

SSRoboime

Generated by Doxygen 1.9.8

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 Agent Namespace Reference	9
5.2 AgentPenalty Namespace Reference	9
5.3 BaseAgent Namespace Reference	9
5.4 Booting Namespace Reference	9
5.5 exec_booting Namespace Reference	9
5.6 Printing Namespace Reference	9
5.7 RobotPositionManager Namespace Reference	10
5.7.1 Variable Documentation	10
5.7.1.1 root	10
5.8 RobotVision Namespace Reference	10
5.8.1 Detailed Description	10
5.8.2 Variable Documentation	11
5.8.2.1 HEIGHT	11
5.8.2.2 WIDTH	11
5.9 run_full_team Namespace Reference	11
5.9.1 Variable Documentation	11
5.9.1.1 boot	11
5.9.1.2 p	11
5.9.1.3 players	11
5.10 run_player Namespace Reference	12
5.10.1 Variable Documentation	12
5.10.1.1 boot	12
5.10.1.2 p	12
5.11 ServerComm Namespace Reference	12
6 Class Documentation	13
6.1 Agent.Agent Class Reference	13
6.1.1 Detailed Description	14
6.1.2 Constructor & Destructor Documentation	14
6.1.2.1 __init__()	14
6.1.3 Member Data Documentation	15

6.1.3.1 <code>unum</code>	15
6.2 <code>RobotVision.Ball</code> Class Reference	16
6.2.1 Detailed Description	18
6.2.2 Constructor & Destructor Documentation	18
6.2.2.1 <code>__init__()</code>	18
6.2.3 Member Function Documentation	18
6.2.3.1 <code>draw()</code>	18
6.2.3.2 <code>projection_to_2d()</code>	19
6.2.4 Member Data Documentation	19
6.2.4.1 <code>color</code>	19
6.2.4.2 <code>position_on_sphere</code>	19
6.2.4.3 <code>position_on_window</code>	19
6.3 <code>BaseAgent.BaseAgent</code> Class Reference	20
6.3.1 Detailed Description	22
6.3.2 Constructor & Destructor Documentation	22
6.3.2.1 <code>__init__()</code>	22
6.3.3 Member Function Documentation	22
6.3.3.1 <code>beam()</code>	22
6.3.4 Member Data Documentation	22
6.3.4.1 <code>AGENTS_IN_THE_MATCH</code>	22
6.3.4.2 <code>env</code>	22
6.3.4.3 <code>init_position</code>	23
6.3.4.4 <code>INITIAL_POSITION</code>	23
6.3.4.5 <code>logger</code>	23
6.3.4.6 <code>scom</code>	23
6.3.4.7 <code>unum</code>	23
6.4 <code>Booting.Booting</code> Class Reference	23
6.4.1 Detailed Description	24
6.4.2 Constructor & Destructor Documentation	24
6.4.2.1 <code>__init__()</code>	24
6.4.3 Member Function Documentation	25
6.4.3.1 <code>cpp_builder()</code>	25
6.4.3.2 <code>get_team_params()</code>	25
6.4.3.3 <code>show_spinner()</code>	25
6.4.4 Member Data Documentation	25
6.4.4.1 <code>CONFIG_PATH</code>	25
6.4.4.2 <code>options</code>	26
6.5 <code>RobotVision.Elemento</code> Class Reference	26
6.5.1 Detailed Description	27
6.5.2 Constructor & Destructor Documentation	27
6.5.2.1 <code>__init__()</code>	27
6.5.3 Member Function Documentation	28

6.5.3.1 draw()	28
6.5.3.2 project_point()	28
6.5.3.3 projection_to_2d()	28
6.5.4 Member Data Documentation	28
6.5.4.1 color	28
6.5.4.2 fov_h	28
6.5.4.3 fov_v	29
6.5.4.4 height	29
6.5.4.5 width	29
6.6 Environment::Enabler_Stringview_Hash Struct Reference	29
6.6.1 Detailed Description	30
6.6.2 Member Typedef Documentation	30
6.6.2.1 is_transparent	30
6.6.3 Member Function Documentation	30
6.6.3.1 operator() [1/2]	30
6.6.3.2 operator() [2/2]	30
6.7 Environment Class Reference	31
6.7.1 Detailed Description	33
6.7.2 Member Enumeration Documentation	33
6.7.2.1 PlayMode	33
6.7.2.2 PlayModeGroup	34
6.7.3 Constructor & Destructor Documentation	34
6.7.3.1 Environment()	34
6.7.4 Member Function Documentation	34
6.7.4.1 print_status()	34
6.7.4.2 update_from_server()	34
6.7.5 Member Data Documentation	35
6.7.5.1 current_mode	35
6.7.5.2 goals_conceded	35
6.7.5.3 goals_scored	35
6.7.5.4 is_left	35
6.7.5.5 logger	36
6.7.5.6 play_modes	36
6.7.5.7 time_match	36
6.7.5.8 time_server	36
6.7.5.9 unum	36
6.8 RobotVision.Goal Class Reference	37
6.8.1 Detailed Description	39
6.8.2 Constructor & Destructor Documentation	39
6.8.2.1 __init__()	39
6.8.3 Member Function Documentation	39
6.8.3.1 draw()	39

