

SSRoboime

Generated by Doxygen 1.9.8



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>7</b>
4.1 File List	7
<b>5 Namespace Documentation</b>	<b>9</b>
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
5.9 ServerComm Namespace Reference	11
<b>6 Class Documentation</b>	<b>13</b>
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 environment	18
6.2.4.3 init_position	19
6.2.4.4 INITIAL_POSITION	19
6.2.4.5 scom	19
6.2.4.6 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	23
6.4.2 Member Enumeration Documentation	23
6.4.2.1 PlayMode	23
6.4.2.2 PlayModeGroup	24
6.4.3 Member Function Documentation	24
6.4.3.1 update_from_server()	24
6.4.4 Member Data Documentation	25
6.4.4.1 time_match	25
6.4.4.2 time_server	25
6.5 Printing.Printing Class Reference	25
6.5.1 Detailed Description	26
6.5.2 Member Function Documentation	26
6.5.2.1 get_input()	26
6.5.2.2 print_message()	27
6.5.2.3 print_table()	27
6.5.3 Member Data Documentation	28
6.5.3.1 CONSOLE	28
6.5.3.2 IF_IN_DEBUG	28
6.5.3.3 TABLE_COLORS	28
6.6 RobotPositionManager.RobotPositionManager Class Reference	28
6.6.1 Detailed Description	32
6.6.2 Constructor & Destructor Documentation	32
6.6.2.1 __init__()	32

6.6.3 Member Function Documentation	32
6.6.3.1 _canvas_to_field()	32
6.6.3.2 _field_to_canvas()	33
6.6.3.3 apagar_config()	33
6.6.3.4 clear_grid()	33
6.6.3.5 click_on_grid()	33
6.6.3.6 criar_widgets()	34
6.6.3.7 destroy()	34
6.6.3.8 draw_player()	34
6.6.3.9 get_config_positions()	34
6.6.3.10 nova_config()	35
6.6.3.11 on_double_click_in_configs()	35
6.6.3.12 salvar_config()	35
6.6.3.13 save_config_positions()	35
6.6.3.14 update_table_config()	36
6.6.4 Member Data Documentation	36
6.6.4.1 canvas	36
6.6.4.2 canvas_height	36
6.6.4.3 canvas_width	36
6.6.4.4 click_on_grid	36
6.6.4.5 CONFIG_POSITION_PATH	36
6.6.4.6 config_positions	37
6.6.4.7 FIELD_HEIGHT	37
6.6.4.8 FIELD_WIDTH	37
6.6.4.9 GRID_SCALE	37
6.6.4.10 marcadores_jogadores	37
6.6.4.11 MAX_JOGADORES	37
6.6.4.12 nome_de_config_selecionada	37
6.6.4.13 on_double_click_in_configs	37
6.6.4.14 posicoes_atuais	38
6.6.4.15 tv_configs	38
6.6.4.16 X_MAX	38
6.6.4.17 X_MIN	38
6.6.4.18 Y_MAX	38
6.6.4.19 Y_MIN	38
6.7 ServerComm.ServerComm Class Reference	39
6.7.1 Detailed Description	40
6.7.2 Constructor & Destructor Documentation	40
6.7.2.1 __init__()	40
6.7.3 Member Function Documentation	41
6.7.3.1 __receive_async()	41
6.7.3.2 clear_queue()	41

6.7.3.3 close()	41
6.7.3.4 commit()	41
6.7.3.5 commit_beam()	42
6.7.3.6 receive()	42
6.7.3.7 send()	42
6.7.3.8 send_immediate()	42
6.7.4 Member Data Documentation	43
6.7.4.1 buffer	43
6.7.4.2 buffer_size	43
6.7.4.3 environment	43
6.7.4.4 message_queue	43
6.7.4.5 socket	43
6.7.4.6 unum	43
<b>7 File Documentation</b>	<b>45</b>
7.1 src/agent/Agent.py File Reference	45
7.1.1 Detailed Description	45
7.2 Agent.py	45
7.3 src/agent/AgentPenalty.py File Reference	46
7.3.1 Detailed Description	46
7.4 AgentPenalty.py	46
7.5 src/agent/BaseAgent.py File Reference	46
7.5.1 Detailed Description	47
7.6 BaseAgent.py	47
7.7 src/communication/ServerComm.py File Reference	48
7.7.1 Detailed Description	48
7.8 ServerComm.py	48
7.9 src/cpp/environment/debug.cc File Reference	51
7.9.1 Function Documentation	52
7.9.1.1 main()	52
7.9.2 Variable Documentation	52
7.9.2.1 example	52
7.9.2.2 example1	53
7.9.2.3 size	53
7.9.2.4 size1	53
7.10 debug.cc	54
7.11 src/cpp/environment/Environment.hpp File Reference	54
7.11.1 Macro Definition Documentation	55
7.11.1.1 False	55
7.11.1.2 True	56
7.12 Environment.hpp	56
7.13 src/cpp/environment/module_main.cpp File Reference	57

---

7.13.1 Function Documentation . . . . .	58
7.13.1.1 NB_MODULE() . . . . .	58
7.14 module_main.cpp . . . . .	58
7.15 src/run_full_team.py File Reference . . . . .	59
7.16 run_full_team.py . . . . .	59
7.17 src/run_player.py File Reference . . . . .	59
7.18 run_player.py . . . . .	59
7.19 src/term/Booting.py File Reference . . . . .	60
7.19.1 Detailed Description . . . . .	60
7.20 Booting.py . . . . .	60
7.21 src/term/Printing.py File Reference . . . . .	63
7.21.1 Detailed Description . . . . .	63
7.22 Printing.py . . . . .	63
7.23 src/utils/RobotPositionManager.py File Reference . . . . .	65
7.23.1 Detailed Description . . . . .	66
7.24 RobotPositionManager.py . . . . .	66
<b>Index</b>	<b>73</b>





# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">Agent</a>	9
<a href="#">AgentPenalty</a>	9
<a href="#">BaseAgent</a>	9
<a href="#">Booting</a>	9
<a href="#">Printing</a>	9
<a href="#">RobotPositionManager</a>	10
<a href="#">run_full_team</a>	10
<a href="#">run_player</a>	11
<a href="#">ServerComm</a>	11



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

Booting.Booting . . . . .	19
Environment . . . . .	22
Printing.Printing . . . . .	25
ServerComm.ServerComm . . . . .	39
tk.Tk	
RobotPositionManager.RobotPositionManager . . . . .	28
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:

<a href="#">Agent.Agent</a>	Classe que representará os agentes de campo, possuindo métodos correspondentes . . . . .	13
<a href="#">BaseAgent.BaseAgent</a>	Classe que agrupará todas as funcionalidades comuns a qualquer agente . . . . .	15
<a href="#">Booting.Booting</a>	Responsável por inicializar todas as necessidades de execução do time . . . . .	19
<a href="#">Environment</a>	Responsável por representar o ambiente externo ao robô . . . . .	22
<a href="#">Printing.Printing</a>	Responsável pela comunicação usuário - terminal . . . . .	25
<a href="#">RobotPositionManager.RobotPositionManager</a>	Responsável por permitir ao usuário a criação de diversas formações táticas . . . . .	28
<a href="#">ServerComm.ServerComm</a>	Responsável pela comunicação com servidor . . . . .	39



# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

src/run_full_team.py . . . . .	59
src/run_player.py . . . . .	59
src/agent/Agent.py . . . . .	
Implementação de Lógica de Agente de Campo . . . . .	45
src/agent/AgentPenalty.py . . . . .	
Implementação de Lógica de Goleiro . . . . .	46
src/agent/BaseAgent.py . . . . .	
Implementação da classe de jogador base, que deve ser comum a todos os agentes . . . . .	46
src/communication/ServerComm.py . . . . .	
Implementação da Comunicação com Servidor . . . . .	48
src/cpp/environment/debug.cc . . . . .	51
src/cpp/environment/Environment.hpp . . . . .	54
src/cpp/environment/module_main.cpp . . . . .	57
src/term/Booting.py . . . . .	
Implementação do Booting do time . . . . .	60
src/term/Printing.py . . . . .	
Implementação de Interface no terminal . . . . .	63
src/utils/RobotPositionManager.py . . . . .	
Implementação de lógica organizadora de posições iniciais de partida . . . . .	65





## 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()`

### 5.8.1 Variable Documentation

#### 5.8.1.1 boot

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

Definition at line 4 of file `run_player.py`.

