

```
ServerComm.ServerComm.buffer
```

Definition at line 25 of file [ServerComm.py](#).

6.9.4.2 buffer_size

```
ServerComm.ServerComm.buffer_size
```

Definition at line 24 of file [ServerComm.py](#).

6.9.4.3 env

```
ServerComm.ServerComm.env
```

Definition at line 35 of file [ServerComm.py](#).

6.9.4.4 message_queue

```
ServerComm.ServerComm.message_queue
```

Definition at line 33 of file [ServerComm.py](#).

6.9.4.5 socket

```
ServerComm.ServerComm.socket
```

Definition at line 26 of file [ServerComm.py](#).

6.9.4.6 unum

```
ServerComm.ServerComm.unum
```

Definition at line 34 of file [ServerComm.py](#).

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

- [src/communication/ServerComm.py](#)

Chapter 7

File Documentation

7.1 src/agent/Agent.py File Reference

Implementação de Lógica de Agente de Campo.

Classes

- class [Agent.Agent](#)

Classe que representará os agentes de campo, possuindo métodos correspondentes.

Namespaces

- namespace [Agent](#)

7.1.1 Detailed Description

Implementação de Lógica de Agente de Campo.

Definition in file [Agent.py](#).

7.2 Agent.py

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

```
00001 """
00002 @file Agent.py
00003 @brief Implementação de Lógica de Agente de Campo
00004 """
00005 from agent.BaseAgent import BaseAgent
00006
00007 class Agent(BaseAgent):
00008     """
00009     @brief Classe que representará os agentes de campo, possuindo métodos correspondentes.
00010     """
00011
00012     def __init__(self, creation_options: list[list[str | int]]):
00013         """
00014         @brief Construtor da classe agente de campo, inicializando informações gerais.
00015         @param creation_options Lista de Parâmetros de Criação de Agente
```

```

00016         @details
00017         Parâmetros presentes em `creation_options`:
00018             - IP Server
00019             - Porta de Agente
00020             - Porta de Monitor
00021             - Nome do time
00022             - Número de Uniforme
00023             - Tipo de Robô
00024             - Tiro livre Penâlti
00025             - Proxy
00026             - Modo de Debug
00027         """
00028
00029         self.unum = creation_options[4][1]
00030         creation_options[5][1] = (0,1,1,1,2,3,3,3,4,4,4)[self.unum - 1]
00031
00032         super().__init__(creation_options)
00033

```

7.3 src/agent/AgentPenalty.py File Reference

Implementação de Lógica de Goleiro.

Namespaces

- namespace [AgentPenalty](#)

7.3.1 Detailed Description

Implementação de Lógica de Goleiro.

Definition in file [AgentPenalty.py](#).

7.4 AgentPenalty.py

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

```

00001 """
00002 @file AgentPenalty.py
00003 @brief Implementação de Lógica de Goleiro
00004 """

```

7.5 src/agent/BaseAgent.py File Reference

Implementação da classe de jogador base, que deve ser comum a todos os agentes.

Classes

- class [BaseAgent.BaseAgent](#)

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

Namespaces

- namespace [BaseAgent](#)

7.5.1 Detailed Description

Implementação da classe de jogador base, que deve ser comum a todos os agentes.

Definition in file [BaseAgent.py](#).

7.6 BaseAgent.py

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

```

00001 """
00002 @file BaseAgent.py
00003 @brief Implementação da classe de jogador base, que deve ser comum a todos os agentes.
00004 """
00005 from abc import ABC # para conseguirmos criar classes abstratas em Python
00006 from communication.ServerComm import ServerComm
00007 from pathlib import Path
00008 import pickle
00009
00010 class BaseAgent(ABC):
00011     """
00012     @brief Classe que agrupará todas as funcionalidades comuns a qualquer agente.
00013     """
00014
00015     AGENTS_IN_THE_MATCH = []
00016     INITIAL_POSITION = []
00017
00018     def __init__(self, creation_options: list[list[str | int]]):
00019         """
00020         @brief Construtor da classe base de agente, chamando todos os construtores de outras
00021         classes mínimas para cada agente.
00022         @param creation_options Lista de Parâmetros de Criação de Agente
00023         """
00024
00025         # --- Importações do C++ ---
00026         from cpp.logger.logger import Logger
00027         from cpp.environment.environment import Environment
00028
00029         self.logger = Logger.get() # Todos os jogadores utilizarão o único
00030         self.env = Environment(self.logger)
00031         self.scom = ServerComm(
00032             creation_options,
00033             self.env,
00034             # Passamos o ponteiro da lista de jogadores
00035             # Conforme eles são inseridos, teremos novos na partida
00036             BaseAgent.AGENTS_IN_THE_MATCH
00037         )
00038         # Chamaremos os construtores mínimos conforme formos criando-os
00039
00040         self.unum = creation_options[4][1]
00041         # Note que colocamos apenas por último
00042         BaseAgent.AGENTS_IN_THE_MATCH.append(self)
00043
00044         # Garantimos que as posições são existentes
00045         # E executamos apenas uma vez
00046         if not BaseAgent.INITIAL_POSITION:
00047             with open(
00048                 Path(__file__).resolve().parent / "tactical_formation.pkl",
00049                 "rb"
00050             ) as f:
00051                 BaseAgent.INITIAL_POSITION = pickle.load(f) ["default"]
00052
00053         self.init_position = BaseAgent.INITIAL_POSITION[self.unum - 1]
00054
00055     def beam(self) -> None:
00056         """
00057         @brief Responsável por gerenciar o teletransporte dos jogadores
00058         """
00059
00060         self.scom.commit_beam(self.init_position, 0)
00061
00062

```

7.7 src/communication/ServerComm.py File Reference

Implementação da Comunicação com Servidor.

Classes

- class [ServerComm.ServerComm](#)
Responsável pela comunicação com servidor.

Namespaces

- namespace [ServerComm](#)

7.7.1 Detailed Description

Implementação da Comunicação com Servidor.

Definition in file [ServerComm.py](#).

7.8 ServerComm.py

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

```

00001 """
00002 @file ServerComm.py
00003 @brief Implementação da Comunicação com Servidor
00004 """
00005 import socket
00006 from time import sleep
00007 from term.Printing import Printing
00008 from select import select
00009
00010 class ServerComm:
00011     """
00012     @brief Responsável pela comunicação com servidor.
00013     """
00014
00015     def __init__(self, creation_options: list[list[str]], environment, other_players: list):
00016         """
00017         @brief Construtor da classe, inicializando buffers e a conexão de cada agente com servidor.
00018         @param creation_options Lista de parâmetros de criação, self ainda não foi incluído na lista.
00019         @param environment
00020         @param other_players
00021         """
00022
00023         # Características da comunicação
00024         self.buffer_size = 4096 # Posteriormente, devemos analisar se realmente vale a pena ter um
buffer com este comprimento
00025         self.buffer = bytearray(self.buffer_size)
00026         self.socket = socket.socket(
00027             socket.AF_INET,
00028             socket.SOCK_STREAM # TCP
00029         )
00030         self.socket.settimeout(2)
00031
00032         # Características alheias
00033         self.message_queue = []
00034         self.unum = creation_options[4][1]
00035         self.env = environment
00036
00037         # Fazemos a conexão com servidor
00038         Printing.print_message(f"Tentando conexão do jogador {self.unum}", "info")
00039         while True:
00040             try:
00041
00042                 self.socket.connect(

```



```

00043         (
00044             creation_options[0][1], # Host
00045             creation_options[1][1] # Porta de Agentes
00046         )
00047     )
00048     break
00049 except ConnectionRefusedError:
00050     sleep(1)
00051     Printing.print_message(".")
00052
00053 Printing.print_message("\tAgente Conectado!\n", "info")
00054
00055 # Fazemos o pedido de criação de robô
00056 self.send_immediate(
00057     f"(scene rsg/agent/nao/nao_hetero.rsg {creation_options[5][1]})".encode()
00058 )
00059 self.__receive_async(other_players)
00060 self.send_immediate(
00061     f"(init (unum {self.unum}) (teamname {creation_options[3][1]}))".encode()
00062 )
00063 self.__receive_async(other_players)
00064
00065 # Aqui podem ser realizados testes de execução de quaisquer funções do ServerComm
00066
00067 for _ in range(3):
00068     self.send_immediate(b'(syn)')
00069     for p in other_players:
00070         p.scom.send_immediate(b'(syn)')
00071     for p in other_players:
00072         p.scom.receive()
00073     self.receive()
00074
00075
00076 # self.close()
00077
00078 # Métodos Mínimos da Classe de Comunicação com servidor
00079 def send_immediate(self, message: bytes) -> None:
00080     """
00081     @brief Envia uma mensagem instantânea ao servidor, verificando se a conexão continua ativa
00082     @param message String em forma de bytes para ser transmitida
00083     @details
00084     Coloca-se na frente uma informação de tamanho da mensagem dentro de 4 bytes.
00085     """
00086
00087     try:
00088         self.socket.send(
00089             len(message).to_bytes(4, byteorder="big") + message
00090         )
00091     except BrokenPipeError:
00092         Printing.print_message("Error: socket foi fechado por rcssserver3d", "error")
00093
00094 def receive(self) -> None:
00095     """
00096     @brief Receberá informações diretamente do servidor, fazendo todas as verificações
00097     necessárias.
00098     """
00099     msg_size = None
00100     while True:
00101         try:
00102             # Verificamos se há 4 bytes no cabeçalho e nos preparamos para ler.
00103             if self.socket.recv_into(
00104                 self.buffer, nbytes=4
00105             ) != 4:
00106                 raise ConnectionResetError
00107
00108             # Lemos o comprimento total da mensagem
00109             msg_size = int.from_bytes(
00110                 self.buffer[:4], # Garantimos leitura de apenas 4 bytes
00111                 byteorder="big", # ordem de significativo
00112                 signed=False # se tem sinal
00113             )
00114
00115             # Lemos o restante da mensagem
00116             if(
00117                 self.socket.recv_into(
00118                     self.buffer,
00119                     nbytes=msg_size
00120                 )
00121             ) != msg_size:
00122                 raise ConnectionResetError
00123
00124         except ConnectionResetError:
00125             Printing.print_message("\nError: socket foi fechado pelo rcssserver3d.", "error")
00126             exit()
00127
00128     except TimeoutError:

```

```

00129         pass
00130
00131     if len(
00132         select( # Monitora sockets/arquivos para I/O
00133             [self.socket], # Lista de sockets/arquivos para verificar leitura
00134             [], # Lista vazia para escrita
00135             [], # Lista vazia para exceções
00136             0.0 # timeout zero (não bloqueante)
00137         )[0] # Pegamos o primeiro socket para leitura
00138     ) == 0: # Logo, não há dados disponíveis para leitura
00139         break
00140
00141     # Como há algo para ser lido, devemos aplicar o parser
00142     self.env.update_from_server(self.buffer[:msg_size])
00143
00144     def __receive_async(self, other_players: list) -> None:
00145         """
00146         @brief Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de
00147         execução
00148         @details
00149         Essa função foi criada com o único propósito de impedir que a espera por resposta
00150         do servidor interrompa o fluxo de execução. Não deve ser executada posteriormente.
00151         @param other_players Lista de jogadores de mesmo time presentes na partida
00152         """
00153
00154         # Caso não haja ninguém além dele
00155         if not other_players:
00156             # Sem isso, um loop infinito existiria
00157             return self.receive()
00158
00159         # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00160         self.socket.setblocking(False)
00161
00162         while True:
00163             try:
00164                 Printing.print_message(".")
00165                 self.receive()
00166                 break
00167             except BlockingIOError:
00168                 pass
00169
00170         # Força que todos estejam em condições
00171         for p in other_players:
00172             p.scom.send_immediate(b"(syn)")
00173
00174         for p in other_players:
00175             p.scom.receive()
00176
00177         # Voltamos ao padrão
00178         self.socket.setblocking(True)
00179         return None
00180
00181     def commit(self, message: bytes) -> None:
00182         """
00183         @brief Responsável por adicionar uma nova mensagem à fila de mensagens
00184         @param message String em bytes a ser adicionada à fila
00185         """
00186         assert isinstance(message, bytes), "Mensagem deve estar em bytes"
00187         self.message_queue.append(message)
00188
00189     def close(self) -> None:
00190         """
00191         @brief Responsável por fazer o encerramento dos canais de comunicação
00192         """
00193
00194         self.socket.close()
00195
00196     def send(self) -> None:
00197         """
00198         @brief Enviará ao servidor todas as mensagens commitadas.
00199         """
00200         if len(
00201             select(
00202                 [self.socket],
00203                 [],
00204                 [],
00205                 0.0
00206             )[0]
00207         ) == 0:
00208             # Se não há nenhum socket para ler neste momento, enviarei
00209             self.message_queue.append(b"(syn)")
00210             self.send_immediate(b"".join(self.message_queue))
00211         else:
00212             Printing.print_message("\nHavia sockets de leitura disponíveis enquanto tentava enviar
00213             fila de mensagens commitadas.", "warning")