6.8.3.2 projection_to_2d()	40
6.8.4 Member Data Documentation	40
6.8.4.1 color	40
6.8.4.2 position_on_sphere	40
6.8.4.3 position_on_window	40
6.9 RobotVision.Line Class Reference	41
6.9.1 Detailed Description	43
6.9.2 Constructor & Destructor Documentation	43
6.9.2.1 __init__()	43
6.9.3 Member Function Documentation	43
6.9.3.1 draw()	43
6.9.3.2 projection_to_2d()	43
6.9.4 Member Data Documentation	44
6.9.4.1 color	44
6.9.4.2 position_on_sphere	44
6.9.4.3 position_on_window	44
6.10 Logger Class Reference	44
6.10.1 Detailed Description	46
6.10.2 Constructor & Destructor Documentation	47
6.10.2.1 Logger() [1/2]	47
6.10.2.2 Logger() [2/2]	47
6.10.2.3 ~Logger()	47
6.10.3 Member Function Documentation	47
6.10.3.1 __init_file()	47
6.10.3.2 _worker_loop()	47
6.10.3.3 error() [1/2]	48
6.10.3.4 error() [2/2]	48
6.10.3.5 get()	48
6.10.3.6 info() [1/2]	48
6.10.3.7 info() [2/2]	49
6.10.3.8 log()	49
6.10.3.9 operator=()	49
6.10.3.10 warn() [1/2]	49
6.10.3.11 warn() [2/2]	50
6.10.4 Member Data Documentation	51
6.10.4.1 _current_buffer	51
6.10.4.2 _cv	51
6.10.4.3 _file_stream	51
6.10.4.4 _is_running	51
6.10.4.5 _mutex	51
6.10.4.6 _worker	51
6.10.4.7 _write_buffer	52

6.10.4.8 <code>is_the_first</code>	52
6.11 <code>RobotVision.Marker</code> Class Reference	53
6.11.1 Detailed Description	55
6.11.2 Constructor & Destructor Documentation	55
6.11.2.1 <code>__init__()</code>	55
6.11.3 Member Function Documentation	55
6.11.3.1 <code>draw()</code>	55
6.11.3.2 <code>projection_to_2d()</code>	56
6.11.4 Member Data Documentation	56
6.11.4.1 <code>color</code>	56
6.11.4.2 <code>position_on_sphere</code>	56
6.11.4.3 <code>position_on_window</code>	56
6.12 <code>Environment::Parsing</code> Class Reference	57
6.12.1 Detailed Description	58
6.12.2 Constructor & Destructor Documentation	59
6.12.2.1 <code>Parsing()</code>	59
6.12.3 Member Function Documentation	59
6.12.3.1 <code>advance()</code>	59
6.12.3.2 <code>get()</code>	59
6.12.3.3 <code>get_str()</code>	60
6.12.3.4 <code>get_value()</code>	60
6.12.3.5 <code>is_valid()</code>	60
6.12.3.6 <code>parse_accelerometer()</code>	60
6.12.3.7 <code>parse_force_resistance()</code>	61
6.12.3.8 <code>parse_gamestate()</code>	61
6.12.3.9 <code>parse_gyroscope()</code>	61
6.12.3.10 <code>parse_hear()</code>	61
6.12.3.11 <code>parse_hingejoint()</code>	62
6.12.3.12 <code>parse_time()</code>	62
6.12.3.13 <code>parse_vision()</code>	62
6.12.3.14 <code>skip_until_char()</code>	62
6.12.4 Member Data Documentation	63
6.12.4.1 <code>buffer</code>	63
6.12.4.2 <code>end</code>	63
6.12.4.3 <code>env</code>	63
6.13 <code>Printing.Printing</code> Class Reference	64
6.13.1 Detailed Description	65
6.13.2 Member Function Documentation	65
6.13.2.1 <code>get_input()</code>	65
6.13.2.2 <code>print_message()</code>	65
6.13.2.3 <code>print_table()</code>	66
6.13.3 Member Data Documentation	66

6.13.3.1 CONSOLE	66
6.13.3.2 IF_IN_DEBUG	66
6.13.3.3 TABLE_COLORS	67
6.14 RobotPositionManager.RobotPositionManager Class Reference	67
6.14.1 Detailed Description	71
6.14.2 Constructor & Destructor Documentation	71
6.14.2.1 __init__()	71
6.14.3 Member Function Documentation	71
6.14.3.1 _canvas_to_field()	71
6.14.3.2 _field_to_canvas()	72
6.14.3.3 apagar_config()	72
6.14.3.4 clear_grid()	72
6.14.3.5 click_on_grid()	72
6.14.3.6 criar_widgets()	73
6.14.3.7 destroy()	73
6.14.3.8 draw_player()	73
6.14.3.9 get_config_positions()	73
6.14.3.10 nova_config()	74
6.14.3.11 on_double_click_in_configs()	74
6.14.3.12 salvar_config()	74
6.14.3.13 save_config_positions()	74
6.14.3.14 update_table_config()	75
6.14.4 Member Data Documentation	75
6.14.4.1 canvas	75
6.14.4.2 canvas_height	75
6.14.4.3 canvas_width	75
6.14.4.4 click_on_grid	75
6.14.4.5 CONFIG_POSITION_PATH	75
6.14.4.6 config_positions	76
6.14.4.7 FIELD_HEIGHT	76
6.14.4.8 FIELD_WIDTH	76
6.14.4.9 GRID_SCALE	76
6.14.4.10 marcadores_jogadores	76
6.14.4.11 MAX_JOGADORES	76
6.14.4.12 nome_de_config_selecionada	76
6.14.4.13 on_double_click_in_configs	76
6.14.4.14 posicoes_atuais	77
6.14.4.15 tv_configs	77
6.14.4.16 X_MAX	77
6.14.4.17 X_MIN	77
6.14.4.18 Y_MAX	77
6.14.4.19 Y_MIN	77