## 5.9 ServerComm Namespace Reference

### Classes

- class `ServerComm`  
*Responsável pela comunicação com servidor.*



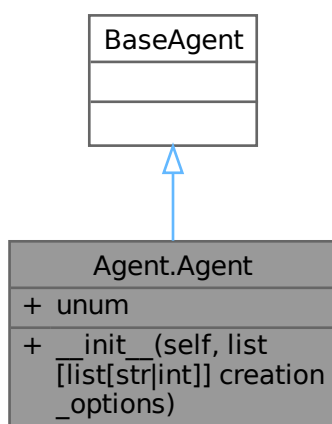
## Chapter 6

# Class Documentation

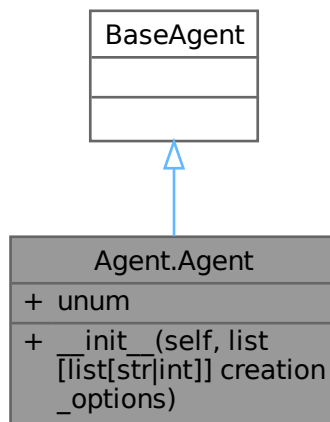
### 6.1 Agent.Agent Class Reference

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](#).

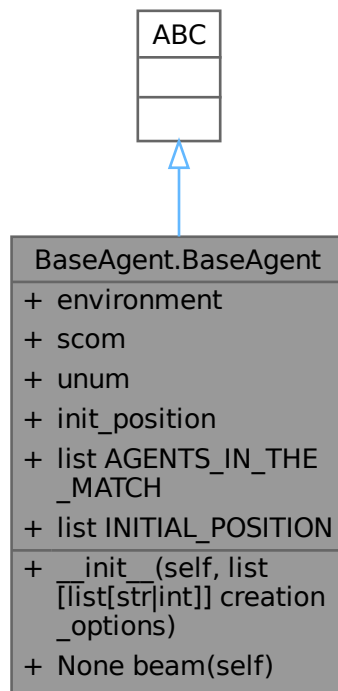
The documentation for this class was generated from the following file:

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

## 6.2 BaseAgent.BaseAgent Class Reference

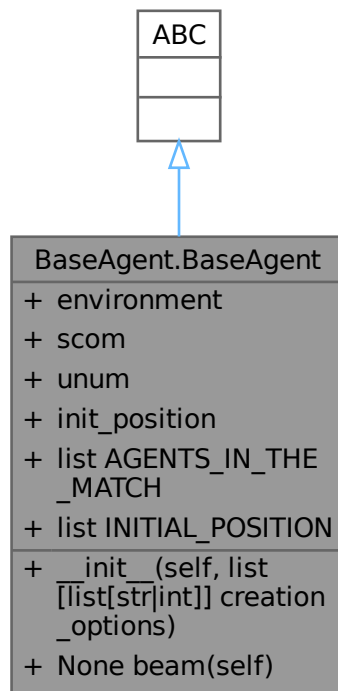
Classe que agrupará todas as funcionalidades comuns a qualquer agente.

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

- `environment`
- `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 11 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 19 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 51 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 16 of file [BaseAgent.py](#).

### 6.2.4.2 `environment`

```
BaseAgent.BaseAgent.environment
```

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

#### 6.2.4.3 init\_position

`BaseAgent.BaseAgent.init_position`

Definition at line 49 of file [BaseAgent.py](#).

#### 6.2.4.4 INITIAL\_POSITION

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

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

#### 6.2.4.5 scom

`BaseAgent.BaseAgent.scom`

Definition at line 27 of file [BaseAgent.py](#).

#### 6.2.4.6 unum

`BaseAgent.BaseAgent.unum`

Definition at line 36 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:

Environment
+ float time_server
+ float time_match
+ int update_from_server (std::string_view msg)

### 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

- int [update\\_from\\_server](#) (std::string\_view msg)  
*Completamente responsável por realizar o parsing da mensagem do servidor.*

## Public Attributes

- float `time_server` = 0  
*Instante de Tempo do Servidor, útil apenas para sincronização entre agentes.*
- float `time_match` = 0  
*Instante de Tempo de Partida.*

### 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 18 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 24 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 52 of file [Environment.hpp](#).

## 6.4.3 Member Function Documentation

### 6.4.3.1 update\_from\_server()

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

Completamente responsável por realizar o parsing da mensagem do servidor.

#### Parameters

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

#### Returns

Atualização de todas as variáveis de ambiente

< Com isso, podemos mover o ponteiro sem modificar o conteúdo

< Garante que chegamos no ( anterior a tag

< Pulamos mais um para o início da tag

< Buffer se encontra no espaço(' ') imediatamente após a tag

< Lê-se 'time' de momento, em inglês, não grupo de jogadores.

< O caso do time é o mais trivial

< Função especial para máxima performance

< Game State

< Score Left

Definition at line 70 of file [Environment.hpp](#).



## 6.4.4 Member Data Documentation

### 6.4.4.1 time\_match

```
float Environment::time_match = 0
```

Instante de Tempo de Partida.

Definition at line 22 of file [Environment.hpp](#).

### 6.4.4.2 time\_server

```
float Environment::time_server = 0
```

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

Definition at line 21 of file [Environment.hpp](#).

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

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

## 6.5 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()

## 6.5.1 Detailed Description

Responsável pela comunicação usuário - terminal.

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

## 6.5.2 Member Function Documentation

### 6.5.2.1 [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](#).

### 6.5.2.2 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](#).

### 6.5.2.3 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](#).

## 6.5.3 Member Data Documentation

### 6.5.3.1 CONSOLE

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

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

### 6.5.3.2 IF\_IN\_DEBUG

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

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

### 6.5.3.3 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](#).

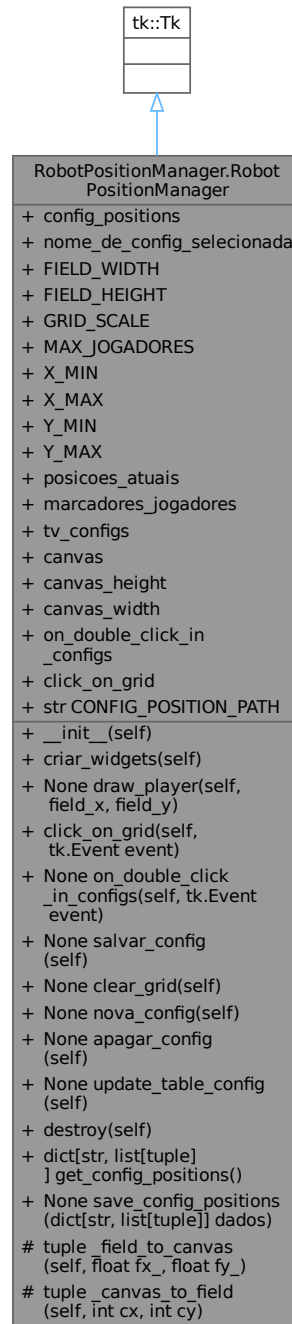
The documentation for this class was generated from the following file:

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

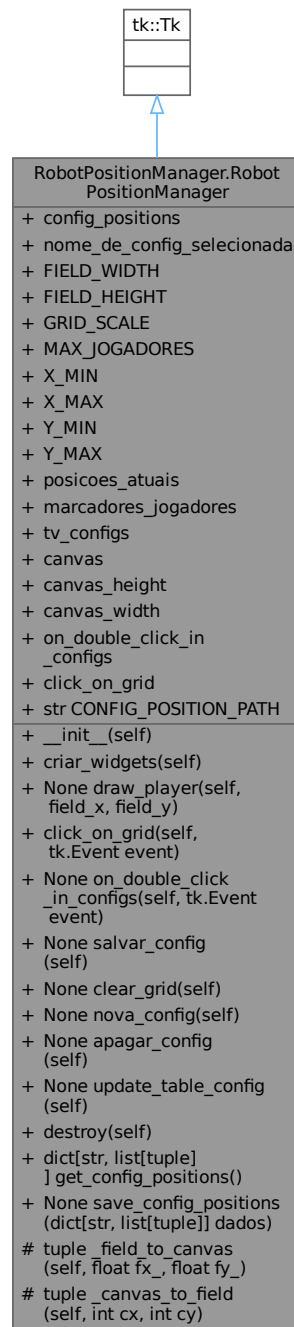
## 6.6 RobotPositionManager.RobotPositionManager Class Reference

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.*

### 6.6.1 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](#).

### 6.6.2 Constructor & Destructor Documentation

#### 6.6.2.1 `__init__()`

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

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

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

### 6.6.3 Member Function Documentation

#### 6.6.3.1 `_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](#).



### 6.6.3.2 \_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](#).

### 6.6.3.3 apagar\_config()

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

Apagará uma configuração selecionada.