```

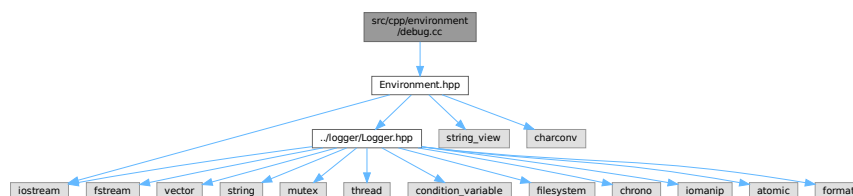
```

00214         self.message_queue.clear() # Limpamos buffer
00215
00216     def clear_queue(self) -> None:
00217         """
00218         @brief Limpará a fila de commits.
00219         """
00220         self.message_queue.clear() # Assim usamos o mesmo ponteiro
00221
00222     # Métodos Derivados
00223     def commit_beam(self, vector_position2d: list, rotation: float):
00224         """
00225         @brief Comando de beam oficial do agente
00226         @param vector_position2d Sequência de dois valores, x e y finais do agente
00227         @param rotation Valor de rotação a ser dado ao robô
00228         """
00229         assert len(vector_position2d) == 2, "O beam oficial permite apenas posições 2D."
00230         self.commit(
00231             f"(beam {vector_position2d[0]} {vector_position2d[1]} {rotation})".encode()
00232         )
00233
00234
00235
00236
00237
00238
00239
00240
00241
00242
00243
00244

```

7.9 src/cpp/environment/debug.cc File Reference

#include "Environment.hpp"
 Include dependency graph for debug.cc:



Functions

- int `main` ()

Variables

- const char * `example` = "(time (now 10.06))(GS (sl 3) (sr 2) (t 5.12) (pm BeforeKickOff))(GYR (n torso) (rt 0.01 -0.00 0.00))(ACC (n torso) (a -0.00 -0.00 0.01))(HJ (n hj1) (ax 0.00))(HJ (n hj2) (ax -0.00))(See (P (team RobolME) (id 1) (rlowerarm (pol 0.18 -35.30 -22.17)) (llowerarm (pol 0.18 36.49 -21.66))) (G2R (pol 30.92 -19.31 0.55)) (G1R (pol 30.30 -15.73 0.47)) (F1R (pol 29.27 1.62 -1.01)) (F2R (pol 34.87 -33.26 -0.82)) (B (pol 16.91 -32.71 -1.64)) (L (pol 23.88 -53.55 -1.53) (pol 14.22 3.30 -2.23)) (L (pol 34.95 -33.18 -0.98) (pol 29.18 1.37 -1.25)) (L (pol 29.20 1.45 -1.09) (pol 1.07 59.96 -29.70)) (L (pol 34.98 -33.31 -0.90) (pol 22.18 -60.01 -1.25)) (L (pol 28.07 -12.48 -0.97) (pol 29.94 -23.73 -1.00)) (L (pol 28.07 -12.88 -1.02) (pol 29.83 -11.92 -1.07)) (L (pol 29.99 -23.90 -1.00) (pol 31.66 -22.86 -0.96)) (L (pol 18.62 -29.50 -1.68) (pol 17.73 -26.93 -1.4776)) (L (pol 17.76 -26.80 -1.58) (pol 16.53 -26.27 -1.95)) (L (pol 16.52 -26.24 -1.94) (pol 15.44 -28.34 -2.03)) (L (pol 15.42 -28.55 -1.86) (pol 14.92 -32.55 -1.98)) (L (pol 14.90 -32.54 -2.25) (pol 15.26 -37.08 -1.89)) (L

- ```
(pol 15.28 -37.21 -2.06) (pol 16.31 -39.67 -1.78)) (L (pol 16.28 -39.55 -1.64) (pol 17.54 -39.17 -1.67)) (L (pol 17.55 -39.31 -1.67) (pol 18.51 -36.89 -1.61)) (L (pol 18.55 -36.88 -1.69) (pol 18.93 -33.46 -1.78)) (L (pol 18.93 -33.32 -1.51) (pol 18.64 -29.59 -1.54)))(HJ (n raj1) (ax 0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax 0.00))(HJ (n laj2) (ax -0.00))(HJ (n laj3) (ax 0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.00))(HJ (n rlj2) (ax -0.00))(HJ (n rlj3) (ax -0.00))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax -0.00))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax 0.00))(HJ (n llj2) (ax 0.00))(HJ (n llj3) (ax -0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax -0.00))(HJ (n llj6) (ax 0.00))"
```
- int [size](#) = 1815
  - const char \* [example1](#) = "(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso)(rt 0.24 -0.05 0.02))(ACC (n torso) (a -0.01 0.05 9.80))(HJ (n hj1)(ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R (pol 20.11 -18.92 0.84))(G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73 -33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L (pol 22.78 -33.20 -1.23)(pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97)(pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64)(pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71) (pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L (pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax -0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax 0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17 22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax 0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax 0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63))(HJ (n llj6) (ax 0.00))"
  - int [size1](#) = 1808

## 7.9.1 Function Documentation

### 7.9.1.1 main()

```
int main ()
```

Definition at line 10 of file [debug.cc](#).

## 7.9.2 Variable Documentation

### 7.9.2.1 example

```
const char* example = "(time (now 10.06))(GS (sl 3) (sr 2) (t 5.12) (pm BeforeKickOff))(GYR (n torso) (rt 0.01 -0.00 0.00))(ACC (n torso) (a -0.00 -0.00 0.01))(HJ (n hj1) (ax 0.00))(HJ (n hj2) (ax -0.00))(See (P (team RoboIME) (id 1) (rlowerarm (pol 0.18 -35.30 -22.17)) (llowerarm (pol 0.18 36.49 -21.66))) (G2R (pol 30.92 -19.31 0.55)) (G1R (pol 30.30 -15.73 0.47)) (F1R (pol 29.27 1.62 -1.01)) (F2R (pol 34.87 -33.26 -0.82)) (B (pol 16.91 -32.71 -1.64)) (L (pol 23.88 -53.55 -1.53) (pol 14.22 3.30 -2.23)) (L (pol 34.95 -33.18 -0.98) (pol 29.18 1.37 -1.↵25)) (L (pol 29.20 1.45 -1.09) (pol 1.07 59.96 -29.70)) (L (pol 34.98 -33.31 -0.90) (pol 22.↵18 -60.01 -1.25)) (L (pol 28.07 -12.48 -0.97) (pol 29.94 -23.73 -1.00)) (L (pol 28.07 -12.88 -1.02) (pol 29.83 -11.92 -1.07)) (L (pol 29.99 -23.90 -1.00) (pol 31.66 -22.86 -0.96)) (L (pol 18.62 -29.50 -1.68) (pol 17.73 -26.93 -1.76)) (L (pol 17.76 -26.80 -1.58) (pol 16.53 -26.↵27 -1.95)) (L (pol 16.52 -26.24 -1.94) (pol 15.44 -28.34 -2.03)) (L (pol 15.42 -28.55 -1.86) (pol 14.92 -32.55 -1.98)) (L (pol 14.90 -32.54 -2.25) (pol 15.26 -37.08 -1.89)) (L (pol 15.28 -37.21 -2.06) (pol 16.31 -39.67 -1.78)) (L (pol 16.28 -39.55 -1.64) (pol 17.54 -39.17 -1.67))
```

```
(L (pol 17.55 -39.31 -1.67) (pol 18.51 -36.89 -1.61)) (L (pol 18.55 -36.88 -1.69) (pol 18.↵
93 -33.46 -1.78)) (L (pol 18.93 -33.32 -1.51) (pol 18.64 -29.59 -1.54)) (HJ (n raj1) (ax 0.↵
00)) (HJ (n raj2) (ax 0.00)) (HJ (n raj3) (ax 0.00)) (HJ (n raj4) (ax 0.00)) (HJ (n laj1) (ax 0.↵
00)) (HJ (n laj2) (ax -0.00)) (HJ (n laj3) (ax 0.00)) (HJ (n laj4) (ax -0.00)) (HJ (n rlj1) (ax
0.00)) (HJ (n rlj2) (ax -0.00)) (HJ (n rlj3) (ax -0.00)) (HJ (n rlj4) (ax -0.00)) (HJ (n rlj5) (ax
-0.00)) (HJ (n rlj6) (ax -0.00)) (HJ (n llj1) (ax 0.00)) (HJ (n llj2) (ax 0.00)) (HJ (n llj3) (ax
-0.00)) (HJ (n llj4) (ax -0.00)) (HJ (n llj5) (ax -0.00)) (HJ (n llj6) (ax 0.00))"
```

Definition at line 3 of file [debug.cc](#).

### 7.9.2.2 example1

```
const char* example1 = "(time (now 104.87)) (GS (t 0.00) (pm BeforeKickOff)) (GYR (n torso) (rt
0.24 -0.05 0.02)) (ACC (n torso) (a -0.01 0.05 9.80)) (HJ (n hj1) (ax -0.00)) (HJ (n hj2) (ax
-0.00)) (See (G2R (pol 20.11 -18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58
-1.54)) (F2R (pol 22.73 -33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.↵
03) (pol 8.67 10.24 -3.34)) (L (pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.↵
08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L (pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79))
(L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38 -1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.↵
35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97) (pol 20.43 -21.36 -1.45)) (L (pol 11.68 -28.↵
23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol 9.84 -22.59 -3.02)) (L
(pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34) (pol 8.35 -32.↵
24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71) (pol
9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.↵
28 -2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L
(pol 11.94 -33.31 -2.52) (pol 11.70 -28.03 -2.56))) (HJ (n raj1) (ax -0.00)) (HJ (n raj2) (ax
0.00)) (HJ (n raj3) (ax 0.00)) (HJ (n raj4) (ax 0.00)) (HJ (n laj1) (ax -0.01)) (HJ (n laj2) (ax
0.00)) (HJ (n laj3) (ax -0.00)) (HJ (n laj4) (ax -0.00)) (HJ (n rlj1) (ax 0.01)) (HJ (n rlj2) (ax
0.00)) (HJ (n rlj3) (ax 0.01)) (HJ (n rlj4) (ax -0.00)) (HJ (n rlj5) (ax 0.00)) (FRP (n rf) (c
-0.02 -0.00 -0.02) (f -0.02 -0.17 22.52)) (HJ (n rlj6) (ax -0.00)) (HJ (n llj1) (ax -0.01)) (HJ
(n llj2) (ax 0.01)) (HJ (n llj3) (ax 0.00)) (HJ (n llj4) (ax -0.00)) (HJ (n llj5) (ax 0.00)) (FRP
(n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20 22.63)) (HJ (n llj6) (ax 0.00))"
```

Definition at line 6 of file [debug.cc](#).

### 7.9.2.3 size

```
int size = 1815
```

Definition at line 4 of file [debug.cc](#).

### 7.9.2.4 size1

```
int size1 = 1808
```

Definition at line 7 of file [debug.cc](#).