6.15 RobotVision.RobotVision Class Reference	78
6.15.1 Detailed Description	79
6.15.2 Constructor & Destructor Documentation	79
6.15.2.1 <code>__init__()</code>	79
6.15.3 Member Function Documentation	79
6.15.3.1 <code>_get_only_tag_See()</code>	79
6.15.3.2 <code>draw_legend()</code>	79
6.15.3.3 <code>load_frames_from_file()</code>	80
6.15.3.4 <code>mainloop()</code>	80
6.15.3.5 <code>parse_frame()</code>	80
6.15.4 Member Data Documentation	80
6.15.4.1 <code>current_index</code>	80
6.15.4.2 <code>frames</code>	80
6.15.4.3 <code>FRAMES_VISION_PATH</code>	80
6.15.4.4 <code>need_to_update</code>	81
6.15.4.5 <code>objects</code>	81
6.16 ServerComm.ServerComm Class Reference	81
6.16.1 Detailed Description	82
6.16.2 Constructor & Destructor Documentation	82
6.16.2.1 <code>__init__()</code>	82
6.16.3 Member Function Documentation	83
6.16.3.1 <code>__receive_async()</code>	83
6.16.3.2 <code>clear_queue()</code>	83
6.16.3.3 <code>close()</code>	83
6.16.3.4 <code>commit()</code>	83
6.16.3.5 <code>commit_beam()</code>	84
6.16.3.6 <code>receive()</code>	84
6.16.3.7 <code>send()</code>	84
6.16.3.8 <code>send_immediate()</code>	84
6.16.4 Member Data Documentation	85
6.16.4.1 <code>buffer</code>	85
6.16.4.2 <code>buffer_size</code>	85
6.16.4.3 <code>env</code>	85
6.16.4.4 <code>message_queue</code>	85
6.16.4.5 <code>socket</code>	85
6.16.4.6 <code>unum</code>	85
7 File Documentation	87
7.1 <code>src/agent/Agent.py</code> File Reference	87
7.1.1 Detailed Description	87
7.2 <code>Agent.py</code>	87
7.3 <code>src/agent/AgentPenalty.py</code> File Reference	88

7.3.1 Detailed Description	88
7.4 AgentPenalty.py	88
7.5 src/agent/BaseAgent.py File Reference	88
7.5.1 Detailed Description	89
7.6 BaseAgent.py	89
7.7 src/communication/ServerComm.py File Reference	90
7.7.1 Detailed Description	90
7.8 ServerComm.py	90
7.9 src/cpp/environment/debug.cc File Reference	93
7.9.1 Function Documentation	94
7.9.1.1 main()	94
7.9.2 Variable Documentation	94
7.9.2.1 example	94
7.9.2.2 example1	95
7.9.2.3 size	95
7.9.2.4 size1	95
7.10 debug.cc	96
7.11 src/cpp/logger/debug.cc File Reference	96
7.11.1 Function Documentation	97
7.11.1.1 main()	97
7.11.1.2 tarefaPesada()	97
7.12 debug.cc	97
7.13 src/cpp/environment/Environment.cpp File Reference	99
7.14 Environment.cpp	99
7.15 src/cpp/environment/Environment.hpp File Reference	100
7.15.1 Macro Definition Documentation	101
7.15.1.1 False	101
7.15.1.2 True	102
7.16 Environment.hpp	102
7.17 src/cpp/environment/module_main.cpp File Reference	107
7.17.1 Function Documentation	107
7.17.1.1 NB_MODULE()	107
7.18 module_main.cpp	107
7.19 src/cpp/logger/module_main.cpp File Reference	108
7.19.1 Function Documentation	109
7.19.1.1 NB_MODULE()	109
7.20 module_main.cpp	109
7.21 src/cpp/logger/Logger.hpp File Reference	110
7.21.1 Macro Definition Documentation	111
7.21.1.1 False	111
7.21.1.2 True	111
7.22 Logger.hpp	111

7.23 src/exec_booting.py File Reference	113
7.24 exec_booting.py	113
7.25 src/run_full_team.py File Reference	113
7.26 run_full_team.py	114
7.27 src/run_player.py File Reference	114
7.28 run_player.py	114
7.29 src/term/Bootning.py File Reference	114
7.29.1 Detailed Description	115
7.30 Booting.py	115
7.31 src/term/Printing.py File Reference	118
7.31.1 Detailed Description	118
7.32 Printing.py	118
7.33 src/utils/RobotPositionManager.py File Reference	120
7.33.1 Detailed Description	121
7.34 RobotPositionManager.py	121
7.35 src/utils/RobotVision.py File Reference	126
7.36 RobotVision.py	126
Index	133

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Agent	9
AgentPenalty	9
BaseAgent	9
Booting	9
exec_booting	9
Printing	9
RobotPositionManager	10
RobotVision	10
Implementação de Classe que nos permitirá ter a visão do robô	10
run_full_team	11
run_player	12
ServerComm	12