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

### 6.6.3.4 clear\_grid()

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

Responsável por limpar as posições e a grade.

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

### 6.6.3.5 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](#).

### 6.6.3.6 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](#).

### 6.6.3.7 destroy()

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

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

### 6.6.3.8 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](#).

### 6.6.3.9 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](#).

**6.6.3.10 nova\_config()**

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

Preparará uma nova configuração para ser criada.

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

**6.6.3.11 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](#).

**6.6.3.12 salvar\_config()**

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

Salvará uma configuração selecionada.

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

**6.6.3.13 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](#).

#### 6.6.3.14 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](#).

### 6.6.4 Member Data Documentation

#### 6.6.4.1 canvas

```
RobotPositionManager.RobotPositionManager.canvas
```

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

#### 6.6.4.2 canvas\_height

```
RobotPositionManager.RobotPositionManager.canvas_height
```

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

#### 6.6.4.3 canvas\_width

```
RobotPositionManager.RobotPositionManager.canvas_width
```

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

#### 6.6.4.4 click\_on\_grid

```
RobotPositionManager.RobotPositionManager.click_on_grid
```

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

#### 6.6.4.5 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](#).

#### 6.6.4.6 config\_positions

RobotPositionManager.RobotPositionManager.config\_positions

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

#### 6.6.4.7 FIELD\_HEIGHT

RobotPositionManager.RobotPositionManager.FIELD\_HEIGHT

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

#### 6.6.4.8 FIELD\_WIDTH

RobotPositionManager.RobotPositionManager.FIELD\_WIDTH

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

#### 6.6.4.9 GRID\_SCALE

RobotPositionManager.RobotPositionManager.GRID\_SCALE

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

#### 6.6.4.10 marcadores\_jogadores

RobotPositionManager.RobotPositionManager.marcadores\_jogadores

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

#### 6.6.4.11 MAX\_JOGADORES

RobotPositionManager.RobotPositionManager.MAX\_JOGADORES

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

#### 6.6.4.12 nome\_de\_config\_selecionada

RobotPositionManager.RobotPositionManager.nome\_de\_config\_selecionada

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

#### 6.6.4.13 on\_double\_click\_in\_configs

RobotPositionManager.RobotPositionManager.on\_double\_click\_in\_configs

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

#### 6.6.4.14 posicoes\_atuais

`RobotPositionManager.RobotPositionManager.posicoes_atuais`

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

#### 6.6.4.15 tv\_configs

`RobotPositionManager.RobotPositionManager.tv_configs`

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

#### 6.6.4.16 X\_MAX

`RobotPositionManager.RobotPositionManager.X_MAX`

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

#### 6.6.4.17 X\_MIN

`RobotPositionManager.RobotPositionManager.X_MIN`

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

#### 6.6.4.18 Y\_MAX

`RobotPositionManager.RobotPositionManager.Y_MAX`

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

#### 6.6.4.19 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](#)

## 6.7 ServerComm.ServerComm Class Reference

Responsável pela comunicação com servidor.

Collaboration diagram for ServerComm.ServerComm:

ServerComm.ServerComm
+ buffer_size + buffer + socket + message_queue + unum + environment
+ <code>__init__</code> (self, list [list[str]] creation_options, Environment environment, list other_players) + None send_immediate(self, bytes message) + None receive(self) + None commit(self, bytes message) + None close(self) + None send(self) + None clear_queue(self) + commit_beam(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 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)  
*Enviará ao servidor todas as mensagens commitadas.*
- None [clear\\_queue](#) (self)  
*Limpará 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](#)
- [environment](#)

## Private Member Functions

- None [\\_\\_receive\\_async](#) (self, list other\_players)  
*Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de execução.*

### 6.7.1 Detailed Description

Responsável pela comunicação com servidor.

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

### 6.7.2 Constructor & Destructor Documentation

#### 6.7.2.1 `__init__()`

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

Construtor da classe, inicializando buffers e a conexão de cada agente com servidor.

#### Parameters

<i>creation_options</i>	Lista de parâmetros de criação, self ainda não foi incluído na lista.
<i>environment</i>	
<i>other_players</i>	

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



## 6.7.3 Member Function Documentation

### 6.7.3.1 \_\_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 146 of file [ServerComm.py](#).

### 6.7.3.2 clear\_queue()

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

Limpará a fila de commits.

Definition at line 218 of file [ServerComm.py](#).

### 6.7.3.3 close()

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

Responsável por fazer o encerramento dos canais de comunicação.

Definition at line 191 of file [ServerComm.py](#).

### 6.7.3.4 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 183 of file [ServerComm.py](#).

#### 6.7.3.5 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 225 of file [ServerComm.py](#).

#### 6.7.3.6 receive()

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

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

Definition at line 96 of file [ServerComm.py](#).

#### 6.7.3.7 send()

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

Enviará ao servidor todas as mensagens commitadas.

Definition at line 198 of file [ServerComm.py](#).

#### 6.7.3.8 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 81 of file [ServerComm.py](#).

## 6.7.4 Member Data Documentation

### 6.7.4.1 buffer

`ServerComm.ServerComm.buffer`

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

### 6.7.4.2 buffer\_size

`ServerComm.ServerComm.buffer_size`

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

### 6.7.4.3 environment

`ServerComm.ServerComm.environment`

Definition at line 36 of file [ServerComm.py](#).

### 6.7.4.4 message\_queue

`ServerComm.ServerComm.message_queue`

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

### 6.7.4.5 socket

`ServerComm.ServerComm.socket`

Definition at line 27 of file [ServerComm.py](#).

### 6.7.4.6 unum

`ServerComm.ServerComm.unum`

Definition at line 35 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 cpp.environment.environment import Environment
00008 from pathlib import Path
00009 import pickle
00010
00011 class BaseAgent(ABC):
00012     """
00013     @brief Classe que agrupará todas as funcionalidades comuns a qualquer agente.
00014     """
00015
00016     AGENTS_IN_THE_MATCH = []
00017     INITIAL_POSITION = []
00018
00019     def __init__(self, creation_options: list[list[str | int]]):
00020         """
00021         @brief Construtor da classe base de agente, chamando todos os construtores de outras
00022         classes mínimas para cada agente.
00023         @param creation_options Lista de Parâmetros de Criação de Agente
00024         """
00025
00026         self.environment = Environment()
00027         self.scom = ServerComm(
00028             creation_options,
00029             self.environment,
00030             # Passamos o ponteiro da lista de jogadores
00031             # Conforme eles são inseridos, teremos novos na partida
00032             BaseAgent.AGENTS_IN_THE_MATCH
00033         )
00034         # Chamaremos os construtores mínimos conforme formos criando-os
00035
00036         self.unum = creation_options[4][1]
00037         # Note que colocamos apenas por último
00038         BaseAgent.AGENTS_IN_THE_MATCH.append(self)
00039
00040         # Garantimos que as posições são existentes
00041         # E executamos apenas uma vez
00042         if not BaseAgent.INITIAL_POSITION:
00043             with open(
00044                 Path(__file__).resolve().parent / "tactical_formation.pkl",
00045                 "rb"
00046             ) as f:
00047                 BaseAgent.INITIAL_POSITION = pickle.load(f)["default"]
00048
00049         self.init_position = BaseAgent.INITIAL_POSITION[self.unum - 1]
00050
00051     def beam(self) -> None:
00052         """
00053         @brief Responsável por gerenciar o teletransporte dos jogadores
00054         """
00055
00056         self.scom.commit_beam(self.init_position, 0)
00057
00058 
```

## 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 from cpp.environment.environment import Environment
00010
00011 class ServerComm:
00012     """
00013     @brief Responsável pela comunicação com servidor.
00014     """
00015
00016     def __init__(self, creation_options: list[list[str]], environment: Environment, other_players:
00017 list):
00018         """
00019         @brief Construtor da classe, inicializando buffers e a conexão de cada agente com servidor.
00020         @param creation_options Lista de parâmetros de criação, self ainda não foi incluído na lista.
00021         @param environment
00022         @param other_players
00023         """
00024
00025         # Características da comunicação
00026         self.buffer_size = 4096 # Posteriormente, devemos analisar se realmente vale a pena ter um
00027         buffer com este comprimento
00028         self.buffer = bytearray(self.buffer_size)
00029         self.socket = socket.socket(
00030             socket.AF_INET,
00031             socket.SOCK_STREAM # TCP
00032         )
00033         self.socket.settimeout(2)
00034
00035         # Características alheias
00036         self.message_queue = []
00037         self.unum = creation_options[4][1]
00038         self.environment = environment
00039
00040         # Fazemos a conexão com servidor
00041         Printing.print_message(f"Tentando conexão do jogador {self.unum}", "info")
00042         while True:
00043             try:

```



```

00042
00043         self.socket.connect(
00044             (
00045                 creation_options[0][1], # Host
00046                 creation_options[1][1] # Porta de Agentes
00047             )
00048         )
00049         break
00050     except ConnectionRefusedError:
00051         sleep(1)
00052         Printing.print_message(".")
00053
00054     Printing.print_message("\tAgente Conectado!\n", "info")
00055
00056     # Fazemos o pedido de criação de robô
00057     self.send_immediate(
00058         f"(scene rsg/agent/nao/nao_hetero.rsg {creation_options[5][1]})".encode()
00059     )
00060     self.__receive_async(other_players)
00061     self.send_immediate(
00062         f"(init (unum {self.unum}) (teamname {creation_options[3][1]}))".encode()
00063     )
00064     self.__receive_async(other_players)
00065     Printing.print_message(f"Jogador {self.unum} recebeu do servidor assincronamente\n", "info")
00066
00067     # Aqui podem ser realizados testes de execução de quaisquer funções do ServerComm
00068
00069     for _ in range(3):
00070         self.send_immediate(b'(syn)')
00071         for p in other_players:
00072             p.scom.send_immediate(b'(syn)')
00073         for p in other_players:
00074             p.scom.receive()
00075         self.receive()
00076
00077
00078     # self.close()
00079
00080     # Métodos Mínimos da Classe de Comunicação com servidor
00081     def send_immediate(self, message: bytes) -> None:
00082         """
00083         @brief Envia uma mensagem instantânea ao servidor, verificando se a conexão continua ativa
00084         @param message String em forma de bytes para ser transmitida
00085         @details
00086         Coloca-se na frente uma informação de tamanho da mensagem dentro de 4 bytes.
00087         """
00088
00089         try:
00090             self.socket.send(
00091                 len(message).to_bytes(4, byteorder="big") + message
00092             )
00093         except BrokenPipeError:
00094             Printing.print_message("Error: socket foi fechado por rcssserver3d", "error")
00095
00096     def receive(self) -> None:
00097         """
00098         @brief Receberá informações diretamente do servidor, fazendo todas as verificações
00099         necessárias.
00100         """
00101
00102         msg_size = None
00103         while True:
00104             try:
00105                 # Verificamos se há 4 bytes no cabeçalho e nos preparamos para ler.
00106                 if self.socket.recv_into(
00107                     self.buffer, nbytes=4
00108                 ) != 4:
00109                     raise ConnectionResetError
00110
00111                 # Lemos o comprimento total da mensagem
00112                 msg_size = int.from_bytes(
00113                     self.buffer[:4], # Garantimos leitura de apenas 4 bytes
00114                     byteorder="big", # ordem de significativo
00115                     signed=False # se tem sinal
00116                 )
00117
00118                 # Lemos o restante da mensagem
00119                 if(
00120                     self.socket.recv_into(
00121                         self.buffer,
00122                         nbytes=msg_size
00123                     )
00124                 ) != msg_size:
00125                     raise ConnectionResetError
00126
00127             except ConnectionResetError:
00128                 Printing.print_message("\nError: socket foi fechado pelo rcssserver3d.", "error")

```

```

00128         exit()
00129
00130     except TimeoutError:
00131         pass
00132
00133     if len(
00134         select( # Monitora sockets/arquivos para I/O
00135             [self.socket], # Lista de sockets/arquivos para verificar leitura
00136             [], # Lista vazia para escrita
00137             [], # Lista vazia para exceções
00138             0.0 # timeout zero (não bloqueante)
00139         )[0] # Pegamos o primeiro socket para leitura
00140     ) == 0: # Logo, não há dados disponíveis para leitura
00141         break
00142
00143     # Como há algo para ser lido, devemos aplicar o parser
00144     self.environment.update_from_server(self.buffer[:msg_size])
00145
00146 def __receive_async(self, other_players: list) -> None:
00147     """
00148     @brief Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de
00149     execução
00150     @details
00151     Essa função foi criada com o único propósito de impedir que a espera por resposta
00152     do servidor interrompa o fluxo de execução. Não deve ser executada posteriormente.
00153     @param other_players Lista de jogadores de mesmo time presentes na partida
00154     """
00155
00156     # Caso não haja ninguém além dele
00157     if not other_players:
00158         # Sem isso, um loop infinito existiria
00159         return self.receive()
00160
00161     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00162     self.socket.setblocking(False)
00163
00164     while True:
00165         try:
00166             Printing.print_message(".")
00167             self.receive()
00168             break
00169         except BlockingIOError:
00170             pass
00171
00172     # Força que todos estejam em condições
00173     for p in other_players:
00174         p.scom.send_immediate(b"(syn)")
00175
00176     for p in other_players:
00177         p.scom.receive()
00178
00179     # Voltamos ao padrão
00180     self.socket.setblocking(True)
00181     return None
00182
00183 def commit(self, message: bytes) -> None:
00184     """
00185     @brief Responsável por adicionar uma nova mensagem à fila de mensagens
00186     @param message String em bytes a ser adicionada à fila
00187     """
00188     assert isinstance(message, bytes), "Mensagem deve estar em bytes"
00189     self.message_queue.append(message)
00190
00191 def close(self) -> None:
00192     """
00193     @brief Responsável por fazer o encerramento dos canais de comunicação
00194     """
00195
00196     self.socket.close()
00197
00198 def send(self) -> None:
00199     """
00200     @brief Envia ao servidor todas as mensagens commitadas.
00201     """
00202     if len(
00203         select(
00204             [self.socket],
00205             [],
00206             [],
00207             0.0
00208         )[0]
00209     ) == 0:
00210         # Se não há nenhum socket para ler neste momento, enviarei
00211         self.message_queue.append(b"(syn)")
00212         self.send_immediate(b"".join(self.message_queue))
00213     else:

```

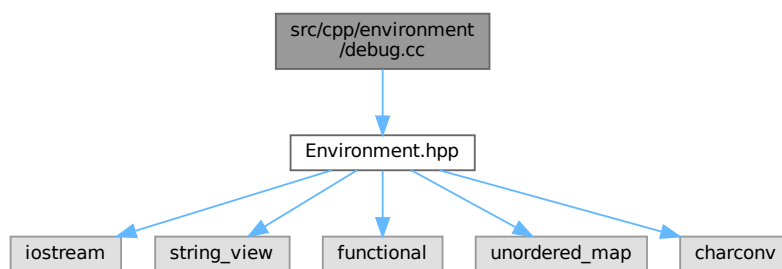
```

00214         Printing.print_message("\nHavia sockets de leitura disponíveis enquanto tentava enviar
        fila de mensagens commitadas.", "warning")
00215
00216         self.message_queue.clear() # Limpamos buffer
00217
00218     def clear_queue(self) -> None:
00219         """
00220         @brief Limpará a fila de commits.
00221         """
00222         self.message_queue.clear() # Assim usamos o mesmo ponteiro
00223
00224     # Métodos Derivados
00225     def commit_beam(self, vector_position2d: list, rotation: float):
00226         """
00227         @brief Comando de beam oficial do agente
00228         @param vector_position2d Sequência de dois valores, x e y finais do agente
00229         @param rotation Valor de rotação a ser dado ao robô
00230         """
00231         assert len(vector_position2d) == 2, "O beam oficial permite apenas posições 2D."
00232         self.commit(
00233             f"(beam {vector_position2d[0]} {vector_position2d[1]} {rotation})".encode()
00234         )
00235
00236
00237
00238
00239
00240
00241
00242
00243
00244
00245
00246

```

## 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 0) (sr 0) (t 0.00) (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 -1.67)) (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 0) (sr 0) (t 0.00) (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 -26.47 -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

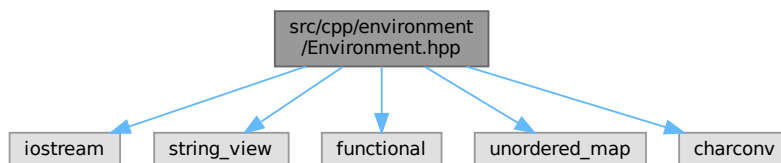
[Go to the documentation of this file.](#)

```
00001 #include "Environment.hpp"
00002
00003 const char* example = "(time (now 10.06))(GS (sl 0) (sr 0) (t 0.00) (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();
00014     ex.update_from_server(message_from_server);
00015
00016     return 0;
00017 }
```