## 7.10 debug.cc

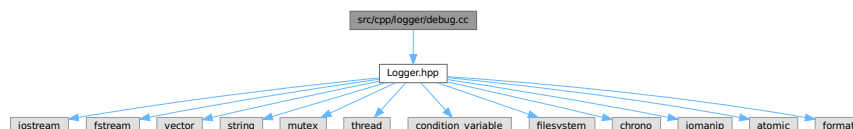
[Go to the documentation of this file.](#)

```
00001 #include "Environment.hpp"
00002
00003 const char* example = "(time (now 10.06))(GS (sl 3) (sr 2) (t 5.12) (pm BeforeKickOff))(GYR (n torso)
(rt 0.01 -0.00 0.00))(ACC (n torso) (a -0.00 -0.00 0.01))(HJ (n hj1) (ax 0.00))(HJ (n hj2) (ax
-0.00))(See (P (team RoboIME) (id 1) (rlowerarm (pol 0.18 -35.30 -22.17)) (llowerarm (pol 0.18 36.49
-21.66))) (G2R (pol 30.92 -19.31 0.55)) (G1R (pol 30.30 -15.73 0.47)) (F1R (pol 29.27 1.62 -1.01))
(F2R (pol 34.87 -33.26 -0.82)) (B (pol 16.91 -32.71 -1.64)) (L (pol 23.88 -53.55 -1.53) (pol 14.22
3.30 -2.23)) (L (pol 34.95 -33.18 -0.98) (pol 29.18 1.37 -1.25)) (L (pol 29.20 1.45 -1.09) (pol 1.07
59.96 -29.70)) (L (pol 34.98 -33.31 -0.90) (pol 22.18 -60.01 -1.25)) (L (pol 28.07 -12.48 -0.97) (pol
29.94 -23.73 -1.00)) (L (pol 28.07 -12.88 -1.02) (pol 29.83 -11.92 -1.07)) (L (pol 29.99 -23.90 -1.00)
(pol 31.66 -22.86 -0.96)) (L (pol 18.62 -29.50 -1.68) (pol 17.73 -26.93 -1.76)) (L (pol 17.76 -26.80
-1.58) (pol 16.53 -26.27 -1.95)) (L (pol 16.52 -26.24 -1.94) (pol 15.44 -28.34 -2.03)) (L (pol 15.42
-28.55 -1.86) (pol 14.92 -32.55 -1.98)) (L (pol 14.90 -32.54 -2.25) (pol 15.26 -37.08 -1.89)) (L (pol
15.28 -37.21 -2.06) (pol 16.31 -39.67 -1.78)) (L (pol 16.28 -39.55 -1.64) (pol 17.54 -39.17 -1.67)) (L
(pol 17.55 -39.31 -1.67) (pol 18.51 -36.89 -1.61)) (L (pol 18.55 -36.88 -1.69) (pol 18.93 -33.46
-1.78)) (L (pol 18.93 -33.32 -1.51) (pol 18.64 -29.59 -1.54)))(HJ (n raj1) (ax 0.00))(HJ (n raj2) (ax
0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax 0.00))(HJ (n laj2) (ax -0.00))(HJ
(n laj3) (ax 0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.00))(HJ (n rlj2) (ax -0.00))(HJ (n rlj3)
(ax -0.00))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax -0.00))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax
0.00))(HJ (n llj2) (ax 0.00))(HJ (n llj3) (ax -0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax
-0.00))(HJ (n llj6) (ax 0.00))";
00004 int size = 1815;
00005
00006 const char* example1 = "(time (now 104.87))(GS (t 0.00) (pm BeforeKickOff))(GYR (n torso)(rt 0.24
-0.05 0.02))(ACC (n torso) (a -0.01 0.05 9.80))(HJ (n hj1) (ax -0.00))(HJ (n hj2) (ax -0.00))(See (G2R
(pol 20.11 -18.92 0.84)) (G1R (pol 19.53 -13.04 0.90)) (F1R (pol 19.08 4.58 -1.54)) (F2R (pol 22.73
-33.49 -1.47)) (B (pol 10.12 -33.09 -2.94)) (L (pol 15.13 -55.78 -2.03) (pol 8.67 10.24 -3.34)) (L
(pol 22.78 -33.20 -1.23) (pol 19.05 4.32 -1.76)) (L (pol 19.08 4.57 -1.55) (pol 1.81 60.14 -17.11)) (L
(pol 22.77 -33.23 -1.26) (pol 14.49 -59.60 -1.79)) (L (pol 17.56 -11.77 -1.83) (pol 18.76 -23.38
-1.60)) (L (pol 17.58 -11.67 -1.74) (pol 19.35 -10.53 -1.53)) (L (pol 18.71 -23.82 -1.97) (pol 20.43
-21.36 -1.45)) (L (pol 11.68 -28.23 -2.73) (pol 10.93 -23.90 -2.69)) (L (pol 10.91 -24.22 -2.95) (pol
9.84 -22.59 -3.02)) (L (pol 9.84 -22.64 -3.06) (pol 8.81 -25.74 -3.68)) (L (pol 8.83 -25.33 -3.34)
(pol 8.35 -32.24 -3.68)) (L (pol 8.35 -32.20 -3.64) (pol 8.69 -39.32 -3.48)) (L (pol 8.68 -39.59 -3.71)
(pol 9.63 -43.18 -3.37)) (L (pol 9.65 -42.85 -3.10) (pol 10.75 -42.17 -2.80)) (L (pol 10.75 -42.28
-2.89) (pol 11.61 -38.36 -2.50)) (L (pol 11.62 -38.15 -2.33) (pol 11.94 -33.38 -2.58)) (L (pol 11.94
-33.31 -2.52) (pol 11.70 -28.03 -2.56)))(HJ (n raj1) (ax -0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3)
(ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax -0.01))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax
-0.00))(HJ (n laj4) (ax -0.00))(HJ (n rlj1) (ax 0.01))(HJ (n rlj2) (ax 0.00))(HJ (n rlj3) (ax
0.01))(HJ (n rlj4) (ax -0.00))(HJ (n rlj5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17
22.52))(HJ (n rlj6) (ax -0.00))(HJ (n llj1) (ax -0.01))(HJ (n llj2) (ax 0.01))(HJ (n llj3) (ax
0.00))(HJ (n llj4) (ax -0.00))(HJ (n llj5) (ax 0.00))(FRP (n lf) (c 0.02 -0.01 -0.01) (f -0.08 -0.20
22.63))(HJ (n llj6) (ax 0.00))";
00007 int size1 = 1808;
00008
00009 int
00010 main(){
00011
00012 std::string_view message_from_server(example, size);
00013 Environment ex = Environment(Logger.get());
00014 ex.update_from_server(message_from_server);
00015
00016 return 0;
00017 }
```

## 7.11 src/cpp/logger/debug.cc File Reference

```
#include "Logger.hpp"
```

Include dependency graph for debug.cc:



### Functions

- void [tarefaPesada](#) (int id)
- int [main](#) ()

## 7.11.1 Function Documentation

### 7.11.1.1 main()

```
int main ()
```

Definition at line 9 of file [debug.cc](#).

### 7.11.1.2 tarefaPesada()

```
void tarefaPesada (
 int id)
```

Definition at line 3 of file [debug.cc](#).

## 7.12 debug.cc

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

```
00001 #include "Logger.hpp"
00002
00003 void tarefaPesada(int id) {
00004 for (int i = 0; i < 1000; ++i) {
00005 Logger::get().info("Thread " + std::to_string(id) + " msg " + std::to_string(i));
00006 }
00007 }
00008
00009 int main() {
00010
00011 /* --- Testar Assincronicamente --- */
00012
00013 auto start = std::chrono::high_resolution_clock::now();
00014
00015 std::vector<std::thread> threads;
00016 for (int i = 0; i < 10; ++i) { // 10 Threads
00017 threads.emplace_back(tarefaPesada, i);
00018 }
00019
00020 for (auto& t : threads) t.join();
00021
00022 auto end = std::chrono::high_resolution_clock::now();
00023 std::chrono::duration<double> diff = end - start;
00024
00025 std::cout << "10.000 logs escritos em: " << diff.count() << " s\n";
00026
00027 /* --- Testar Sincronicamente --- */
00028 // std::cout << "Iniciando teste C++ (Single Thread / 10.000 logs)...\n";
00029 //
00030 // // Ponto de início da medição
00031 // auto start = std::chrono::high_resolution_clock::now();
00032 //
00033 // // Loop sequencial na thread principal
00034 // for (int i = 0; i < 10000; ++i) {
00035 // Logger::get().info("SingleThread msg " + std::to_string(i));
00036 // }
00037 //
00038 // // Ponto final da medição (Tempo que a thread principal ficou ocupada)
00039 // auto end = std::chrono::high_resolution_clock::now();
00040 // std::chrono::duration<double> diff = end - start;
00041 //
00042 // std::cout << "Tempo de execucao (Main Thread): " << diff.count() << " segundos.\n";
00043
00044 return 0;
00045 }
00046
00047 /*
00048 Código Python para eventual comparação:
00049
00050 -----
00051 import threading
00052 import time
```

```

00053 from pathlib import Path
00054 from datetime import datetime
00055 import random
00056 from string import ascii_uppercase
00057
00058 class Logger():
00059 _folder = None
00060
00061 def __init__(self, is_enabled: bool, topic: str) -> None:
00062 self.no_of_entries = 0
00063 self.enabled = is_enabled
00064 self.topic = topic
00065
00066 def write(self, msg: str, timestamp: bool = True, step: int = None) -> None:
00067 """
00068 Write `msg` to file named `self.topic`
00069 """
00070 if not self.enabled: return
00071
00072 # The log folder is only created if needed
00073 if Logger._folder is None:
00074 rnd = "".join(
00075 random.choices(ascii_uppercase, k=6)) # Useful if multiple processes are running in
parallel
00076 Logger._folder = "./logs_python/" + datetime.now().strftime("%Y-%m-%d_%H.%M.%S__") + rnd +
"/"
00077 print("\nLogger Info: see", Logger._folder)
00078 Path(Logger._folder).mkdir(parents=True, exist_ok=True)
00079
00080 self.no_of_entries += 1
00081
00082 # O GARGALO ESTÁ AQUI: Abrir e fechar arquivo a cada linha
00083 with open(Logger._folder + self.topic + ".log", 'a') as f:
00084 prefix = ""
00085 write_step = step is not None
00086 if timestamp or write_step:
00087 prefix = "{"
00088 if timestamp:
00089 prefix += datetime.now().strftime("%a %H:%M:%S")
00090 if write_step: prefix += " "
00091 if write_step:
00092 prefix += f'Step:{step}'
00093 prefix += " } "
00094 f.write(prefix + msg + "\n")
00095
00096 def tarefa_pesada(logger_instance, thread_id):
00097 """
00098 Simula o workerThread do C++:
00099 Envia 1000 mensagens para o log.
00100 """
00101 for i in range(1000):
00102 # Formatando a mensagem igual ao exemplo C++
00103 logger_instance.write(f"Thread {thread_id} msg {i}")
00104
00105
00106 def main():
00107 # --- Testar Assincronicamente ---
00108 # print("Iniciando teste de performance Python...")
00109 #
00110 # # 1. Instancia o Logger
00111 # logger = Logger(is_enabled=True, topic="performance_test")
00112 #
00113 # start_time = time.time()
00114 #
00115 # threads = []
00116 # num_threads = 10
00117 #
00118 # # 2. Cria e inicia as threads
00119 # for i in range(num_threads):
00120 # t = threading.Thread(target=tarefa_pesada, args=(logger, i))
00121 # threads.append(t)
00122 # t.start()
00123 #
00124 # # 3. Aguarda todas as threads terminarem (join)
00125 # for t in threads:
00126 # t.join()
00127 #
00128 # end_time = time.time()
00129 # duration = end_time - start_time
00130 #
00131 # print(f"\nProcessamento finalizado.")
00132 # print(f"Total de logs: {num_threads * 1000}")
00133 # print(f"Tempo total: {duration:.4f} segundos")
00134
00135 # --- Testar Sincronicamente
00136 print("Iniciando teste Python (Single Thread / 10.000 logs)...")
00137

```



```

00138 # Instancia
00139 logger = Logger(is_enabled=True, topic="single_thread_test")
00140
00141 # Ponto de início da medição
00142 start_time = time.time()
00143
00144 # Loop sequencial na thread principal
00145 for i in range(10000):
00146 logger.write(f"SingleThread msg {i}")
00147
00148 # Ponto final da medição
00149 end_time = time.time()
00150 duration = end_time - start_time
00151
00152 print(f"Tempo de execucao (Main Thread): {duration:.4f} segundos.")
00153
00154
00155 if __name__ == "__main__":
00156 main()
00157 */

```

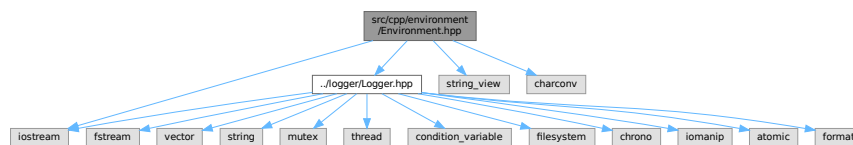
## 7.13 src/cpp/environment/Environment.hpp File Reference

```

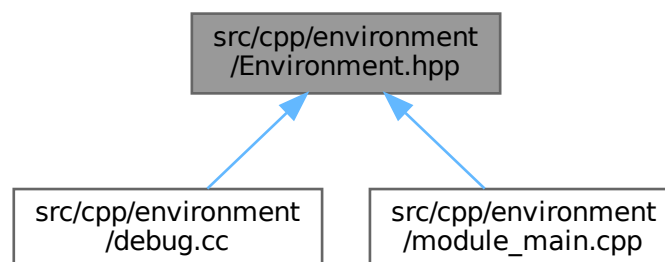
#include "../logger/Logger.hpp"
#include <iostream>
#include <string_view>
#include <charconv>

```

Include dependency graph for Environment.hpp:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Environment](#)  
*Responsável por representar o ambiente externo ao robô*
- class [Environment::Parsing](#)  
*Responsável por prover ferramentas de auxílio de parsing.*

## Macros

- `#define True true`  
    `< std::from_chars`
- `#define False false`

## 7.13.1 Macro Definition Documentation

### 7.13.1.1 False

```
#define False false
```

Definition at line 9 of file [Environment.hpp](#).

### 7.13.1.2 True

```
#define True true
```

```
< std::from_chars
```

Definition at line 8 of file [Environment.hpp](#).