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Booting.Booting	23
RobotVision.Elemento	26
RobotVision.Ball	16
RobotVision.Goal	37
RobotVision.Line	41
RobotVision.Marker	53
Environment::Enabler_Stringview_Hash	29
Environment	31
Logger	44
Environment::Parsing	57
Printing.Printing	64
RobotVision.RobotVision	78
ServerComm.ServerComm	81
tk.Tk	
RobotPositionManager.RobotPositionManager	67
ABC	
BaseAgent.BaseAgent	20
BaseAgent	
Agent.Agent	13

Chapter 3

Class Index

3.1 Class List

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

Agent.Agent	Classe que representará os agentes de campo, possuindo métodos correspondentes	13
RobotVision.Ball		16
BaseAgent.BaseAgent	Classe que agrupará todas as funcionalidades comuns a qualquer agente	20
Booting.Booting	Responsável por inicializar todas as necessidades de execução do time	23
RobotVision.Elemento		26
Environment::Enabler_Stringview_Hash		29
Environment	Responsável por representar o ambiente externo ao robô	31
RobotVision.Goal		37
RobotVision.Line		41
Logger	Singleton para logging assíncrono	44
RobotVision.Marker		53
Environment::Parsing	Responsável por prover ferramentas de auxílio de parsing	57
Printing.Printing	Responsável pela comunicação usuário - terminal	64
RobotPositionManager.RobotPositionManager	Responsável por permitir ao usuário a criação de diversas formações táticas	67
RobotVision.RobotVision	Classe responsável por gerir a aplicação principal	78
ServerComm.ServerComm	Responsável pela comunicação com servidor	81

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

src/exec_booting.py	113
src/run_full_team.py	113
src/run_player.py	114
src/agent/Agent.py	
Implementação de Lógica de Agente de Campo	87
src/agent/AgentPenalty.py	
Implementação de Lógica de Goleiro	88
src/agent/BaseAgent.py	
Implementação da classe de jogador base, que deve ser comum a todos os agentes	88
src/communication/ServerComm.py	
Implementação da Comunicação com Servidor	90
src/cpp/environment/debug.cc	93
src/cpp/environment/Environment.cpp	99
src/cpp/environment/Environment.hpp	100
src/cpp/environment/module_main.cpp	107
src/cpp/logger/debug.cc	96
src/cpp/logger/Logger.hpp	110
src/cpp/logger/module_main.cpp	108
src/term/Booting.py	
Implementação do Booting do time	114
src/term/Printing.py	
Implementação de Interface no terminal	118
src/utils/RobotPositionManager.py	
Implementação de lógica organizadora de posições iniciais de partida	120
src/utils/RobotVision.py	126

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 exec_booting Namespace Reference

5.6 Printing Namespace Reference

Classes

- class [Printing](#)

Responsável pela comunicação usuário - terminal.

5.7 RobotPositionManager Namespace Reference

Classes

- class [RobotPositionManager](#)

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

Variables

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

5.7.1 Variable Documentation

5.7.1.1 root

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

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

5.8 RobotVision Namespace Reference

Implementação de Classe que nos permitirá ter a visão do robô

Classes

- class [Ball](#)
- class [Elemento](#)
- class [Goal](#)
- class [Line](#)
- class [Marker](#)
- class [RobotVision](#)

Classe responsável por gerir a aplicação principal.

Variables

- [WIDTH](#)
- [HEIGHT](#)

5.8.1 Detailed Description

Implementação de Classe que nos permitirá ter a visão do robô

5.8.2 Variable Documentation

5.8.2.1 HEIGHT

RobotVision.HEIGHT

Definition at line 8 of file [RobotVision.py](#).

5.8.2.2 WIDTH

RobotVision.WIDTH

Definition at line 8 of file [RobotVision.py](#).

5.9 run_full_team Namespace Reference

Variables

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

5.9.1 Variable Documentation

5.9.1.1 boot

run_full_team.boot = Booting()

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

5.9.1.2 p

Agent run_full_team.p

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

5.9.1.3 players

list run_full_team.players = []

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

5.10 run_player Namespace Reference

Variables

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

5.10.1 Variable Documentation

5.10.1.1 boot

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

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

5.10.1.2 p

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

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

5.11 ServerComm Namespace Reference

Classes

- class `ServerComm`
Responsável pela comunicação com servidor.

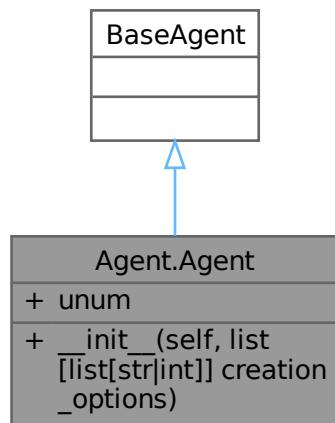
Chapter 6

Class Documentation

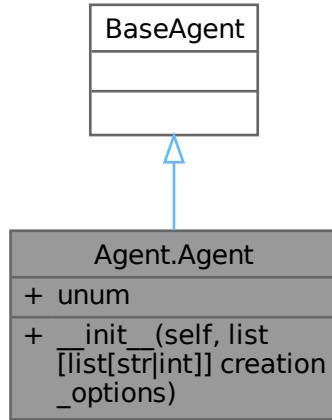
6.1 Agent.Agent Class Reference

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

<i>creation_options</i>	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 Penálti
- 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`