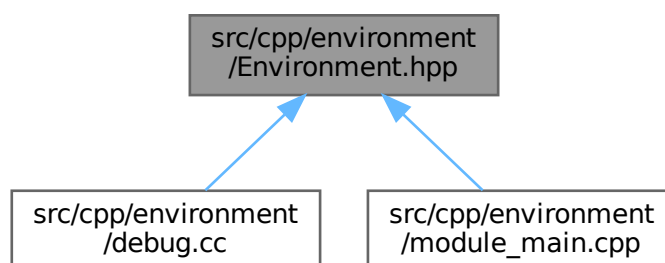
## 7.11 src/cpp/environment/Environment.hpp File Reference

```
#include <iostream>
#include <string_view>
#include <functional>
#include <unordered_map>
#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*

## Macros

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

### 7.11.1 Macro Definition Documentation

#### 7.11.1.1 False

```
#define False false
```

Definition at line 10 of file [Environment.hpp](#).

### 7.11.1.2 True

```
#define True true
```

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

## 7.12 Environment.hpp

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

```
00001 #pragma once
00002
00003 #include <iostream>
00004 #include <string_view>
00005 #include <functional>
00006 #include <unordered_map>
00007 #include <charconv>
00008
00009 #define True true
00010 #define False false
00011
00012 class Environment {
00013 public:
00014     /* Atributos Públicos de Ambiente */
00015     float time_server = 0;
00016     float time_match = 0;
00017
00018     enum class PlayMode : int {
00019         // Ao nosso favor
00020         OUR_KICKOFF = 0,
00021         OUR_KICK_IN = 1,
00022         OUR_CORNER_KICK = 2,
00023         OUR_GOAL_KICK = 3,
00024         OUR_FREE_KICK = 4,
00025         OUR_PASS = 5,
00026         OUR_DIR_FREE_KICK = 6,
00027         OUR_GOAL = 7,
00028         OUR_OFFSIDE = 8,
00029
00030         // Ao favor deles
00031         THEIR_KICKOFF = 9,
00032         THEIR_KICK_IN = 10,
00033         THEIR_CORNER_KICK = 11,
00034         THEIR_GOAL_KICK = 12,
00035         THEIR_FREE_KICK = 13,
00036         THEIR_PASS = 14,
00037         THEIR_DIR_FREE_KICK = 15,
00038         THEIR_GOAL = 16,
00039         THEIR_OFFSIDE = 17,
00040
00041         // Neutros
00042         BEFORE_KICKOFF = 18,
00043         GAME_OVER = 19,
00044         PLAY_ON = 20
00045     };
00046
00047     enum class PlayModeGroup : int {
00048         OUR_KICK = 0,
00049         THEIR_KICK = 1,
00050         ACTIVE_BEAM = 2,
00051         PASSIVE_BEAM = 3,
00052         OTHER = 4
00053     };
00054
00055     /* Métodos Inerentes a Execução da Aplicação */
00056
00057     int
00058     update_from_server(
00059         std::string_view msg
00060     ){
00061
00062         const char* buffer = msg.data();
00063         const char* end_buffer = buffer + msg.size();
00064         while (True) {
00065             while (*buffer != ' '){
00066                 buffer++;
00067                 if (buffer > end_buffer){ return 0; }
00068             }
00069         }
00070     }
00071 }
```



```

00081         }
00082         buffer++;
00083
00084         const char* tag_start = buffer;
00085         while(*buffer != ' '){ buffer++; }
00086
00087         std::string_view tag(tag_start, size_t(buffer - tag_start));
00088
00089         /* Passos seguintes concentram-se no parsing individual de cada informação */
00090         if(
00091             tag == "time"
00092         ){
00093             buffer += 6;
00094             const char* value_start = buffer;
00095             while(*buffer != ' '){ buffer++; }
00096             std::from_chars(value_start, buffer, this->time_server);
00097         }
00098         else if(
00099             tag == "GS"
00100         ){
00101             /* Vamos percorrer pelas possíveis tags que estiverem presentes */
00102             while(True){
00103                 while(*buffer != '('){ buffer++; }
00104                 buffer++;
00105                 const char* subtag_start = buffer;
00106                 while(*buffer != ' '){ buffer++; }
00107                 std::string_view subtag(subtag_start, size_t(buffer - subtag_start));
00108
00109                 if(
00110                     tag == "sl"
00111                 ){
00112                     }
00113                 }
00114
00115                 while(*buffer != ')'){ buffer++; }
00116                 if( *(buffer + 1) == ' ' ){ buffer += 2; break; }
00117             }
00118         }
00119         else{
00120             // Tag Desconhecida
00121         }
00122     }
00123
00124     return 1;
00125 }
00126 };
00127
00128
00129
00130
00131
00132
00133
00134
00135
00136
00137
00138

```

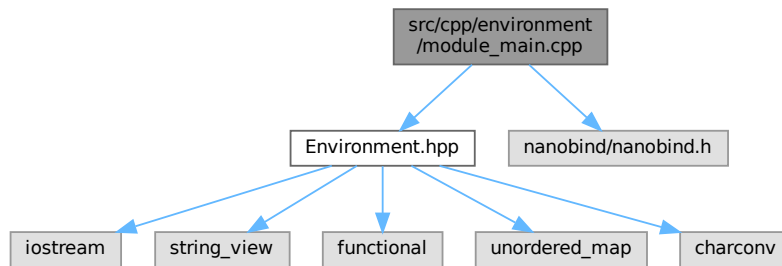
## 7.13 src/cpp/environment/module\_main.cpp File Reference

```

#include "Environment.hpp"
#include <nanobind/nanobind.h>

```

Include dependency graph for module\_main.cpp:



## Functions

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

### 7.13.1 Function Documentation

#### 7.13.1.1 NB\_MODULE()

```

NB_MODULE (
    environment ,
    m )

```

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

## 7.14 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>(m, "Environment")
00011         .def(nb::init<>())
00012         .def(
00013             "update_from_server",
00014             // A função anônima é apenas para convertermos os tipos
00015             [](
00016                 Environment &self,
00017                 const nb::bytearray& b
00018             ){
00019                 return self.update_from_server(std::string_view(reinterpret_cast<const
00020 char*>(b.data()), b.size()));
00021             },
00022             "doc"
00023         );
00024 }

```

## 7.15 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](#)

## 7.16 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(30)
00023
00024 for p in players:
00025     p.scom.close()
```

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

### Namespaces

- namespace [run\\_player](#)

### Variables

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

## 7.18 run\_player.py

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

```
00001 from term.Booting import Booting
00002 # from agent.Agent import Agent
00003
00004 boot = Booting()
00005
00006 # player = Agent(boot.options)
```

## 7.19 src/term/Booting.py File Reference

Implementação do [Booting](#) do time.

### Classes

- class [Booting.Booting](#)

*Responsável por inicializar todas as necessidades de execução do time.*

### Namespaces

- namespace [Booting](#)

### 7.19.1 Detailed Description

Implementação do [Booting](#) do time.

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

## 7.20 Booting.py

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