## 7.14 Environment.hpp

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

```
00001 #pragma once
00002
00003 #include "../logger/Logger.hpp"
00004 #include <iostream>
00005 #include <string_view>
00006 #include <charconv>
00007
00008 #define True true
00009 #define False false
00010
00017 class Environment {
00018 public:
00019
00020 Logger& logger;
00021
00025 Environment (
00026 Logger& logger
00027) : logger(logger) {}
00028
00029 /* Atributos Públicos de Ambiente */
00030 float time_server;
00031 float time_match;
00032 int goals_scored;
00033 int goals_conceded;
00034 int unum;
00035 bool is_left;
00036
00037 enum class PlayMode : int {
00038 // Ao nosso favor
00039 OUR_KICKOFF = 0,
00040 OUR_KICK_IN = 1,
00041 OUR_CORNER_KICK = 2,
00042 OUR_GOAL_KICK = 3,
00043 OUR_FREE_KICK = 4,
00044 OUR_PASS = 5,
00045 OUR_DIR_FREE_KICK = 6,
00046 OUR_GOAL = 7,
00047 OUR_OFFSIDE = 8,
00048
00049 // Ao favor deles
```

```

00050 THEIR_KICKOFF = 9,
00051 THEIR_KICK_IN = 10,
00052 THEIR_CORNER_KICK = 11,
00053 THEIR_GOAL_KICK = 12,
00054 THEIR_FREE_KICK = 13,
00055 THEIR_PASS = 14,
00056 THEIR_DIR_FREE_KICK = 15,
00057 THEIR_GOAL = 16,
00058 THEIR_OFFSIDE = 17,
00059
00060 // Neutros
00061 BEFORE_KICKOFF = 18,
00062 GAME_OVER = 19,
00063 PLAY_ON = 20
00064 };
00065 enum class PlayModeGroup : int {
00066 OUR_KICK = 0,
00067 THEIR_KICK = 1,
00068 ACTIVE_BEAM = 2,
00069 PASSIVE_BEAM = 3,
00070 OTHER = 4
00071 };
00072
00073 /* Métodos Inerentes a Execução da Aplicação */
00074
00075
00076
00077
00078
00079 /* ----- Parser de Mensagem do Servidor ----- */
00080
00081 class Parsing {
00082 private:
00083 const char* buffer = nullptr;
00084 const char* end = nullptr;
00085 Environment* env = nullptr;
00086
00087 public:
00088 /* Métodos Simples de Cursor */
00089
00090 Parsing(
00091 std::string_view& message,
00092 Environment* env
00093) :
00094 buffer(message.data()),
00095 end(message.data() + message.size()),
00096 env(env)
00097 {}
00098
00099 bool
00100 skip_until_char(char caract){
00101 while(*this->buffer != caract){
00102 if(this->buffer > this->end){ return False; }
00103 this->buffer++;
00104 }
00105 this->buffer++;
00106 return True;
00107 }
00108
00109 std::string_view
00110 get_str(){
00111 while(*this->buffer == ' ' || *this->buffer == '(' || *this->buffer == ')'){
00112 this->buffer++; }
00113 const char* value_start = this->buffer;
00114 while(*this->buffer != ' ' && *this->buffer != ')'){ this->buffer++; }
00115 return std::string_view(value_start, size_t(this->buffer - value_start));
00116 }
00117
00118 template<typename T>
00119 bool
00120 get_value(T& out){
00121 const char* value_start = this->buffer;
00122 while(*this->buffer != ' ' && *this->buffer != ')'){ this->buffer++; }
00123 return std::from_chars(value_start, this->buffer, out).ec == std::errc{};
00124 }
00125
00126 bool
00127 advance(int n = 1){ if((this->buffer + n) > this->end){ return False; } this->buffer += n;
00128 return True; }
00129
00130 /* -- Métodos de Parsing -- */
00131
00132 void
00133 parse_time(){
00134 /*
00135 Buffer está aqui.
00136 |

```

```

00169 v
00170 ' (now 10.03)'
00171 */
00172 this->advance(5);
00173 this->get_value(env->time_server);
00174 this->advance();
00175 }
00176
00180 void
00181 parse_gamestate(){
00182 std::string_view lower_tag;
00183 while(True){
00184 lower_tag = this->get_str();
00185
00186 switch(lower_tag[0]){
00187 case 's':
00188 this->get_value((lower_tag[1] == 'l') ? env->goals_scored :
00189 env->goals_conceded);
00190 break;
00191 case 'p':
00192 lower_tag = this->get_str();
00193
00194 std::cout << "Estamos no modo: " << lower_tag << std::endl;
00195 break;
00196 case 't':
00197 if(lower_tag[1] == 'i'){ this->get_value(env->time_match); }
00198 else{
00199 env->is_left = this->get_str()[0] == 'l';
00200 }
00201 break;
00202 case 'u':
00203 this->get_value(env->unum);
00204 break;
00205 default:
00206 env->logger.warn("Flag Desconhecida Encontrada em 'GS': {} \n\t\t\t\t Buffer
00207 Neste momento: {}", lower_tag, buffer);
00208 break;
00209 }
00210 if(*this->buffer == ' '){ break; }
00211 }
00212 }
00213
00214 int
00215 update_from_server(
00216 std::string_view msg
00217){
00218 Parsing cursor(msg, this);
00219 std::string_view upper_tag;
00220 while(True){
00221 if(
00222 !cursor.skip_until_char('(')
00223){ this->print_status(); return 0; }
00224
00225 upper_tag = cursor.get_str();
00226 switch(upper_tag[0]){
00227 case 't':
00228 cursor.parse_time();
00229 break;
00230 case 'G':
00231 if(upper_tag[1] == 'S'){
00232 cursor.parse_gamestate();
00233 }
00234 else if(upper_tag[1] == 'Y'){
00235 }
00236 else{
00237 }
00238 break;
00239 case 'S':
00240 break;
00241 }
00242 }
00243 }

```

```

00263
00264 case 'H':
00265 break;
00266
00267 case 'A':
00268 break;
00269
00270 default:
00271 break;
00272 }
00273 }
00274 }
00275 }
00276
00277 private:
00278
00282 void
00283 print_status() const {
00284 printf("\n== Environment State ==\n");
00285 printf("time_server : %.3f\n", time_server);
00286 printf("time_match : %.3f\n", time_match);
00287 printf("goals_scored : %d\n", goals_scored);
00288 printf("goals_conceded : %d\n", goals_conceded);
00289 }
00290 };
00291
00292
00293
00294

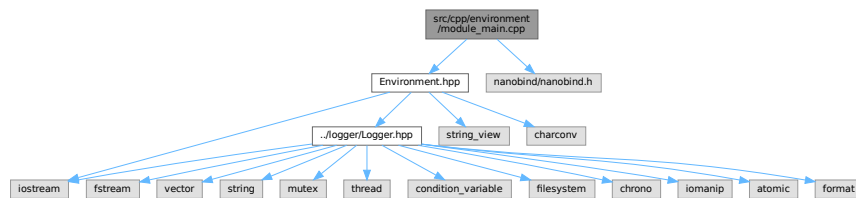
```

## 7.15 src/cpp/environment/module\_main.cpp File Reference

```

#include "Environment.hpp"
#include <nanobind/nanobind.h>
Include dependency graph for module_main.cpp:

```



### Functions

- [NB\\_MODULE](#) (environment, m)

### 7.15.1 Function Documentation

#### 7.15.1.1 NB\_MODULE()

```

NB_MODULE (
 environment ,
 m)

```

Definition at line 6 of file [module\\_main.cpp](#).

## 7.16 module\_main.cpp

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

```

00001 #include "Environment.hpp"
00002 #include <nanobind/nanobind.h>
00003
00004 namespace nb = nanobind;
00005
00006 NB_MODULE(
00007 environment,
00008 m
00009){
00010 nb::class_<Environment>(
00011 m,
00012 "Environment",
00013 "Responsável por representar o ambiente externo ao robô.\n"
00014 "Agrupará todos os métodos de interpretação do mundo."
00015)
00016 .def(
00017 nb::init<Logger&>(),
00018 nb::arg("Logger"),
00019 R"(Classe Logger para imprimirmos possíveis erros.)"
00020)
00021
00022 /* -- Definição de Atributos da Classe -- */
00023 .def_ro("time_server", &Environment::time_server,
00024 "Instante de Tempo do Servidor, útil apenas para sincronização entre agentes")
00025
00026 .def_ro("time_match", &Environment::time_match,
00027 "Instante de Tempo de Partida")
00028
00029 .def_ro("goals_scored", &Environment::goals_scored,
00030 "Nossos Gols, pode ser útil para mudarmos de tática conforme o jogo avança")
00031
00032 .def_ro("goals_conceded", &Environment::goals_conceded,
00033 "Gols adversários, pode ser útil para mudarmos de tática conforme o jogo avança")
00034
00035
00036 /* -- Métodos da Classe -- */
00037 .def(
00038 "update_from_server",
00039 // A função anônima é apenas para convertermos os tipos
00040 [](
00041 Environment &self,
00042 const nb::bytearray& from_server
00043){
00044 return self.update_from_server(std::string_view(reinterpret_cast<const
00045 char*>(from_server.data()), from_server.size()));
00046 },
00047 nb::arg("from_server"),
00048);
00049 }
00050

```

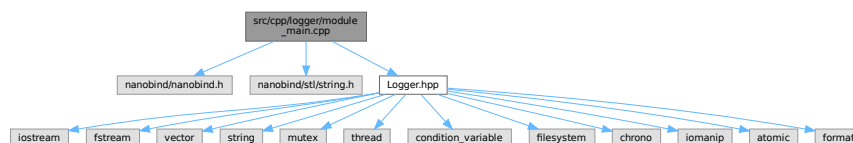
## 7.17 src/cpp/logger/module\_main.cpp File Reference

```

#include <nanobind/nanobind.h>
#include <nanobind/stl/string.h>
#include "Logger.hpp"

```

Include dependency graph for module\_main.cpp:



## Functions

- [NB\\_MODULE](#) (logger, m)  
 < *Necessário para converter std::string <-> str automaticamente*

### 7.17.1 Function Documentation

#### 7.17.1.1 NB\_MODULE()

```
NB_MODULE (
 logger ,
 m)
```

< *Necessário para converter std::string <-> str automaticamente*

Definition at line 7 of file [module\\_main.cpp](#).

## 7.18 module\_main.cpp

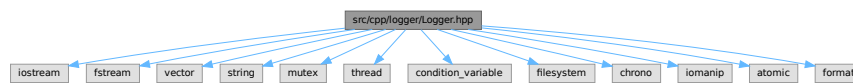
[Go to the documentation of this file.](#)

```
00001 #include <nanobind/nanobind.h>
00002 #include <nanobind/stl/string.h>
00003 #include "Logger.hpp"
00004
00005 namespace nb = nanobind;
00006
00007 NB_MODULE(logger, m) {
00008
00009 // Vinculamos a classe Logger.
00010 // Note que não usamos .def(nb::init<...>) pois o construtor é privado.
00011 nb::class_<Logger>(m, "Logger")
00012 // nb::rv_policy::reference -> Diz ao Python para criar apenas uma referência
00013 // para o objeto estático existente no C++, sem tentar gerenciá-lo ou deletá-lo.
00014 .def_static("get", &Logger::get, nb::rv_policy::reference,
00015 "Acesso à instância única")
00016 .def("info", [](Logger& self, std::string msg) {
00017 self.info(std::move(msg));
00018 },
00019 nb::arg("msg"),
00020 "Adiciona log nível INFO.\n\n"
00021 "Args:\n"
00022 " msg (str): Mensagem a ser imprimida.\n\n"
00023 "Details:\n"
00024 " Recebe por valor para permitir std::move (otimização de r-values).")
00025 .def("warn", [](Logger& self, std::string msg) {
00026 self.warn(std::move(msg));
00027 },
00028 nb::arg("msg"),
00029 "Adiciona log nível WARN.\n\n"
00030 "Args:\n"
00031 " msg (str): Mensagem a ser imprimida.\n\n"
00032 "Details:\n"
00033 " Recebe por valor para permitir std::move (otimização de r-values).")
00034 .def("error", [](Logger& self, std::string msg) {
00035 self.error(std::move(msg));
00036 },
00037 nb::arg("msg"),
00038 "Adiciona log nível ERROR.\n\n"
00039 "Args:\n"
00040 " msg (str): Mensagem a ser imprimida.\n\n"
00041 "Details:\n"
00042 " Recebe por valor para permitir std::move (otimização de r-values).");
00043 }
00044
```

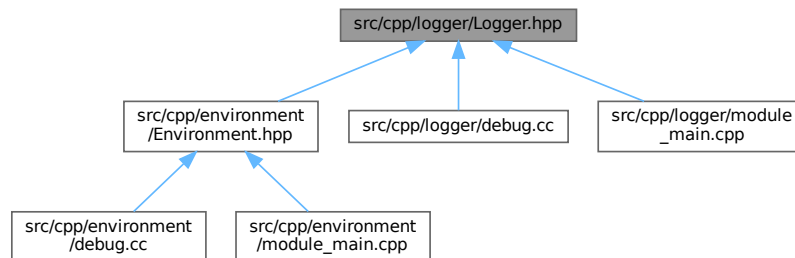
## 7.19 src/cpp/logger/Logger.hpp File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <mutex>
#include <thread>
#include <condition_variable>
#include <filesystem>
#include <chrono>
#include <iomanip>
#include <atomic>
#include <format>
```

Include dependency graph for Logger.hpp:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Logger](#)  
*Singleton para logging assíncrono.*

### Macros

- `#define True true`
- `#define False false`

### 7.19.1 Macro Definition Documentation

#### 7.19.1.1 False

```
#define False false
```

Definition at line 19 of file [Logger.hpp](#).



## 7.19.1.2 True

```
#define True true
```

Definition at line 18 of file [Logger.hpp](#).

## 7.20 Logger.hpp

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