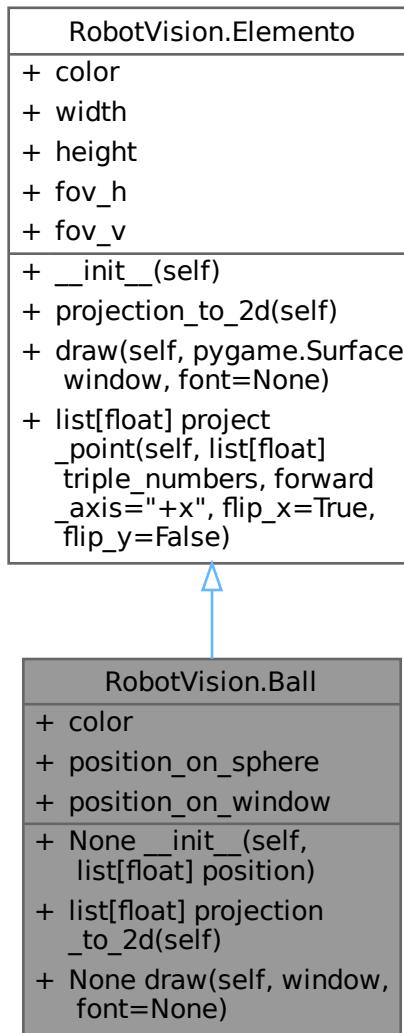
Definition at line 29 of file [Agent.py](#).

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

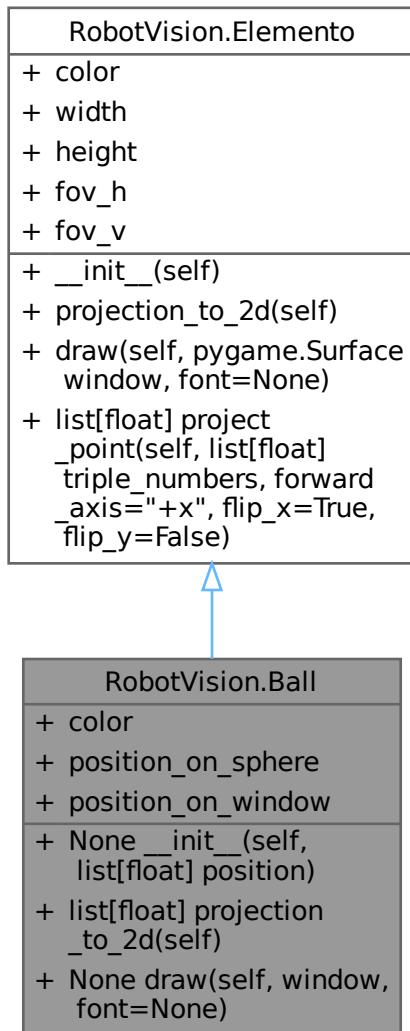
- `src/agent/Agent.py`

6.2 RobotVision.Ball Class Reference

Inheritance diagram for RobotVision.Ball:



Collaboration diagram for RobotVision.Ball:



Public Member Functions

- None [__init__](#) (self, list[float] position)
Construtor responsável por inicializar bola no e prover.
- list[float] [projection_to_2d](#) (self)
- None [draw](#) (self, window, font=None)

Public Member Functions inherited from [RobotVision.Elemento](#)

- list[float] [project_point](#) (self, list[float] triple_numbers, forward_axis="+x", flip_x=[True](#), flip_y=[False](#))

Public Attributes

- `color`
- `position_on_sphere`
- `position_on_window`

Public Attributes inherited from [RobotVision.Elemento](#)

- `color`
- `width`
- `height`
- `fov_h`
- `fov_v`

6.2.1 Detailed Description

Definition at line 68 of file [RobotVision.py](#).

6.2.2 Constructor & Destructor Documentation

6.2.2.1 `__init__()`

```
None RobotVision.Ball.__init__ (
    self,
    list[float] position )
```

Construtor responsável por inicializara bola no e prover.

Reimplemented from [RobotVision.Elemento](#).

Definition at line 70 of file [RobotVision.py](#).

6.2.3 Member Function Documentation

6.2.3.1 `draw()`

```
None RobotVision.Ball.draw (
    self,
    window,
    font = None )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 86 of file [RobotVision.py](#).

6.2.3.2 `projection_to_2d()`

```
list[float] RobotVision.Ball.projection_to_2d (
    self )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 81 of file [RobotVision.py](#).

6.2.4 Member Data Documentation

6.2.4.1 `color`

```
RobotVision.Ball.color
```

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

6.2.4.2 `position_on_sphere`

```
RobotVision.Ball.position_on_sphere
```

Definition at line 78 of file [RobotVision.py](#).

6.2.4.3 `position_on_window`

```
RobotVision.Ball.position_on_window
```

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

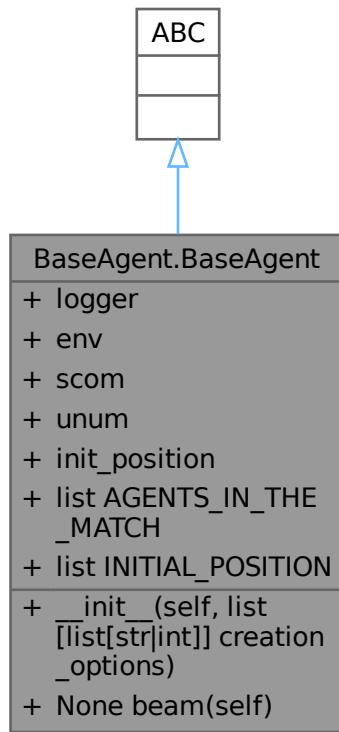
The documentation for this class was generated from the following file:

- src/utils/[RobotVision.py](#)

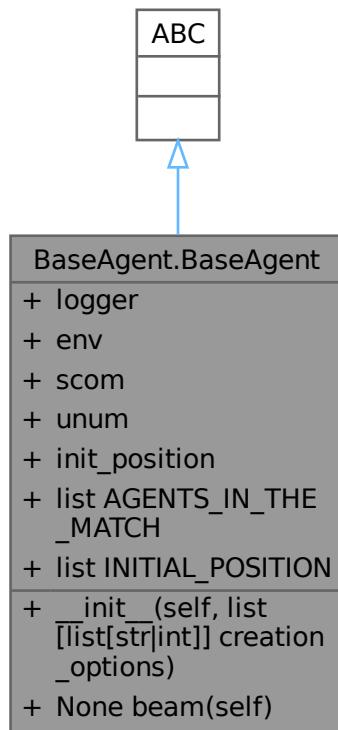
6.3 BaseAgent.BaseAgent Class Reference

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

Inheritance diagram for BaseAgent.BaseAgent:



Collaboration diagram for BaseAgent.BaseAgent:



Public Member Functions

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

Public Attributes

- logger
 - env
 - scom
 - unum
 - init position

Static Public Attributes

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

6.3.1 Detailed Description

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

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

6.3.2 Constructor & Destructor Documentation

6.3.2.1 `__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

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

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

6.3.3 Member Function Documentation

6.3.3.1 `beam()`

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

Responsável por gerenciar o teletransporte dos jogadores.

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

6.3.4 Member Data Documentation

6.3.4.1 `AGENTS_IN_THE_MATCH`

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

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

6.3.4.2 `env`

```
BaseAgent.BaseAgent.env
```

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

6.3.4.3 init_position

BaseAgent.BaseAgent.init_position

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

6.3.4.4 INITIAL_POSITION

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

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

6.3.4.5 logger

BaseAgent.BaseAgent.logger

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

6.3.4.6 scom

BaseAgent.BaseAgent.scom

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

6.3.4.7 unum

BaseAgent.BaseAgent.unum

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

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

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

6.4 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"`

6.4.1 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](#).

6.4.2 Constructor & Destructor Documentation

6.4.2.1 `__init__()`

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

Responsável por chamar as inicializações mínimas.

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

6.4.3 Member Function Documentation

6.4.3.1 cpp_builder()

```
None Booting.Booting.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 [Booting.py](#).

6.4.3.2 get_team_params()

```
list[list[str | int]] Booting.Booting.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 [Booting.py](#).

6.4.3.3 show_spinner()

```
None Booting.Booting.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 [Booting.py](#).

6.4.4 Member Data Documentation

6.4.4.1 CONFIG_PATH

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

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

6.4.4.2 options

`Booting.Booting.options`

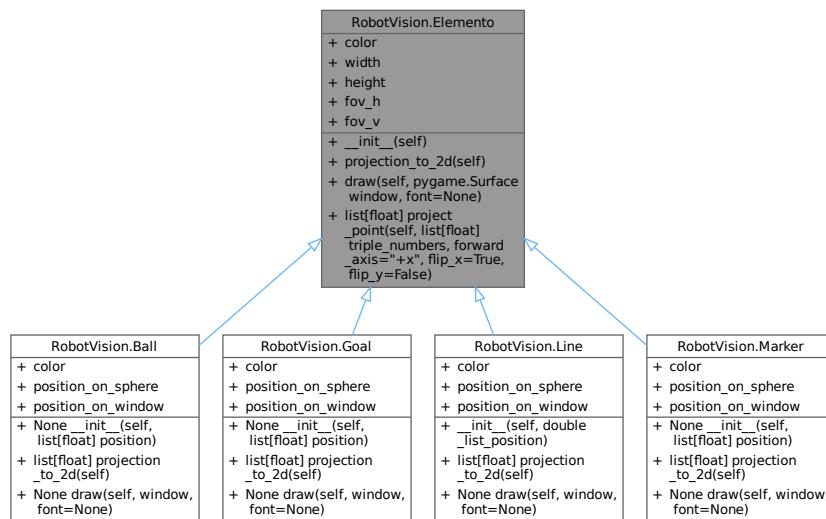
Definition at line 31 of file [Booting.py](#).

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

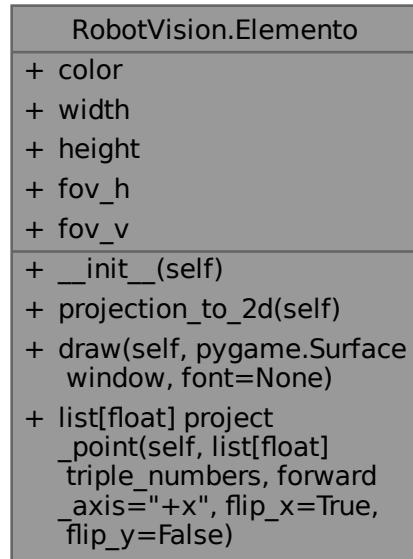
- [src/term/Booting.py](#)

6.5 RobotVision.Elemento Class Reference

Inheritance diagram for RobotVision.Elemento:



Collaboration diagram for RobotVision.Elemento:



Public Member Functions

- `__init__ (self)`
- `projection_to_2d (self)`
- `draw (self, pygame.Surface window, font=None)`
- `list[float] project_point (self, list[float] triple_numbers, forward_axis="+x", flip_x=True, flip_y=False)`

Public Attributes

- `color`
- `width`
- `height`
- `fov_h`
- `fov_v`

6.5.1 Detailed Description

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

6.5.2 Constructor & Destructor Documentation

6.5.2.1 __init__()