```

00001 """
00002 @file Booting.py
00003 @brief Implementação do Booting 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 Booting:
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
00068         config_team_params = [
00069             ["IP Server", "localhost"],
00070             ["Agent Port", 3100], # Onde nos conectaremos com rcssserver3d
00071             ["Monitor Port", 3200], # Onde nos conectaremos com Roboviz
00072             ["Team Name", "RoboIME"],
00073             ["Uniform Number", 1],
00074             ["Robot Type", 1],
00075             ["Penalty Shootout", 0],
00076             ["MagmaFatProxy", 0],
00077             ["Debug Mode", 1]
00078         ]
00079
00080         # E criamos o arquivo
00081         with open(
00082             Booting.CONFIG_PATH,
00083             "w+"
00084         ) as file_team_params:
00085             for doc, value in config_team_params:
00086                 file_team_params.write(
00087                     f"{doc},{value}\n"
00088                 )
00089
00090         return config_team_params
00091
00092     @staticmethod
00093     def show_spinner(
00094         running_flag: list[bool]
00095     ) -> None:
00096         """
00097         @brief Por motivos estéticos, mostrará um spinner enquanto há o carregamento de módulos C++
00098         """
00099
00100         spinner = ['|', '/', '-', '\\']
00101         i = 0
00102         while running_flag[0] and i < 1000:
00103             print(f"{spinner[i % len(spinner)]}", end="", flush=True)
00104             i += 1
00105             sleep(0.5)
00106             print("\b", end="")
00107
00108     @staticmethod
00109     def cpp_builder() -> None:
00110         """
00111         @brief Responsável por buildar os arquivos .cpp presentes na pasta cpp.
00112         @return Funcionalidades C++ em condições de interoperabilidade.
00113         """
00114
00115         # Vamos verificar quais arquivos .cpp estão disponíveis para buildar
00116         cpp_path = Path(__file__).resolve().parents[1] / "cpp"
00117         cpp_modules = [
00118             module for module in os.listdir(
00119                 cpp_path
00120             ) if os.path.isdir(os.path.join(cpp_path, module))
00121         ]
00122
00123         if not cpp_modules:
00124             return None # Não há nenhum para construirmos
00125
00126         # Servirá para verificarmos quais binários estão atualizados com a versão
00127         python_cmd = f"python{sys.version_info.major}.{sys.version_info.minor}"
00128
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         print("\033[1;7m/* ---- Construção de Funcionalidades C++ ---- */\033[0m")
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 + 30 > code_mod_time:
00185                             continue
00186
00187             msg = f"\033[1;7mConstruindo: \033[32;40m{cpp_module}\033[0m"
00188             print(f"{msg:.<{60}}", end="", flush=True)
00189
00190             processo = subprocess.Popen(
00191                 command_chain,
00192                 cwd=cpp_module_path,
00193                 stdout=subprocess.PIPE,
00194                 stderr=subprocess.PIPE,
00195                 text=False
00196             )
00197
00198             # Iniciamos thread de spinner
00199             running_flag = [True]
00200             worker = threading.Thread(target=Booting.show_spinner, args=(running_flag,))
00201             worker.start()
00202
00203             output, error = processo.communicate()
00204             return_code = processo.wait()
00205
00206             running_flag[0] = False
00207             worker.join()
00208
00209             if return_code == 0:
00210                 print("\033[7m\033[1mSucesso\033[0m")
00211
00212                 # Podemos construir um arquivo de fiscalização
00213                 with open(
00214                     os.path.join(cpp_module_path, f"{cpp_module}.cpp_info"),
00215                     "wb"
00216                 ) as f:

```

```

00216             # noinspection PyTypeChecker
00217             pickle.dump(python_cmd, f)
00218         else:
00219             Printing.print_message("Abortando", "error")
00220             print(output.decode(), error.decode())
00221             exit()
00222
00223         subprocess.run(
00224             ["make", "clean"],
00225             stdout=subprocess.PIPE,
00226             stderr=subprocess.PIPE,
00227             cwd=cpp_module_path
00228         )
00229
00230         return None

```

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

Implementação de Interface no terminal.

### Classes

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

### Namespaces

- namespace [Printing](#)

### 7.21.1 Detailed Description

Implementação de Interface no terminal.

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

## 7.22 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             finally:
00152                 # Restaura configurações originais do terminal
00153                 # Garante que o terminal volta ao normal mesmo com erros
00154                 termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
00155
00156         try:
00157             return return_type(buffer)
00158         except (ValueError, TypeError):
00159             Printing.print_message("Erro de entrada!", "error")
00160             return None
00161
00162
00163
00164
00165
00166
00167
00168
00169

```

## 7.23 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()`

### 7.23.1 Detailed Description

Implementação de lógica organizadora de posições iniciais de partida.

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

## 7.24 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),
00075                                     (2, 2)]}
00076
00077     @staticmethod
00078     def save_config_positions(dados: dict[str, list[tuple]]) -> None:
00079         """
00080         @brief Responsável por salvar uma estrutura de dados em arquivo binário
00081         @param dados Estrutura de dados a ser salva
00082         """
00083
00084         with open(
00085             RobotPositionManager.CONFIG_POSITION_PATH,
00086             "wb"
00087         ) as f:
00088             # Colocamos esse comentário já que estava dando erro no interpretador da IDE
00089             pickle.dump(dados, f) # type: ignore
00090
00091     def _field_to_canvas(self, fx_: float, fy_: float) -> tuple:
00092         """
00093         @brief Responsável por converter coordenadas do campo para pixels no canvas
00094         @param fx_ Coordenada real em x
00095         @param fy_ Coordenada real em y
00096         @return Coordenadas corrigidas para o grid
00097         """
00098         return (
00099             (fx_ - self.X_MIN) * self.GRID_SCALE,
00100             (self.Y_MAX - fy_) * self.GRID_SCALE
00101         )
00102
00103     def _canvas_to_field(self, cx: int, cy: int) -> tuple:
00104         """
00105         @brief Converterá o pixel clicado para o quadrado correspondente
00106         @param cx Posição X do pixel
00107         @param cy Posição Y do pixel
00108         @return tupla de posições reais
00109         """
00110
00111         # Converte pixel X para coordenada de campo
00112         fx_raw = (cx / self.GRID_SCALE) + self.X_MIN
00113
00114         # Converte pixel Y para coordenada de campo (invertendo a lógica)
00115         fy_raw = self.Y_MAX - (cy / self.GRID_SCALE)
00116
00117         # Arredonda para o 0.5 mais próximo
00118         fx_rounded = round(fx_raw * 2) / 2
00119         fy_rounded = round(fy_raw * 2) / 2
00120
00121         # Garante que o clique (mesmo fora) se encaixe nos limites
00122         return (
00123             max(self.X_MIN, min(self.X_MAX, fx_rounded)),
00124             max(self.Y_MIN, min(self.Y_MAX, fy_rounded))
00125         )
00126
00127     # -- Métodos de Interface
00128     def criar_widgets(self):
00129         """
00130         @brief Disporá os widgets da interface de forma inteligente, provendo informações úteis.
00131         """
00132
00133         upper_frame = ttk.Frame(self)
00134         upper_frame.pack(side="top", fill="x", padx=10, pady=10)
00135
00136         config_frame = ttk.Frame(upper_frame)
00137         config_frame.pack(side="left", fill="both", expand=True)
00138
00139         # Dispostemos a tabela
00140         self.tv_configs = ttk.Treeview(config_frame, columns=("Nome", "Configuração"),
00141                                     show="headings")
00142         self.tv_configs.heading("Nome", text="Nome")
00143         self.tv_configs.heading("Configuração", text="Configuração")
00144         self.tv_configs.column("Nome", width=50, anchor="center")
00145         self.tv_configs.column("Configuração", width=250)
00146
00147         self.tv_configs.pack(side="left", fill="both", expand=True)
00148         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",
00152     pady=2)
00153     ttk.Button(frame_botoes, text="Salvar Atual", command=self.salvar_config).pack(fill="x",
00154     pady=2)
00155     ttk.Button(frame_botoes, text="Apagar Seleccionada", command=self.apagar_config).pack(fill="x",
00156     pady=2)
00157     ttk.Button(frame_botoes, text="Limpar Grade", command=lambda: (self.clear_grid(),
00158     self.posicoes_atuais.clear())).pack(fill="x", pady=10)
00159
00160     # ----- Focando no campo
00161     frame_grid = ttk.Frame(self)
00162     frame_grid.pack(side="top", fill="both", expand=True, padx=10, pady=10)
00163
00164     # Canvas para o campo
00165     self.canvas = tk.Canvas(
00166         frame_grid,
00167         width=self.canvas_width,
00168         height=self.canvas_height,
00169         bg="#42f545" # Verde para o campo
00170     )
00171     self.canvas.pack()
00172
00173     # Bind do clique no canvas
00174     self.canvas.bind("<Button-1>", self.click_on_gridclick_on_grid)
00175
00176     self.clear_grid()
00177
00178     def draw_player(self, field_x, field_y) -> None:
00179         """
00180         @brief Desenharemos um jogador na posição especificada
00181         @param field_x Posição real em X
00182         @param field_y Posição real em Y
00183         """
00184
00185         # Converte as coordenadas do campo (ex: -14, 0) para pixels
00186         cx, cy = self._field_to_canvas(field_x, field_y)
00187
00188         r = self.GRID_SCALE / 3
00189
00190         oval_id = self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r,
00191         fill="yellow", outline="black", width=2)
00192
00193         self.marcadores_jogadores.append((oval_id, (field_x, field_y)))
00194
00195     # -- Métodos de Interação
00196     def click_on_grid(self, event: tk.Event):
00197         """
00198         @brief Responsável por identificar onde o usuário clicou e adicionar essa posição na lista
00199         @param event Argumento default do bind
00200         """
00201
00202         new_pos = self._canvas_to_field(event.x, event.y)
00203
00204         # Verificamos se clicamos em cima de um jogador
00205         for i, (oval_id, pos) in enumerate(self.marcadores_jogadores):
00206             if pos == new_pos:
00207                 self.canvas.delete(oval_id)
00208                 self.marcadores_jogadores.pop(i)
00209                 self.posicoes_atuais.remove(new_pos)
00210                 return
00211
00212         # Verificamos se o limite de jogadores foi atingido
00213         if len(self.posicoes_atuais) >= self.MAX_JOGADORES:
00214             messagebox.showwarning("Limite Atingido",
00215             f"Não é possível adicionar mais de {self.MAX_JOGADORES}
00216             jogadores.\n"
00217             "Clique em um jogador existente para removê-lo.")
00218
00219         return
00220
00221         # Caso nenhuma das opções anteriores, adicionamos
00222         self.posicoes_atuais.append(new_pos)
00223         self.draw_player(*new_pos)
00224
00225     def on_double_click_in_configs(self, event: tk.Event) -> None:
00226         """
00227         @brief Responsável por plotar a configuração de jogadores selecionada
00228         @param event Argumento Default de bind
00229         """
00230
00231         item_selecionado = self.tv_configs.focus()
00232         if not item_selecionado:
00233             return
00234

```