```
00001 #pragma once
00002
00003 #include <iostream>
00004 #include <fstream>
00005 #include <vector>
00006 #include <string>
00007 #include <mutex>
00008 #include <thread>
00009 #include <condition_variable>
00010 #include <filesystem>
00011 #include <chrono>
00012 #include <iomanip>
00013 #include <atomic>
00014 #include <format>
00015
00016 namespace fs = std::filesystem;
00017
00018 #define True true
00019 #define False false
00020
00025 class Logger {
00026 public:
00030 static Logger& get(){ static Logger instance; return instance; }
00031
00032 Logger(const Logger&) = delete;
00033 void operator=(const Logger&) = delete;
00034
00040 void
00041 info(std::string msg){ this->log("[INFO] ", std::move(msg)); }
00042
00048 void
00049 warn(std::string msg){ this->log("[WARN] ", std::move(msg)); }
00050
00056 void
00057 error(std::string msg){ this->log("[ERROR] ", std::move(msg)); }
00058
00064 template<typename... Args>
00065 void info(std::format_string<Args...> fmt, Args&&... args) {
00066 // std::format gera a std::string final de forma otimizada.
00067 // std::forward garante que não haja cópias desnecessárias dos argumentos.
00068 this->log("[INFO] ", std::format(fmt, std::forward<Args>(args)...));
00069 }
00070
00074 template<typename... Args>
00075 void warn(std::format_string<Args...> fmt, Args&&... args) {
00076 this->log("[WARN] ", std::format(fmt, std::forward<Args>(args)...));
00077 }
00078
00082 template<typename... Args>
00083 void error(std::format_string<Args...> fmt, Args&&... args) {
00084 this->log("[ERROR] ", std::format(fmt, std::forward<Args>(args)...));
00085 }
00086
00087 private:
00088 // Buffers para técnica de Double Buffering
00089 std::vector<std::string> _current_buffer;
00090 std::vector<std::string> _write_buffer;
00091
00092 std::mutex _mutex;
00093 std::condition_variable _cv;
00094 std::thread _worker;
00095 std::atomic<bool> _is_running;
00096 std::atomic<bool> is_the_first = True;
00097 std::ofstream _file_stream;
00098
00103 Logger() : _is_running(True) {
00104 // Reserva memória prévia para evitar realocações frequentes no vetor
00105 this->_current_buffer.reserve(30);
00106 this->_write_buffer.reserve(30);
```

```

00107 }
00108
00112 ~Logger(){
00113 this->_is_running = false;
00114 this->_cv.notify_one();
00115
00116 if(this->_worker.joinable()){ this->_worker.join(); }
00117 if(this->_file_stream.is_open()){ this->_file_stream.close(); }
00118 }
00119
00124 void
00125 _init_file(){
00126
00127 if(!fs::exists("logs")){ fs::create_directory("logs"); }
00128
00129 auto now = std::chrono::system_clock::now();
00130 auto in_time_t = std::chrono::system_clock::to_time_t(now);
00131
00132 std::stringstream ss;
00133 ss << "logs/" << std::put_time(std::localtime(&in_time_t), "%Y-%m-%d_%H-%M-%S") << ".log";
00134
00135 // std::ios::app não é necessário se o arquivo é único por execução
00136 // mas útil se reiniciarmos o logger no mesmo segundo -> Impossível?
00137 this->_file_stream.open(ss.str(), std::ios::out | std::ios::app);
00138
00139 // Desabilita sincronização automática com stdio para performance
00140 std::ios_base::sync_with_stdio(false);
00141 }
00142
00149 void
00150 log(const char* prefixo, std::string&& msg) {
00151 {
00152 std::lock_guard<std::mutex> lock(this->_mutex);
00153 // Constrói a string final na memória RAM
00154 this->_current_buffer.emplace_back(prefixo + msg);
00155 }
00156
00157 if(this->is_the_first){ this->_init_file();
00158 this->_worker = std::thread(&Logger::_worker_loop, this);
00159 this->is_the_first = false;
00160 }
00161
00162 // Notifica a thread de escrita que há dados
00163 _cv.notify_one();
00164 }
00165
00171 void
00172 _worker_loop() {
00173
00174 while(
00175 _is_running || !_current_buffer.empty()
00176){
00177
00178 std::unique_lock<std::mutex> lock(_mutex);
00179
00180 /*
00181 A thread fica bloqueada pelo sistema operacional, sem consumir CPU.
00182 Pesquise, isso é muito foda.
00183 */
00184 _cv.wait(
00185 lock,
00186 [this]() { return !this->_current_buffer.empty() || !this->_is_running; }
00187);
00188
00189
00190
00191
00192 if(this->_current_buffer.empty() && !this->_is_running){ break; }
00193
00194 // --- A MÁGICA DA PERFORMANCE (SWAP) ---
00195 // Trocamos o vetor cheio pelo vazio instantaneamente.
00196 // O Mutex é liberado log depois disso.
00197 std::swap(this->_current_buffer, this->_write_buffer);
00198 lock.unlock();
00199
00200 if(this->_file_stream.is_open()) {
00201 for(const auto& line : this->_write_buffer){ this->_file_stream << line << "\n"; }
00202 // Flush manual apenas após lote grande
00203 this->_file_stream.flush();
00204 }
00205
00206 // Limpa o buffer de escrita para ser reusado no próximo swap
00207 this->_write_buffer.clear();
00208 }
00209 }
00210 }
00211 };
00212

```

## 7.21 src/run\_full\_team.py File Reference

### Namespaces

- namespace [run\\_full\\_team](#)

### Variables

- [run\\_full\\_team.boot](#) = [Booting\(\)](#)
- list [run\\_full\\_team.players](#) = []
- Agent [run\\_full\\_team.p](#)

## 7.22 run\_full\_team.py

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

```
00001 from term.Booting import Booting
00002 from agent.Agent import Agent
00003 from time import sleep
00004
00005 boot = Booting()
00006
00007 players = []
00008 for i in range(0, 11):
00009 players.append(Agent(boot.options))
00010 boot.options[4][1] += 1
00011
00012 for p in players:
00013 p: Agent
00014 p.beam()
00015 p.scom.send()
00016
00017 for p in players:
00018 p.scom.receive()
00019
00020
00021
00022 sleep(60)
00023
00024 for p in players:
00025 p.scom.close()
```

## 7.23 src/run\_player.py File Reference

### Namespaces

- namespace [run\\_player](#)

### Variables

- [run\\_player.boot](#) = [Booting\(\)](#)
- [run\\_player.p](#) = [Agent\(boot.options\)](#)

## 7.24 run\_player.py

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

```
00001 from term.Bootimg import Bootimg
00002 from agent.Agent import Agent
00003
00004 boot = Bootimg()
00005
00006 p = Agent(boot.options)
00007
00008 while True:
00009 p.scom.send()
00010 p.scom.receive()
```

## 7.25 src/term/Bootimg.py File Reference

Implementação do [Bootimg](#) do time.

### Classes

- class [Bootimg.Bootimg](#)  
*Responsável por inicializar todas as necessidades de execução do time.*

### Namespaces

- namespace [Bootimg](#)

### 7.25.1 Detailed Description

Implementação do [Bootimg](#) do time.

Definition in file [Bootimg.py](#).

## 7.26 Bootimg.py

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

```
00001 """
00002 @file Bootimg.py
00003 @brief Implementação do Bootimg do time
00004 """
00005 import os
00006 import sys
00007 import subprocess
00008 import sysconfig
00009 import nanobind
00010 import threading
00011 import pickle
00012 from time import sleep
00013 from term.Printing import Printing
00014 from pathlib import Path
00015
00016 class Bootimg:
00017 """
00018 @brief Responsável por inicializar todas as necessidades de execução do time
00019 @details
00020 Assume as seguintes responsabilidades:
00021 - Estabelece um arquivo de configurações default caso já não exista um.
00022 """
00023
```

```

00024 CONFIG_PATH = Path(__file__).resolve().parent / "config_team_params.txt"
00025
00026 def __init__(self):
00027 """
00028 @brief Responsável por chamar as inicializações mínimas.
00029 """
00030
00031 self.options = Booting.get_team_params()
00032
00033 if getattr(sys, 'frozen', False):
00034 # Então estamos executando o binário!
00035 # Devemos forçar que o debug seja 0.
00036 self.options[8][1] = '0'
00037 Printing.IF_IN_DEBUG = False
00038 else:
00039 # Note que isso só faz sentido quando não estamos executando o código em binário
00040 # Já que esta execução não conteria os arquivos .hpp, por exemplo.
00041 Booting.cpp_builder()
00042
00043 @staticmethod
00044 def get_team_params() -> list[list[str | int]]:
00045 """
00046 @brief Verifica existência de arquivo de parâmetros de time, caso não exista, usará o default.
00047 @details
00048 Faremos em tupla para permitir uso mínimo de memória.
00049 @return
00050 """
00051
00052 if os.path.exists(Booting.CONFIG_PATH):
00053 with open(
00054 Booting.CONFIG_PATH,
00055 "r"
00056) as file_team_params:
00057 config_team_params: list[list[str | int]] = [
00058 string_tupla.split(",") for string_tupla in
00059 file_team_params.read().split("\n")[:-1]
00060]
00061
00062 for idx in range(0, len(config_team_params)):
00063 # Somente o IP Server e Team Name são palavras
00064 if idx not in {0, 3}:
00065 config_team_params[idx][1] = int(config_team_params[idx][1])
00066
00067 config_team_params = [
00068 ["IP Server", "localhost"],
00069 ["Agent Port", 3100], # Onde nos conectaremos com rcssserver3d
00070 ["Monitor Port", 3200], # Onde nos conectaremos com Roboviz
00071 ["Team Name", "RoboIME"],
00072 ["Uniform Number", 1],
00073 ["Robot Type", 1],
00074 ["Penalty Shootout", 0],
00075 ["MagmaFatProxy", 0],
00076 ["Debug Mode", 1]
00077]
00078
00079 # E criamos o arquivo
00080 with open(
00081 Booting.CONFIG_PATH,
00082 "w+"
00083) as file_team_params:
00084 for doc, value in config_team_params:
00085 file_team_params.write(
00086 f"{doc},{value}\n"
00087)
00088
00089 return config_team_params
00090
00091 @staticmethod
00092 def show_spinner(
00093 running_flag: list[bool]
00094) -> None:
00095 """
00096 @brief Por motivos estéticos, mostrará um spinner enquanto há o carregamento de módulos C++
00097 """
00098
00099 spinner = ['|', '/', '-', '\\']
00100 i = 0
00101 while running_flag[0] and i < 1000:
00102 print(f"{spinner[i % len(spinner)]}", end="", flush=True)
00103 i += 1
00104 sleep(0.5)
00105 print("\b", end="")
00106
00107 @staticmethod

```

```

00110 def cpp_builder() -> None:
00111 """
00112 @brief Responsável por buildar os arquivos .cpp presentes na pasta cpp.
00113 @return Funcionalidades C++ em condições de interoperabilidade.
00114 """
00115
00116 # Vamos verificar quais arquivos .cpp estão disponíveis para buildar
00117 cpp_path = Path(__file__).resolve().parents[1] / "cpp"
00118 cpp_modules = [
00119 module for module in os.listdir(
00120 cpp_path
00121) if os.path.isdir(os.path.join(cpp_path, module))
00122]
00123
00124 if not cpp_modules:
00125 return None # Não há nenhum para construirmos
00126
00127 # Servirá para verificarmos quais binários estão atualizados com a versão
00128 python_cmd = f"python{sys.version_info.major}.{sys.version_info.minor}"
00129
00130 # -- Os includes que serão necessários
00131 nb_root = os.path.dirname(nanobind.__file__)
00132 py_inc = sysconfig.get_path("include") # Python.h
00133 nb_inc = nanobind.include_dir() # nanobind.h
00134 robin_inc = os.path.join(nb_root, "ext", "robin_map", "include") # robin_map.h
00135 nb_src = os.path.join(nb_root, "src", "nb_combined.cpp")
00136 n_proc = str(os.cpu_count())
00137 command_chain = [
00138 "make",
00139 f"-j{n_proc}",
00140 f"PY_INC={py_inc}",
00141 f"NB_INC={nb_inc}",
00142 f"ROBIN_INC={robin_inc}",
00143 f"NB_SRC={nb_src}"
00144]
00145
00146 first = True
00147 for cpp_module in cpp_modules:
00148 cpp_module_path = os.path.join(cpp_path, cpp_module)
00149
00150 # Verificamos se já existe um binário pronto
00151 if os.path.isfile(
00152 os.path.join(
00153 cpp_module_path,
00154 f"{cpp_module}.so"
00155)
00156):
00157 # Caso exista, devemos verificar se ele foi modificado em um limite de tempo
00158 with open(
00159 os.path.join(
00160 cpp_module_path,
00161 f"{cpp_module}.cpp_info"
00162),
00163 "rb"
00164) as f:
00165 info_version = pickle.load(f)
00166
00167 if info_version == python_cmd:
00168 # Considerando que está na mesma versão, ainda devemos verificar modificações
00169
00170 code_mod_time = max(
00171 os.path.getmtime(
00172 os.path.join(
00173 cpp_module_path,
00174 file_in_the_module
00175)
00176) for file_in_the_module in os.listdir(
00177 cpp_module_path
00178) if file_in_the_module.endswith(".cpp") or
00179 file_in_the_module.endswith(".hpp")
00180)
00181
00182 bin_mod_time = os.path.getmtime(os.path.join(cpp_module_path, f"{cpp_module}.so"))
00183
00184 if bin_mod_time + 15 > code_mod_time:
00185 continue
00186
00187 if first:
00188 print("\033[1;7m/* ---- Construção de Funcionalidades C++ ---- */\033[0m")
00189 first = False
00190 msg = f"\033[1;7mConstruindo: \033[32;40m{cpp_module}\033[0m"
00191 print(f"{msg:.<{60}}", end="", flush=True)
00192
00193 processo = subprocess.Popen(
00194 command_chain,
00195 cwd=cpp_module_path,
00196 stdout=subprocess.PIPE,

```