```
RobotVision.Elemento.__init__ (
    self )
```

Reimplemented in [RobotVision.Line](#), [RobotVision.Ball](#), [RobotVision.Marker](#), and [RobotVision.Goal](#).

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

6.5.3 Member Function Documentation

6.5.3.1 draw()

```
RobotVision.Elemento.draw (
    self,
    pygame.Surface window,
    font = None )
```

Reimplemented in [RobotVision.Ball](#), [RobotVision.Marker](#), [RobotVision.Goal](#), and [RobotVision.Line](#).

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

6.5.3.2 project_point()

```
list[float] RobotVision.Elemento.project_point (
    self,
    list[float] triple_numbers,
    forward_axis = "+x",
    flip_x = True,
    flip_y = False )
```

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

6.5.3.3 projection_to_2d()

```
RobotVision.Elemento.projection_to_2d (
    self )
```

Reimplemented in [RobotVision.Ball](#), [RobotVision.Marker](#), [RobotVision.Goal](#), and [RobotVision.Line](#).

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

6.5.4 Member Data Documentation

6.5.4.1 color

```
RobotVision.Elemento.color
```

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

6.5.4.2 fov_h

```
RobotVision.Elemento.fov_h
```

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

6.5.4.3 `fov_v`

`RobotVision.Elemento.fov_v`

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

6.5.4.4 `height`

`RobotVision.Elemento.height`

Definition at line 14 of file [RobotVision.py](#).

6.5.4.5 `width`

`RobotVision.Elemento.width`

Definition at line 14 of file [RobotVision.py](#).

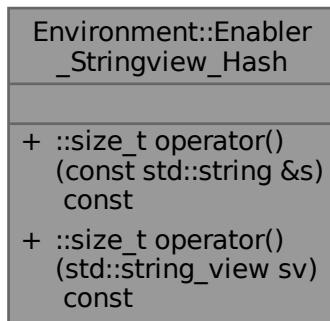
The documentation for this class was generated from the following file:

- [src/utils/RobotVision.py](#)

6.6 Environment::Enabler_Stringview_Hash Struct Reference

#include <Environment.hpp>

Collaboration diagram for Environment::Enabler_Stringview_Hash:



Public Types

- using `is_transparent` = void

Sinaliza ao unordered_map que essa struct suporta tipos heterogêneos para pesquisa.

Public Member Functions

- `::size_t operator()(const std::string &s) const`
- `::size_t operator()(std::string_view sv) const`

6.6.1 Detailed Description

Definition at line 66 of file [Environment.hpp](#).

6.6.2 Member Typedef Documentation

6.6.2.1 `is_transparent`

```
using Environment::Enabler_Stringview_Hash::is_transparent = void
```

Sinaliza ao `unordered_map` que essa struct suporta tipos heterogêneos para pesquisa.

Definition at line 67 of file [Environment.hpp](#).

6.6.3 Member Function Documentation

6.6.3.1 `operator()()` [1/2]

```
::size_t Environment::Enabler_Stringview_Hash::operator() (
    const std::string & s ) const [inline]
```

Definition at line 69 of file [Environment.hpp](#).

6.6.3.2 `operator()()` [2/2]

```
::size_t Environment::Enabler_Stringview_Hash::operator() (
    std::string_view sv ) const [inline]
```

Definition at line 71 of file [Environment.hpp](#).

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

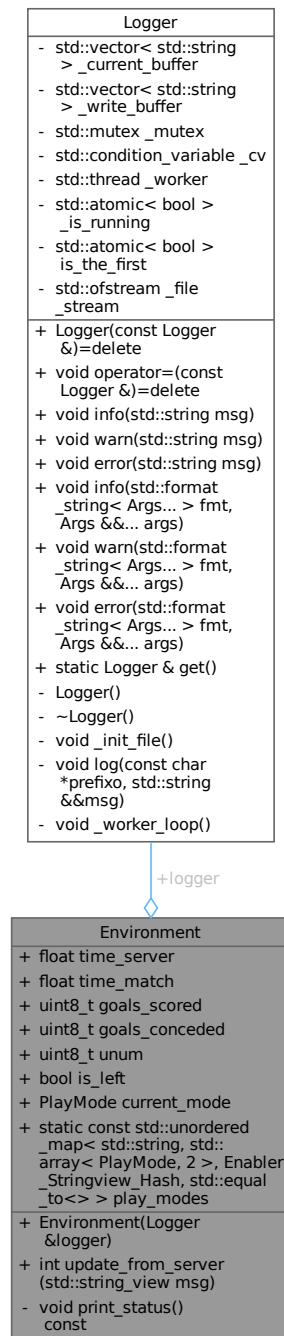
- `src/cpp/environment/Environment.hpp`

6.7 Environment Class Reference

Responsável por representar o ambiente externo ao robô

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

Collaboration diagram for Environment:



Classes

- struct [Enabler_Stringview_Hash](#)
- class [Parsing](#)

Responsável por prover ferramentas de auxílio de parsing.

Public Types

- enum class [PlayMode](#) : uint8_t {

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 }

< Tentaremos utilizar o mínimo possível de memória
- enum class [PlayModeGroup](#) : uint8_t {

OUR_KICK = 0 , THEIR_KICK = 1 , ACTIVE_BEAM = 2 , PASSIVE_BEAM = 3 ,

OTHER = 4 }

Public Member Functions

- [Environment \(Logger &logger\)](#)

Construtor da Classe.
- int [update_from_server](#) (std::string_view msg)

Interpretará as mensagens do servidor.

Public Attributes

- [Logger & logger](#)
- float [time_server](#)

Instante de Tempo do Servidor, útil apenas para sincronização entre agentes.
- float [time_match](#)

Instante de Tempo de Partida.
- uint8_t [goals_scored](#)

Nossos Gols, pode ser útil para mudarmos de tática conforme o jogo avança.
- uint8_t [goals_conceded](#)

Gols adversários, pode ser útil para mudarmos de tática conforme o jogo avança.
- uint8_t [unum](#)

Número do Jogador.
- bool [is_left](#)

De qual lado estamos.
- [PlayMode current_mode](#)

Static Public Attributes

- static const std::unordered_map< std::string, std::array< [PlayMode](#), 2 >, [Enabler_Stringview_Hash](#), std::equal_to<> > [play_modes](#)

Vamos precisar definir essa princesinha em outro lugar.

Private Member Functions

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

6.7.1 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](#).

6.7.2 Member Enumeration Documentation

6.7.2.1 PlayMode