```

00229         nome_config = self.tv_configs.item(item_selecionado, "values")[0]
00230     if nome_config in self.config_positions:
00231         self.posicoes_atuais = self.config_positions[nome_config][:]
00232         self.clear_grid()
00233         for (fx, fy) in self.posicoes_atuais:
00234             self.draw_player(fx, fy)
00235         self.nome_de_config_selecionada = nome_config
00236     else:
00237         messagebox.showwarning("Erro", f"Configuração '{nome_config}' não encontrada.")
00238
00239     def salvar_config(self) -> None:
00240         """
00241         @brief Salvará uma configuração selecionada
00242         """
00243
00244         item_selecionado = self.tv_configs.focus()
00245         if not item_selecionado:
00246             if not self.nome_de_config_selecionada:
00247                 messagebox.showwarning("Inválido", "Não há selecionado")
00248                 return
00249             else:
00250                 nome_config = self.nome_de_config_selecionada
00251         else:
00252             nome_config = self.tv_configs.item(item_selecionado, "values")[0]
00253
00254         if messagebox.askyesno(
00255             "Certeza?",
00256             f"Realmente deseja salvar a configuração de jogadores presentes na grade em
(nome_config)?"
00257         ):
00258             # Atualizaremos
00259             self.config_positions[nome_config] = self.posicoes_atuais.copy()
00260             self.update_table_config()
00261             for item in self.tv_configs.get_children():
00262                 if self.tv_configs.item(item, 'values')[0] == nome_config: # [0] = primeira coluna
00263                     self.tv_configs.selection_set(item)
00264                     self.nome_de_config_selecionada = nome_config
00265                     break
00266
00267     def clear_grid(self) -> None:
00268         """
00269         @brief Responsável por limpar as posições e a grade
00270         """
00271
00272         self.canvas.delete("all")
00273         self.marcadores_jogadores = []
00274
00275         # Círculo central (usando a conversão de coordenadas)
00276         cx, cy = self._field_to_canvas(0,0)
00277         r = self.GRID_SCALE * 4 # Raio de 4 unidades
00278         self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r, outline="white", width=2)
00279
00280         # --- Desenhando Linhas da Grade (Quadrados) ---
00281
00282         # Total de passos de 0.5
00283         n_steps_x = int((self.FIELD_WIDTH * 2) + 1)
00284         n_steps_y = int((self.FIELD_HEIGHT * 2) + 1)
00285
00286         # Linhas Verticais (eixo X)
00287         for i in range(n_steps_x):
00288             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 = simpledialog.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](#)
    - `Bootling.Bootling`, [20](#)
    - `RobotPositionManager.RobotPositionManager`, [32](#)
    - `ServerComm.ServerComm`, [40](#)
  - `__receive_async`
    - `ServerComm.ServerComm`, [41](#)
  - `_canvas_to_field`
    - `RobotPositionManager.RobotPositionManager`, [32](#)
  - `_field_to_canvas`
    - `RobotPositionManager.RobotPositionManager`, [32](#)
- `ACTIVE_BEAM`
  - `Environment`, [24](#)
- `Agent`, [9](#)
- `Agent.Agent`, [13](#)
  - `__init__`, [14](#)
  - `unum`, [15](#)
- `AgentPenalty`, [9](#)
- `AGENTS_IN_THE_MATCH`
  - `BaseAgent.BaseAgent`, [18](#)
- `apagar_config`
  - `RobotPositionManager.RobotPositionManager`, [33](#)
- `BaseAgent`, [9](#)
- `BaseAgent.BaseAgent`, [15](#)
  - `__init__`, [18](#)
  - `AGENTS_IN_THE_MATCH`, [18](#)
  - `beam`, [18](#)
  - `environment`, [18](#)
  - `init_position`, [18](#)
  - `INITIAL_POSITION`, [19](#)
  - `scom`, [19](#)
  - `unum`, [19](#)
- `beam`
  - `BaseAgent.BaseAgent`, [18](#)
- `BEFORE_KICKOFF`
  - `Environment`, [23](#)
- `boot`
  - `run_full_team`, [10](#)
  - `run_player`, [11](#)
- `Bootling`, [9](#)
- `Bootling.Bootling`, [19](#)
  - `__init__`, [20](#)
  - `CONFIG_PATH`, [21](#)
  - `cpp_builder`, [21](#)
  - `get_team_params`, [21](#)
  - `options`, [21](#)
  - `show_spinner`, [21](#)
- `buffer`
  - `ServerComm.ServerComm`, [43](#)
- `buffer_size`
  - `ServerComm.ServerComm`, [43](#)
- `canvas`
  - `RobotPositionManager.RobotPositionManager`, [36](#)
- `canvas_height`
  - `RobotPositionManager.RobotPositionManager`, [36](#)
- `canvas_width`
  - `RobotPositionManager.RobotPositionManager`, [36](#)
- `clear_grid`
  - `RobotPositionManager.RobotPositionManager`, [33](#)
- `clear_queue`
  - `ServerComm.ServerComm`, [41](#)
- `click_on_grid`
  - `RobotPositionManager.RobotPositionManager`, [33](#), [36](#)
- `close`
  - `ServerComm.ServerComm`, [41](#)
- `commit`
  - `ServerComm.ServerComm`, [41](#)
- `commit_beam`
  - `ServerComm.ServerComm`, [42](#)
- `CONFIG_PATH`
  - `Bootling.Bootling`, [21](#)
- `CONFIG_POSITION_PATH`
  - `RobotPositionManager.RobotPositionManager`, [36](#)
- `config_positions`
  - `RobotPositionManager.RobotPositionManager`, [36](#)
- `CONSOLE`
  - `Printing.Printing`, [28](#)
- `cpp_builder`
  - `Bootling.Bootling`, [21](#)
- `criar_widgets`
  - `RobotPositionManager.RobotPositionManager`, [34](#)
- `debug.cc`
  - `example`, [52](#)
  - `example1`, [53](#)
  - `main`, [52](#)
  - `size`, [53](#)
  - `size1`, [53](#)
- `destroy`
  - `RobotPositionManager.RobotPositionManager`, [34](#)
- `draw_player`
  - `RobotPositionManager.RobotPositionManager`, [34](#)
- `Environment`, [22](#)
  - `ACTIVE_BEAM`, [24](#)