```

00196 stderr=subprocess.PIPE,
00197 text=False
00198)
00199
00200 # Iniciamos thread de spinner
00201 running_flag = [True]
00202 worker = threading.Thread(target=Booting.show_spinner, args=(running_flag,))
00203 worker.start()
00204
00205 output, error = processo.communicate()
00206 return_code = processo.wait()
00207
00208 running_flag[0] = False
00209 worker.join()
00210
00211 if return_code == 0:
00212 print("\033[7m\033[1mSucesso\033[0m")
00213
00214 # Podemos construir um arquivo de fiscalização
00215 with open(
00216 os.path.join(cpp_module_path, f"{cpp_module}.cpp_info"),
00217 "wb"
00218) as f:
00219 # noinspection PyTypeChecker
00220 pickle.dump(python_cmd, f)
00221 else:
00222 Printing.print_message("Abortando", "error")
00223 print(output.decode(), error.decode())
00224 exit()
00225
00226 subprocess.run(
00227 ["make", "clean"],
00228 stdout=subprocess.PIPE,
00229 stderr=subprocess.PIPE,
00230 cwd=cpp_module_path
00231)
00232
00233 return None

```

## 7.27 src/term/Printing.py File Reference

Implementação de Interface no terminal.

### Classes

- class [Printing.Printing](#)  
*Responsável pela comunicação usuário - terminal.*

### Namespaces

- namespace [Printing](#)

### 7.27.1 Detailed Description

Implementação de Interface no terminal.

Definition in file [Printing.py](#).

## 7.28 Printing.py

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

```

00001 """
00002 @file Printing.py
00003 @brief Implementação de Interface no terminal
00004 """
00005 from rich.console import Console, ConsoleRenderable
00006 from rich.table import Table
00007 from rich import box
00008
00009 from select import select
00010 import sys, tty, termios
00011 from typing import Callable
00012
00013 class Printing:
00014 """
00015 @brief Responsável pela comunicação usuário - terminal
00016 """
00017 IF_IN_DEBUG = True
00018 TABLE_COLORS = {
00019 "info": "\033[1;36m",
00020 "warning": "\033[1;33m",
00021 "error": "\033[1;31m"
00022 }
00023 CONSOLE = Console()
00024
00025 @staticmethod
00026 def print_message(message: str, role: str=None) -> None:
00027 """
00028 @brief Apresentará uma mensagem estilizada de forma específica
00029 @param message Mensagem a ser apresentada
00030 @param role String indicando qual o motivo da mensagem
00031 @details
00032 Há uma quantidade específica de roles possíveis:
00033 - info
00034 - warning
00035 - error
00036
00037 Caso nenhuma dessas seja inserida, há a possibilidade de inserir
00038 o comando ASCII de uma vez.
00039 """
00040
00041 if not Printing.IF_IN_DEBUG:
00042 return
00043
00044 if role is None:
00045 print(message, end="", flush=True)
00046 return
00047
00048 if role in Printing.TABLE_COLORS:
00049 print(f"{Printing.TABLE_COLORS[role]}", end="", flush=True)
00050 else:
00051 if role.startswith("\033["):
00052 print(f"{role}", end="", flush=True)
00053 else:
00054 Printing.print_message("Erro: `role` não especificada.", "error")
00055 return
00056
00057 print(message, end="", flush=True)
00058 print("\033[0m", flush=True, end="")
00059
00060 @staticmethod
00061 def print_table(
00062 columns: list[str],
00063 dados: list[list],
00064 # Diversas personalizações
00065 header_style: str = "bold",
00066 row_style: dict[int, str] = None,
00067 width: int = None,
00068 column_styles: dict[str, str] = None,
00069 column_justify: dict[str, str] = None,
00070 column_widths: dict[str, int] = None,
00071 renderable: bool = False
00072) -> None | ConsoleRenderable:
00073 """
00074 @brief Apresentará uma tabela completamente personalizada
00075 @param columns Lista dos nomes das colunas
00076 @param data Lista de listas com os valores de linhas
00077 @details
00078 Assume os seguintes parâmetros de personalização:
00079 columns: Lista de nomes das colunas
00080 data: Lista de listas com dados das linhas
00081 header_style: Estilo do cabeçalho
00082 row_styles: Estilos alternados para linhas

```



```

00083 width: Largura fixa da tabela
00084 column_styles: {nome_coluna: estilo}
00085 column_justify: {nome_coluna: "left"/"center"/"right"}
00086 column_widths: {nome_coluna: largura}
00087 """
00088
00089 row_style = row_style or {}
00090 column_styles = column_styles or {}
00091 column_justify = column_justify or {}
00092 column_widths = column_widths or {}
00093
00094 table = Table(
00095 box=box.ROUNDED,
00096 header_style=header_style,
00097 width=width,
00098 show_lines=True
00099)
00100
00101 for col in columns:
00102 # noinspection PyTypeChecker
00103 table.add_column(
00104 col,
00105 style=column_styles.get(col, ""),
00106 justify=column_justify.get(col, "default"),
00107 width=column_widths.get(col, None)
00108)
00109
00110 for i, row in enumerate(dados):
00111 table.add_row(*[str(item) for item in row], style=row_style.get(i, ""))
00112
00113 return table if renderable else Printing.CONSOLE.print(table)
00114
00115 @staticmethod
00116 def get_input(
00117 bytes_to_be_read: int,
00118 return_type: Callable = str
00119):
00120 """
00121 @brief Função complexa que fará leitura de entrada do usuário
00122 @details
00123 Tome cuidado com a execução dessa função, pois ela é poderosa
00124 @param return_type Tipo de entrada a ser retornado
00125 @param bytes_to_be_read Quantidade de Bytes que serão lidos
00126 @return Entrada do usuário
00127 """
00128
00129 # Obtém o File Descriptor do stdin
00130 fd = sys.stdin.fileno()
00131
00132 # Guarda modo original (echo, buffering, etc) para restaurar depois
00133 old_settings = termios.tcgetattr(fd)
00134
00135 buffer = ""
00136
00137 try:
00138 # - Desativa buffering de linha (não espera Enter)
00139 # - Desativa echo (não mostra teclas na tela)
00140 # - Desativa processamento de caracteres especiais (Ctrl+C, etc)
00141 # - Captura teclas imediatamente
00142 tty.setraw(fd)
00143
00144 while len(buffer) < bytes_to_be_read:
00145 # Verifica se há input disponível (não-bloqueante)
00146 if select([sys.stdin], [], [], 0.5)[0]:
00147 # Adicionamos cada caractere
00148 buffer += sys.stdin.read(1)
00149 if buffer[-1] in {'\r', '\n'}:
00150 break
00151
00152 finally:
00153 # Restaura configurações originais do terminal
00154 # Garante que o terminal volta ao normal mesmo com erros
00155 termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
00156
00157 try:
00158 return return_type(buffer)
00159 except (ValueError, TypeError):
00160 Printing.print_message("Erro de entrada!", "error")
00161 return None
00162
00163
00164
00165
00166
00167
00168
00169

```

## 7.29 src/utils/RobotPositionManager.py File Reference

Implementação de lógica organizadora de posições iniciais de partida.

### Classes

- class [RobotPositionManager.RobotPositionManager](#)  
*Responsável por permitir ao usuário a criação de diversas formações táticas.*

### Namespaces

- namespace [RobotPositionManager](#)

### Variables

- [RobotPositionManager.root](#) = [RobotPositionManager\(\)](#)

### 7.29.1 Detailed Description

Implementação de lógica organizadora de posições iniciais de partida.

Definition in file [RobotPositionManager.py](#).

## 7.30 RobotPositionManager.py

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

```
00001 """
00002 @file RobotPositionManager.py
00003 @brief Implementação de lógica organizadora de posições iniciais de partida.
00004 """
00005 import os
00006 import pickle
00007 import tkinter as tk
00008 from tkinter import ttk, simpledialog, messagebox
00009 from pathlib import Path
00010
00011 class RobotPositionManager(tk.Tk):
00012 """
00013 @brief Responsável por permitir ao usuário a criação de diversas formações táticas.
00014 @details
00015 Focada em diversão e customização, gerencia um binário que é a representação de
00016 dicionário de listas que contém as 11 posições.
00017 Por ter esse objetivo, não faz sentido que haja essa função na lógica geral dos agentes.
00018 """
00019
00020 CONFIG_POSITION_PATH = Path(__file__).resolve().parents[1] / "agent" / "tactical_formation.pkl"
00021
00022
00023 def __init__(self):
00024 """
00025 @brief Construtor da Classe, inicializa variáveis importantes, como o próprio dicionário.
00026 """
00027 # Iniciamos a interface
00028 super().__init__()
00029 self.title("RobotPositionManager")
00030 self.geometry("900x750")
00031
00032 # Configurações já existentes
00033 self.config_positions = RobotPositionManager.get_config_positions()
00034 self.nome_de_config_selecionada = None
00035
```

```

00036 # --- Constantes do Campo ---
00037 self.FIELD_WIDTH = 30
00038 self.FIELD_HEIGHT = 20
00039 self.GRID_SCALE = 25 # Pixels por unidade de campo
00040 self.MAX_JOGADORES = 11
00041 self.X_MIN = -self.FIELD_WIDTH / 2
00042 self.X_MAX = self.FIELD_WIDTH / 2
00043 self.Y_MIN = -self.FIELD_HEIGHT / 2
00044 self.Y_MAX = self.FIELD_HEIGHT / 2
00045
00046 # Variáveis de Estado
00047 self.posicoes_atuais = [] # Lista de tuplas do grid atual
00048 self.marcadores_jogadores = [] # Lista para rastreamos nossos jogadores
00049
00050 # Apenas variáveis que serão utilizadas posteriormente
00051 self.tv_configs = None # Para organizarmos a tabela de configurações
00052 self.canvas = None
00053 self.canvas_height = self.FIELD_HEIGHT * self.GRID_SCALE
00054 self.canvas_width = self.FIELD_WIDTH * self.GRID_SCALE
00055
00056 # Dispostemos as informações de forma inteligente
00057 self.criar_widgets()
00058 self.update_table_config()
00059
00060 # -- Métodos de Ajuda
00061 @staticmethod
00062 def get_config_positions() -> dict[str, list[tuple]]:
00063 """
00064 @brief Verificará existência do arquivo binário correspondente ao dicionário.
00065 @return Caso exista, o retornará restaurado. Caso não, retornará um dicionário vazio.
00066 """
00067
00068 if os.path.exists(RobotPositionManager.CONFIG_POSITION_PATH):
00069 # Caso exista, então devemos apenas restaurar
00070 with open(RobotPositionManager.CONFIG_POSITION_PATH, "rb") as f:
00071 return pickle.load(f)
00072
00073 # Logo, não existe
00074 return {"default": [(1, 2), (2, -3), (5, 4), (2, 2)], "default_1": [(1, 2), (2, 3), (5, 4),
(2, 2)]}
00075
00076 @staticmethod
00077 def save_config_positions(dados: dict[str, list[tuple]]) -> None:
00078 """
00079 @brief Responsável por salvar uma estrutura de dados em arquivo binário
00080 @param dados Estrutura de dados a ser salva
00081 """
00082
00083 with open(
00084 RobotPositionManager.CONFIG_POSITION_PATH,
00085 "wb"
00086) as f:
00087 # Colocamos esse comentário já que estava dando erro no interpretador da IDE
00088 pickle.dump(dados, f) # type: ignore
00089
00090 def _field_to_canvas(self, fx_: float, fy_: float) -> tuple:
00091 """
00092 @brief Responsável por converter coordenadas do campo para pixels no canvas
00093 @param fx_ Coordenada real em x
00094 @param fy_ Coordenada real em y
00095 @return Coordenadas corrigidas para o grid
00096 """
00097 return (
00098 (fx_ - self.X_MIN) * self.GRID_SCALE,
00099 (self.Y_MAX - fy_) * self.GRID_SCALE
00100)
00101
00102 def _canvas_to_field(self, cx: int, cy: int) -> tuple:
00103 """
00104 @brief Converterá o pixel clicado para o quadrado correspondente
00105 @param cx Posição X do pixel
00106 @param cy Posição Y do pixel
00107 @return tupla de posições reais
00108 """
00109
00110 # Converte pixel X para coordenada de campo
00111 fx_raw = (cx / self.GRID_SCALE) + self.X_MIN
00112
00113 # Converte pixel Y para coordenada de campo (invertendo a lógica)
00114 fy_raw = self.Y_MAX - (cy / self.GRID_SCALE)
00115
00116 # Arredonda para o 0.5 mais próximo
00117 fx_rounded = round(fx_raw * 2) / 2
00118 fy_rounded = round(fy_raw * 2) / 2
00119
00120 # Garante que o clique (mesmo fora) se encaixe nos limites
00121 return (

```

```

00122 max(self.X_MIN, min(self.X_MAX, fx_rounded)),
00123 max(self.Y_MIN, min(self.Y_MAX, fy_rounded))
00124)
00125
00126 # -- Métodos de Interface
00127 def criar_widgets(self):
00128 """
00129 @brief Disporá os widgets da interface de forma inteligente, provendo informações úteis.
00130 """
00131
00132 upper_frame = ttk.Frame(self)
00133 upper_frame.pack(side="top", fill="x", padx=10, pady=10)
00134
00135 config_frame = ttk.Frame(upper_frame)
00136 config_frame.pack(side="left", fill="both", expand=True)
00137
00138 # Dispostemos a tabela
00139 self.tv_configs = ttk.Treeview(config_frame, columns=("Nome", "Configuração"),
show="headings")
00140 self.tv_configs.heading("Nome", text="Nome")
00141 self.tv_configs.heading("Configuração", text="Configuração")
00142 self.tv_configs.column("Nome", width=50, anchor="center")
00143 self.tv_configs.column("Configuração", width=250)
00144
00145 self.tv_configs.pack(side="left", fill="both", expand=True)
00146 self.tv_configs.bind("<Double-1>", self.on_double_click_in_configson_double_click_in_configs)
00147
00148 frame_botoes = ttk.Frame(upper_frame)
00149 frame_botoes.pack(side="right", fill="y", padx=10)
00150
00151 ttk.Button(frame_botoes, text="Nova Configuração", command=self.nova_config).pack(fill="x",
pady=2)
00152 ttk.Button(frame_botoes, text="Salvar Atual", command=self.salvar_config).pack(fill="x",
pady=2)
00153 ttk.Button(frame_botoes, text="Apagar Seleccionada", command=self.apagar_config).pack(fill="x",
pady=2)
00154 ttk.Button(frame_botoes, text="Limpar Grade", command=lambda: (self.clear_grid(),
self.posicoes_atuais.clear())).pack(fill="x", pady=10)
00155
00156 # ----- Focando no campo
00157 frame_grid = ttk.Frame(self)
00158 frame_grid.pack(side="top", fill="both", expand=True, padx=10, pady=10)
00159
00160 # Canvas para o campo
00161 self.canvas = tk.Canvas(
00162 frame_grid,
00163 width=self.canvas_width,
00164 height=self.canvas_height,
00165 bg="#42f545" # Verde para o campo
00166)
00167 self.canvas.pack()
00168
00169 # Bind do clique no canvas
00170 self.canvas.bind("<Button-1>", self.click_on_gridclick_on_grid)
00171
00172 self.clear_grid()
00173
00174 def draw_player(self, field_x, field_y) -> None:
00175 """
00176 @brief Desenharemos um jogador na posição especificada
00177 @param field_x Posição real em X
00178 @param field_y Posição real em Y
00179 """
00180
00181 # Converte as coordenadas do campo (ex: -14, 0) para pixels
00182 cx, cy = self._field_to_canvas(field_x, field_y)
00183
00184 r = self.GRID_SCALE / 3
00185
00186 oval_id = self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r,
fill="yellow", outline="black", width=2)
00187
00188 self.marcadores_jogadores.append((oval_id, (field_x, field_y)))
00189
00190
00191 # -- Métodos de Interação
00192 def click_on_grid(self, event: tk.Event):
00193 """
00194 @brief Responsável por identificar onde o usuário clicou e adicionar essa posição na lista
00195 @param event Argumento default do bind
00196 """
00197
00198 new_pos = self._canvas_to_field(event.x, event.y)
00199
00200 # Verificamos se clicamos em cima de um jogador
00201 for i, (oval_id, pos) in enumerate(self.marcadores_jogadores):
00202 if pos == new_pos:
00203 self.canvas.delete(oval_id)

```

```

00204 self.marcadores_jogadores.pop(i)
00205 self.posicoes_atuais.remove(new_pos)
00206 return
00207
00208 # Verificamos se o limite de jogadores foi atingido
00209 if len(self.posicoes_atuais) >= self.MAX_JOGADORES:
00210 messagebox.showwarning("Limite Atingido",
00211 f"Não é possível adicionar mais de {self.MAX_JOGADORES}
jogadores.\n"
 "Clique em um jogador existente para removê-lo.")
00212 return
00213
00214 # Caso nenhuma das opções anteriores, adicionamos
00215 self.posicoes_atuais.append(new_pos)
00216 self.draw_player(*new_pos)
00217
00218 def on_double_click_in_configs(self, event: tk.Event) -> None:
00219 """
00220 @brief Responsável por plotar a configuração de jogadores selecionada
00221 @param event Argumento Default de bind
00222 """
00223
00224 item_selecionado = self.tv_configs.focus()
00225 if not item_selecionado:
00226 return
00227
00228 nome_config = self.tv_configs.item(item_selecionado, "values")[0]
00229 if nome_config in self.config_positions:
00230 self.posicoes_atuais = self.config_positions[nome_config][:]
00231 self.clear_grid()
00232 for (fx, fy) in self.posicoes_atuais:
00233 self.draw_player(fx, fy)
00234 self.nome_de_config_selecionada = nome_config
00235 else:
00236 messagebox.showwarning("Erro", f"Configuração '{nome_config}' não encontrada.")
00237
00238 def salvar_config(self) -> None:
00239 """
00240 @brief Salvará uma configuração selecionada
00241 """
00242
00243 item_selecionado = self.tv_configs.focus()
00244 if not item_selecionado:
00245 if not self.nome_de_config_selecionada:
00246 messagebox.showwarning("Inválido", "Não há selecionado")
00247 return
00248 else:
00249 nome_config = self.nome_de_config_selecionada
00250 else:
00251 nome_config = self.tv_configs.item(item_selecionado, "values")[0]
00252
00253 if messagebox.askyesno(
00254 "Certeza?",
00255 f"Realmente deseja salvar a configuração de jogadores presentes na grade em
{nome_config}?"
00256):
00257 # Atualizaremos
00258 self.config_positions[nome_config] = self.posicoes_atuais.copy()
00259 self.update_table_config()
00260 for item in self.tv_configs.get_children():
00261 if self.tv_configs.item(item, 'values')[0] == nome_config: # [0] = primeira coluna
00262 self.tv_configs.selection_set(item)
00263 self.nome_de_config_selecionada = nome_config
00264 break
00265
00266 def clear_grid(self) -> None:
00267 """
00268 @brief Responsável por limpar as posições e a grade
00269 """
00270
00271 self.canvas.delete("all")
00272 self.marcadores_jogadores = []
00273
00274 # Círculo central (usando a conversão de coordenadas)
00275 cx, cy = self._field_to_canvas(0,0)
00276 r = self.GRID_SCALE * 4 # Raio de 4 unidades
00277 self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r, outline="white", width=2)
00278
00279 # --- Desenhando Linhas da Grade (Quadrados) ---
00280
00281 # Total de passos de 0.5
00282 n_steps_x = int(self.FIELD_WIDTH * 2) + 1
00283 n_steps_y = int(self.FIELD_HEIGHT * 2) + 1
00284
00285 # Linhas Verticais (eixo X)
00286 for i in range(n_steps_x):
00287 fx = self.X_MIN + (i * 0.5)

```

```

00289
00290 # --- Lógica das Cores (Req. 3) ---
00291 cor = "white" if fx == 0 else "#337033"
00292 largura = 2 if fx == 0 else 1
00293
00294 # Converte a coordenada X para pixel
00295 cx, _ = self._field_to_canvas(fx, 0)
00296
00297 # Desenha a linha (Req. 2 - todas as linhas são desenhadas)
00298 self.canvas.create_line(cx, 0, cx, self.canvas_height,
00299 fill=cor, width=largura)
00300
00301 # Linhas Horizontais (eixo Y)
00302 for i in range(n_steps_y):
00303 fy = self.Y_MIN + (i * 0.5)
00304
00305 # --- Lógica das Cores (Req. 3) ---
00306 cor = "white" if fy == 0 else "#337033"
00307 largura = 2 if fy == 0 else 1
00308
00309 # Converte a coordenada Y para pixel
00310 _, cy = self._field_to_canvas(0, fy)
00311
00312 # Desenha a linha (Req. 2)
00313 self.canvas.create_line(0, cy, self.canvas_width, cy,
00314 fill=cor, width=largura)
00315
00316 # Caixas do Gol Esquerda (-15 a -13 em X, 3 a -3 em Y)
00317 coords_gol_esq = (-15, 3, -13, -3)
00318
00319 # Caixas do Gol Direita (13 a 15 em X, 3 a -3 em Y)
00320 coords_gol_dir = (13, 3, 15, -3)
00321
00322 # Converte e desenha o Gol Esquerdo
00323 x1, y1 = self._field_to_canvas(coords_gol_esq[0], coords_gol_esq[1])
00324 x2, y2 = self._field_to_canvas(coords_gol_esq[2], coords_gol_esq[3])
00325 self.canvas.create_rectangle(x1, y1, x2, y2, outline="white", width=3)
00326
00327 # Converte e desenha o Gol Direito
00328 x1, y1 = self._field_to_canvas(coords_gol_dir[0], coords_gol_dir[1])
00329 x2, y2 = self._field_to_canvas(coords_gol_dir[2], coords_gol_dir[3])
00330 self.canvas.create_rectangle(x1, y1, x2, y2, outline="white", width=3)
00331
00332 def nova_config(self) -> None:
00333 """
00334 @brief Prepará uma nova configuração para ser criada
00335 """
00336
00337 nome = simplifiedialog.askstring("Nova Configuração", "Digite o nome desejado:")
00338 if not nome:
00339 return
00340
00341 if nome in self.config_positions:
00342 messagebox.showwarning("Nome Inválido", "Já há uma configuração com este nome")
00343 return
00344
00345 # Atualizamos e setamos
00346 self.config_positions[nome] = []
00347 self.update_table_config()
00348 self.clear_grid()
00349 for item in self.tv_configs.get_children():
00350 if self.tv_configs.item(item, 'values')[0] == nome: # [0] = primeira coluna
00351 self.tv_configs.selection_set(item)
00352 self.nome_de_config_selecionada = nome
00353 break
00354
00355 def apagar_config(self) -> None:
00356 """
00357 @brief Apagará uma configuração selecionada
00358 """
00359
00360 item_selecionado = self.tv_configs.focus()
00361 if not item_selecionado:
00362 if not self.nome_de_config_selecionada:
00363 messagebox.showwarning("Inválido", "Não há nada para ser adicionado")
00364 return
00365 else:
00366 nome_config = self.nome_de_config_selecionada
00367 else:
00368 nome_config = self.tv_configs.item(item_selecionado, "values")[0]
00369
00370 if messagebox.askyesno("Confirmar", f"Tem certeza que deseja apagar a configuração '{nome_config}'?"):
00371 if nome_config in self.config_positions:
00372 self.nome_de_config_selecionada = None
00373 del self.config_positions[nome_config]
00374 self.update_table_config()

```