- BEFORE\_KICKOFF, [23](#)
- GAME\_OVER, [23](#)
- OTHER, [24](#)
- OUR\_CORNER\_KICK, [23](#)
- OUR\_DIR\_FREE\_KICK, [23](#)
- OUR\_FREE\_KICK, [23](#)
- OUR\_GOAL, [23](#)
- OUR\_GOAL\_KICK, [23](#)
- OUR\_KICK, [24](#)
- OUR\_KICK\_IN, [23](#)
- OUR\_KICKOFF, [23](#)
- OUR\_OFFSIDE, [23](#)
- OUR\_PASS, [23](#)
- PASSIVE\_BEAM, [24](#)
- PLAY\_ON, [23](#)
- PlayMode, [23](#)
- PlayModeGroup, [23](#)
- THEIR\_CORNER\_KICK, [23](#)
- THEIR\_DIR\_FREE\_KICK, [23](#)
- THEIR\_FREE\_KICK, [23](#)
- THEIR\_GOAL, [23](#)
- THEIR\_GOAL\_KICK, [23](#)
- THEIR\_KICK, [24](#)
- THEIR\_KICK\_IN, [23](#)
- THEIR\_KICKOFF, [23](#)
- THEIR\_OFFSIDE, [23](#)
- THEIR\_PASS, [23](#)
- time\_match, [25](#)
- time\_server, [25](#)
- update\_from\_server, [24](#)
- environment
  - BaseAgent.BaseAgent, [18](#)
  - ServerComm.ServerComm, [43](#)
- Environment.hpp
  - False, [55](#)
  - True, [55](#)
- example
  - debug.cc, [52](#)
- example1
  - debug.cc, [53](#)
- False
  - Environment.hpp, [55](#)
- FIELD\_HEIGHT
  - RobotPositionManager.RobotPositionManager, [37](#)
- FIELD\_WIDTH
  - RobotPositionManager.RobotPositionManager, [37](#)
- GAME\_OVER
  - Environment, [23](#)
- get\_config\_positions
  - RobotPositionManager.RobotPositionManager, [34](#)
- get\_input
  - Printing.Printing, [26](#)
- get\_team\_params
  - Booting.Booting, [21](#)
- GRID\_SCALE
  - RobotPositionManager.RobotPositionManager, [37](#)
- IF\_IN\_DEBUG
  - Printing.Printing, [28](#)
- init\_position
  - BaseAgent.BaseAgent, [18](#)
- INITIAL\_POSITION
  - BaseAgent.BaseAgent, [19](#)
- main
  - debug.cc, [52](#)
- marcadores\_jogadores
  - RobotPositionManager.RobotPositionManager, [37](#)
- MAX\_JOGADORES
  - RobotPositionManager.RobotPositionManager, [37](#)
- message\_queue
  - ServerComm.ServerComm, [43](#)
- module\_main.cpp
  - NB\_MODULE, [58](#)
- NB\_MODULE
  - module\_main.cpp, [58](#)
- nome\_de\_config\_selecionada
  - RobotPositionManager.RobotPositionManager, [37](#)
- nova\_config
  - RobotPositionManager.RobotPositionManager, [35](#)
- on\_double\_click\_in\_configs
  - RobotPositionManager.RobotPositionManager, [35](#), [37](#)
- options
  - Booting.Booting, [21](#)
- OTHER
  - Environment, [24](#)
- OUR\_CORNER\_KICK
  - Environment, [23](#)
- OUR\_DIR\_FREE\_KICK
  - Environment, [23](#)
- OUR\_FREE\_KICK
  - Environment, [23](#)
- OUR\_GOAL
  - Environment, [23](#)
- OUR\_GOAL\_KICK
  - Environment, [23](#)
- OUR\_KICK
  - Environment, [24](#)
- OUR\_KICK\_IN
  - Environment, [23](#)
- OUR\_KICKOFF
  - Environment, [23](#)
- OUR\_OFFSIDE
  - Environment, [23](#)
- OUR\_PASS
  - Environment, [23](#)
- p
  - run\_full\_team, [10](#)
- PASSIVE\_BEAM
  - Environment, [24](#)
- PLAY\_ON
  - Environment, [23](#)

- players
  - run\_full\_team, 10
- PlayMode
  - Environment, 23
- PlayModeGroup
  - Environment, 23
- posicoes\_atuais
  - RobotPositionManager.RobotPositionManager, 37
- print\_message
  - Printing.Printing, 26
- print\_table
  - Printing.Printing, 27
- Printing, 9
- Printing.Printing, 25
  - CONSOLE, 28
  - get\_input, 26
  - IF\_IN\_DEBUG, 28
  - print\_message, 26
  - print\_table, 27
  - TABLE\_COLORS, 28
- receive
  - ServerComm.ServerComm, 42
- RobotPositionManager, 10
  - root, 10
- RobotPositionManager.RobotPositionManager, 28
  - \_\_init\_\_, 32
  - \_canvas\_to\_field, 32
  - \_field\_to\_canvas, 32
  - apagar\_config, 33
  - canvas, 36
  - canvas\_height, 36
  - canvas\_width, 36
  - clear\_grid, 33
  - click\_on\_grid, 33, 36
  - CONFIG\_POSITION\_PATH, 36
  - config\_positions, 36
  - criar\_widgets, 34
  - destroy, 34
  - draw\_player, 34
  - FIELD\_HEIGHT, 37
  - FIELD\_WIDTH, 37
  - get\_config\_positions, 34
  - GRID\_SCALE, 37
  - marcadores\_jogadores, 37
  - MAX\_JOGADORES, 37
  - nome\_de\_config\_selecionada, 37
  - nova\_config, 35
  - on\_double\_click\_in\_configs, 35, 37
  - posicoes\_atuais, 37
  - salvar\_config, 35
  - save\_config\_positions, 35
  - tv\_configs, 38
  - update\_table\_config, 36
  - X\_MAX, 38
  - X\_MIN, 38
  - Y\_MAX, 38
  - Y\_MIN, 38
- root
  - RobotPositionManager, 10
  - run\_full\_team, 10
    - boot, 10
    - p, 10
    - players, 10
  - run\_player, 11
    - boot, 11
  - salvar\_config
    - RobotPositionManager.RobotPositionManager, 35
  - save\_config\_positions
    - RobotPositionManager.RobotPositionManager, 35
  - scom
    - BaseAgent.BaseAgent, 19
  - send
    - ServerComm.ServerComm, 42
  - send\_immediate
    - ServerComm.ServerComm, 42
  - ServerComm, 11
  - ServerComm.ServerComm, 39
    - \_\_init\_\_, 40
    - \_\_receive\_async, 41
    - buffer, 43
    - buffer\_size, 43
    - clear\_queue, 41
    - close, 41
    - commit, 41
    - commit\_beam, 42
    - environment, 43
    - message\_queue, 43
    - receive, 42
    - send, 42
    - send\_immediate, 42
    - socket, 43
    - unum, 43
  - show\_spinner
    - Bootling.Bootling, 21
  - size
    - debug.cc, 53
  - size1
    - debug.cc, 53
  - socket
    - ServerComm.ServerComm, 43
  - src/agent/Agent.py, 45
  - src/agent/AgentPenalty.py, 46
  - src/agent/BaseAgent.py, 46, 47
  - src/communication/ServerComm.py, 48
  - src/cpp/environment/debug.cc, 51, 54
  - src/cpp/environment/Environment.hpp, 54, 56
  - src/cpp/environment/module\_main.cpp, 57, 58
  - src/run\_full\_team.py, 59
  - src/run\_player.py, 59
  - src/term/Bootling.py, 60
  - src/term/Printing.py, 63
  - src/utils/RobotPositionManager.py, 65, 66
  - TABLE\_COLORS
    - Printing.Printing, 28
  - THEIR\_CORNER\_KICK

- Environment, [23](#)
- THEIR\_DIR\_FREE\_KICK
  - Environment, [23](#)
- THEIR\_FREE\_KICK
  - Environment, [23](#)
- THEIR\_GOAL
  - Environment, [23](#)
- THEIR\_GOAL\_KICK
  - Environment, [23](#)
- THEIR\_KICK
  - Environment, [24](#)
- THEIR\_KICK\_IN
  - Environment, [23](#)
- THEIR\_KICKOFF
  - Environment, [23](#)
- THEIR\_OFFSIDE
  - Environment, [23](#)
- THEIR\_PASS
  - Environment, [23](#)
- time\_match
  - Environment, [25](#)
- time\_server
  - Environment, [25](#)
- True
  - Environment.hpp, [55](#)
- tv\_configs
  - RobotPositionManager.RobotPositionManager, [38](#)
- unum
  - Agent.Agent, [15](#)
  - BaseAgent.BaseAgent, [19](#)
  - ServerComm.ServerComm, [43](#)
- update\_from\_server
  - Environment, [24](#)
- update\_table\_config
  - RobotPositionManager.RobotPositionManager, [36](#)
- X\_MAX
  - RobotPositionManager.RobotPositionManager, [38](#)
- X\_MIN
  - RobotPositionManager.RobotPositionManager, [38](#)
- Y\_MAX
  - RobotPositionManager.RobotPositionManager, [38](#)
- Y\_MIN
  - RobotPositionManager.RobotPositionManager, [38](#)