```
00375 self.clear_grid()
00376 self.posicoes_atuais.clear()
00377 messagebox.showinfo("Apagado", f"Configuração '{nome_config}' foi apagada.")
00378
00379 def update_table_config(self) -> None:
00380 """
00381 @brief Responsável por atualizar e preencher tabela de configurações de posição
00382 """
00383 for i in self.tv_configs.get_children():
00384 self.tv_configs.delete(i)
00385
00386 for chave, value in self.config_positions.items():
00387 self.tv_configs.insert("", "end", values=(chave, value))
00388
00389 # -- Métodos de Overload
00390 def destroy(self):
00391 RobotPositionManager.save_config_positions(self.config_positions)
00392 super().destroy()
00393
00394
00395
00396 if __name__ == '__main__':
00397 root = RobotPositionManager()
00398 root.mainloop()
00399
00400
00401
00402
00403
00404
```





# Index

- `__init__`
    - Agent.Agent, [14](#)
    - BaseAgent.BaseAgent, [18](#)
    - Booting.Booting, [20](#)
    - RobotPositionManager.RobotPositionManager, [48](#)
    - ServerComm.ServerComm, [56](#)
  - `__receive_async`
    - ServerComm.ServerComm, [57](#)
  - `_canvas_to_field`
    - RobotPositionManager.RobotPositionManager, [48](#)
  - `_current_buffer`
    - Logger, [34](#)
  - `_cv`
    - Logger, [34](#)
  - `_field_to_canvas`
    - RobotPositionManager.RobotPositionManager, [48](#)
  - `_file_stream`
    - Logger, [34](#)
  - `_init_file`
    - Logger, [30](#)
  - `_is_running`
    - Logger, [34](#)
  - `_mutex`
    - Logger, [35](#)
  - `_worker`
    - Logger, [35](#)
  - `_worker_loop`
    - Logger, [30](#)
  - `_write_buffer`
    - Logger, [35](#)
  - `~Logger`
    - Logger, [30](#)
- ACTIVE\_BEAM
  - Environment, [25](#)
- advance
  - Environment::Parsing, [38](#)
- Agent, [9](#)
- Agent.Agent, [13](#)
  - `__init__`, [14](#)
  - unum, [15](#)
- AgentPenalty, [9](#)
- AGENTS\_IN\_THE\_MATCH
  - BaseAgent.BaseAgent, [18](#)
- apagar\_config
  - RobotPositionManager.RobotPositionManager, [49](#)
- BaseAgent, [9](#)
- BaseAgent.BaseAgent, [15](#)
  - `__init__`, [18](#)
- AGENTS\_IN\_THE\_MATCH, [18](#)
- beam, [18](#)
- env, [18](#)
- init\_position, [18](#)
- INITIAL\_POSITION, [19](#)
- logger, [19](#)
- scom, [19](#)
- unum, [19](#)
- beam
  - BaseAgent.BaseAgent, [18](#)
- BEFORE\_KICKOFF
  - Environment, [25](#)
- boot
  - run\_full\_team, [10](#)
  - run\_player, [11](#)
- Booting, [9](#)
- Booting.Booting, [19](#)
  - `__init__`, [20](#)
  - CONFIG\_PATH, [21](#)
  - cpp\_builder, [21](#)
  - get\_team\_params, [21](#)
  - options, [21](#)
  - show\_spinner, [21](#)
- buffer
  - Environment::Parsing, [40](#)
  - ServerComm.ServerComm, [59](#)
- buffer\_size
  - ServerComm.ServerComm, [59](#)
- canvas
  - RobotPositionManager.RobotPositionManager, [52](#)
- canvas\_height
  - RobotPositionManager.RobotPositionManager, [52](#)
- canvas\_width
  - RobotPositionManager.RobotPositionManager, [52](#)
- clear\_grid
  - RobotPositionManager.RobotPositionManager, [49](#)
- clear\_queue
  - ServerComm.ServerComm, [57](#)
- click\_on\_grid
  - RobotPositionManager.RobotPositionManager, [49](#), [52](#)
- close
  - ServerComm.ServerComm, [57](#)
- commit
  - ServerComm.ServerComm, [57](#)
- commit\_beam
  - ServerComm.ServerComm, [58](#)
- CONFIG\_PATH
  - Booting.Booting, [21](#)

- CONFIG\_POSITION\_PATH
  - RobotPositionManager.RobotPositionManager, 52
- config\_positions
  - RobotPositionManager.RobotPositionManager, 52
- CONSOLE
  - Printing.Printing, 43
- cpp\_builder
  - Booting.Booting, 21
- criar\_widgets
  - RobotPositionManager.RobotPositionManager, 50
- debug.cc
  - example, 68
  - example1, 69
  - main, 68, 71
  - size, 69
  - size1, 69
  - tarefaPesada, 71
- destroy
  - RobotPositionManager.RobotPositionManager, 50
- draw\_player
  - RobotPositionManager.RobotPositionManager, 50
- end
  - Environment::Parsing, 40
- env
  - BaseAgent.BaseAgent, 18
  - Environment::Parsing, 40
  - ServerComm.ServerComm, 59
- Environment, 22
  - ACTIVE\_BEAM, 25
  - BEFORE\_KICKOFF, 25
  - Environment, 25
  - GAME\_OVER, 25
  - goals\_conceded, 26
  - goals\_scored, 26
  - is\_left, 27
  - logger, 27
  - OTHER, 25
  - OUR\_CORNER\_KICK, 25
  - OUR\_DIR\_FREE\_KICK, 25
  - OUR\_FREE\_KICK, 25
  - OUR\_GOAL, 25
  - OUR\_GOAL\_KICK, 25
  - OUR\_KICK, 25
  - OUR\_KICK\_IN, 25
  - OUR\_KICKOFF, 25
  - OUR\_OFFSIDE, 25
  - OUR\_PASS, 25
  - PASSIVE\_BEAM, 25
  - PLAY\_ON, 25
  - PlayMode, 24
  - PlayModeGroup, 25
  - print\_status, 26
  - THEIR\_CORNER\_KICK, 25
  - THEIR\_DIR\_FREE\_KICK, 25
  - THEIR\_FREE\_KICK, 25
  - THEIR\_GOAL, 25
  - THEIR\_GOAL\_KICK, 25
  - THEIR\_KICK, 25
  - THEIR\_KICK\_IN, 25
  - THEIR\_KICKOFF, 25
  - THEIR\_OFFSIDE, 25
  - THEIR\_PASS, 25
  - time\_match, 27
  - time\_server, 27
  - unum, 27
  - update\_from\_server, 26
- Environment.hpp
  - False, 74
  - True, 74
- Environment::Parsing, 36
  - advance, 38
  - buffer, 40
  - end, 40
  - env, 40
  - get\_str, 38
  - get\_value, 38
  - parse\_gamestate, 39
  - parse\_time, 39
  - Parsing, 37
  - skip\_until\_char, 39
- error
  - Logger, 31
- example
  - debug.cc, 68
- example1
  - debug.cc, 69
- False
  - Environment.hpp, 74
  - Logger.hpp, 80
- FIELD\_HEIGHT
  - RobotPositionManager.RobotPositionManager, 53
- FIELD\_WIDTH
  - RobotPositionManager.RobotPositionManager, 53
- GAME\_OVER
  - Environment, 25
- get
  - Logger, 31
- get\_config\_positions
  - RobotPositionManager.RobotPositionManager, 50
- get\_input
  - Printing.Printing, 42
- get\_str
  - Environment::Parsing, 38
- get\_team\_params
  - Booting.Booting, 21
- get\_value
  - Environment::Parsing, 38
- goals\_conceded
  - Environment, 26
- goals\_scored
  - Environment, 26
- GRID\_SCALE
  - RobotPositionManager.RobotPositionManager, 53

- IF\_IN\_DEBUG
  - Printing.Printing, 43
- info
  - Logger, 31, 33
- init\_position
  - BaseAgent.BaseAgent, 18
- INITIAL\_POSITION
  - BaseAgent.BaseAgent, 19
- is\_left
  - Environment, 27
- is\_the\_first
  - Logger, 35
- log
  - Logger, 33
- Logger, 28
  - \_current\_buffer, 34
  - \_cv, 34
  - \_file\_stream, 34
  - \_init\_file, 30
  - \_is\_running, 34
  - \_mutex, 35
  - \_worker, 35
  - \_worker\_loop, 30
  - \_write\_buffer, 35
  - ~Logger, 30
  - error, 31
  - get, 31
  - info, 31, 33
  - is\_the\_first, 35
  - log, 33
  - Logger, 30
  - operator=, 33
  - warn, 33, 34
- logger
  - BaseAgent.BaseAgent, 19
  - Environment, 27
- Logger.hpp
  - False, 80
  - True, 80
- main
  - debug.cc, 68, 71
- marcadores\_jogadores
  - RobotPositionManager.RobotPositionManager, 53
- MAX\_JOGADORES
  - RobotPositionManager.RobotPositionManager, 53
- message\_queue
  - ServerComm.ServerComm, 59
- module\_main.cpp
  - NB\_MODULE, 77, 79
- NB\_MODULE
  - module\_main.cpp, 77, 79
- nome\_de\_config\_selecionada
  - RobotPositionManager.RobotPositionManager, 53
- nova\_config
  - RobotPositionManager.RobotPositionManager, 51
- on\_double\_click\_in\_configs
  - RobotPositionManager.RobotPositionManager, 51, 53
- operator=
  - Logger, 33
- options
  - Booting.Booting, 21
- OTHER
  - Environment, 25
- OUR\_CORNER\_KICK
  - Environment, 25
- OUR\_DIR\_FREE\_KICK
  - Environment, 25
- OUR\_FREE\_KICK
  - Environment, 25
- OUR\_GOAL
  - Environment, 25
- OUR\_GOAL\_KICK
  - Environment, 25
- OUR\_KICK
  - Environment, 25
- OUR\_KICK\_IN
  - Environment, 25
- OUR\_KICKOFF
  - Environment, 25
- OUR\_OFFSIDE
  - Environment, 25
- OUR\_PASS
  - Environment, 25
- p
  - run\_full\_team, 10
  - run\_player, 11
- parse\_gamestate
  - Environment::Parsing, 39
- parse\_time
  - Environment::Parsing, 39
- Parsing
  - Environment::Parsing, 37
- PASSIVE\_BEAM
  - Environment, 25
- PLAY\_ON
  - Environment, 25
- players
  - run\_full\_team, 10
- PlayMode
  - Environment, 24
- PlayModeGroup
  - Environment, 25
- posicoes\_atuais
  - RobotPositionManager.RobotPositionManager, 53
- print\_message
  - Printing.Printing, 42
- print\_status
  - Environment, 26
- print\_table
  - Printing.Printing, 43
- Printing, 9
- Printing.Printing, 41

- CONSOLE, 43
- get\_input, 42
- IF\_IN\_DEBUG, 43
- print\_message, 42
- print\_table, 43
- TABLE\_COLORS, 43
- receive
  - ServerComm.ServerComm, 58
- RobotPositionManager, 10
  - root, 10
- RobotPositionManager.RobotPositionManager, 44
  - \_\_init\_\_, 48
  - \_canvas\_to\_field, 48
  - \_field\_to\_canvas, 48
  - apagar\_config, 49
  - canvas, 52
  - canvas\_height, 52
  - canvas\_width, 52
  - clear\_grid, 49
  - click\_on\_grid, 49, 52
  - CONFIG\_POSITION\_PATH, 52
  - config\_positions, 52
  - criar\_widgets, 50
  - destroy, 50
  - draw\_player, 50
  - FIELD\_HEIGHT, 53
  - FIELD\_WIDTH, 53
  - get\_config\_positions, 50
  - GRID\_SCALE, 53
  - marcadores\_jogadores, 53
  - MAX\_JOGADORES, 53
  - nome\_de\_config\_selecionada, 53
  - nova\_config, 51
  - on\_double\_click\_in\_configs, 51, 53
  - posicoes\_atuais, 53
  - salvar\_config, 51
  - save\_config\_positions, 51
  - tv\_configs, 54
  - update\_table\_config, 52
  - X\_MAX, 54
  - X\_MIN, 54
  - Y\_MAX, 54
  - Y\_MIN, 54
- root
  - RobotPositionManager, 10
- run\_full\_team, 10
  - boot, 10
  - p, 10
  - players, 10
- run\_player, 11
  - boot, 11
  - p, 11
- salvar\_config
  - RobotPositionManager.RobotPositionManager, 51
- save\_config\_positions
  - RobotPositionManager.RobotPositionManager, 51
- scom
  - BaseAgent.BaseAgent, 19
- send
  - ServerComm.ServerComm, 58
- send\_immediate
  - ServerComm.ServerComm, 58
- ServerComm, 11
- ServerComm.ServerComm, 55
  - \_\_init\_\_, 56
  - \_\_receive\_async, 57
  - buffer, 59
  - buffer\_size, 59
  - clear\_queue, 57
  - close, 57
  - commit, 57
  - commit\_beam, 58
  - env, 59
  - message\_queue, 59
  - receive, 58
  - send, 58
  - send\_immediate, 58
  - socket, 59
  - unum, 59
- show\_spinner
  - Booting.Booting, 21
- size
  - debug.cc, 69
- size1
  - debug.cc, 69
- skip\_until\_char
  - Environment::Parsing, 39
- socket
  - ServerComm.ServerComm, 59
- src/agent/Agent.py, 61
- src/agent/AgentPenalty.py, 62
- src/agent/BaseAgent.py, 62, 63
- src/communication/ServerComm.py, 64
- src/cpp/environment/debug.cc, 67, 70
- src/cpp/environment/Environment.hpp, 73, 74
- src/cpp/environment/module\_main.cpp, 77, 78
- src/cpp/logger/debug.cc, 70, 71
- src/cpp/logger/Logger.hpp, 80, 81
- src/cpp/logger/module\_main.cpp, 78, 79
- src/run\_full\_team.py, 83
- src/run\_player.py, 83, 84
- src/term/Booting.py, 84
- src/term/Printing.py, 87, 88
- src/utils/RobotPositionManager.py, 90
- TABLE\_COLORS
  - Printing.Printing, 43
- taskPesada
  - debug.cc, 71
- THEIR\_CORNER\_KICK
  - Environment, 25
- THEIR\_DIR\_FREE\_KICK
  - Environment, 25
- THEIR\_FREE\_KICK
  - Environment, 25
- THEIR\_GOAL

- Environment, [25](#)
- THEIR\_GOAL\_KICK
  - Environment, [25](#)
- THEIR\_KICK
  - Environment, [25](#)
- THEIR\_KICK\_IN
  - Environment, [25](#)
- THEIR\_KICKOFF
  - Environment, [25](#)
- THEIR\_OFFSIDE
  - Environment, [25](#)
- THEIR\_PASS
  - Environment, [25](#)
- time\_match
  - Environment, [27](#)
- time\_server
  - Environment, [27](#)
- True
  - Environment.hpp, [74](#)
  - Logger.hpp, [80](#)
- tv\_configs
  - RobotPositionManager.RobotPositionManager, [54](#)
- unum
  - Agent.Agent, [15](#)
  - BaseAgent.BaseAgent, [19](#)
  - Environment, [27](#)
  - ServerComm.ServerComm, [59](#)
- update\_from\_server
  - Environment, [26](#)
- update\_table\_config
  - RobotPositionManager.RobotPositionManager, [52](#)
- warn
  - Logger, [33](#), [34](#)
- X\_MAX
  - RobotPositionManager.RobotPositionManager, [54](#)
- X\_MIN
  - RobotPositionManager.RobotPositionManager, [54](#)
- Y\_MAX
  - RobotPositionManager.RobotPositionManager, [54](#)
- Y\_MIN
  - RobotPositionManager.RobotPositionManager, [54](#)