```
enum class Environment::PlayMode : uint8_t [strong]
```

< Tentaremos utilizar o mínimo possível de memória

Enumerator

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

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

6.7.2.2 PlayModeGroup

```
enum class Environment::PlayModeGroup : uint8_t [strong]
```

Enumerator

OUR_KICK	
THEIR_KICK	
ACTIVE_BEAM	
PASSIVE_BEAM	
OTHER	

Definition at line 59 of file [Environment.hpp](#).

6.7.3 Constructor & Destructor Documentation

6.7.3.1 Environment()

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

Construtor da Classe.

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

6.7.4 Member Function Documentation

6.7.4.1 print_status()

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

Apresentará os dados lidos do servidor.

Definition at line 499 of file [Environment.hpp](#).

6.7.4.2 update_from_server()

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

Interpretará as mensagens do servidor.

Parameters

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

Returns

Atualização de todas as variáveis de ambiente.

< Vamos extrair uma tag

< Há apenas 'time'

< Pode ser 'GS' ou 'GYR'

< Tag Desconhecida

< Tag Superior Desconhecida

Definition at line 429 of file [Environment.hpp](#).

6.7.5 Member Data Documentation

6.7.5.1 `current_mode`

```
PlayMode Environment::current_mode
```

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

6.7.5.2 `goals_conceded`

```
uint8_t Environment::goals_conceded
```

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

Definition at line 85 of file [Environment.hpp](#).

6.7.5.3 `goals_scored`

```
uint8_t Environment::goals_scored
```

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

Definition at line 84 of file [Environment.hpp](#).

6.7.5.4 `is_left`

```
bool Environment::is_left
```

De qual lado estamos.

Definition at line 87 of file [Environment.hpp](#).

6.7.5.5 logger

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

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

6.7.5.6 play_modes

```
const std::unordered_map<std::string, std::array<PlayMode, 2>, Enabler_Stringview_Hash, std::equal_to<> > Environment::play_modes [static]
```

Vamos precisar definir essa princesinha em outro lugar.

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

6.7.5.7 time_match

```
float Environment::time_match
```

Instante de Tempo de Partida.

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

6.7.5.8 time_server

```
float Environment::time_server
```

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

Definition at line 82 of file [Environment.hpp](#).

6.7.5.9 unum

```
uint8_t Environment::unum
```

Número do Jogador.

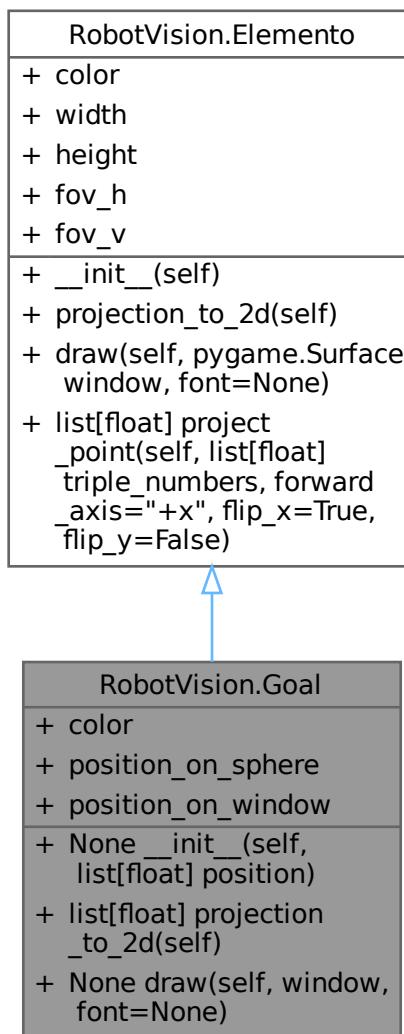
Definition at line 86 of file [Environment.hpp](#).

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

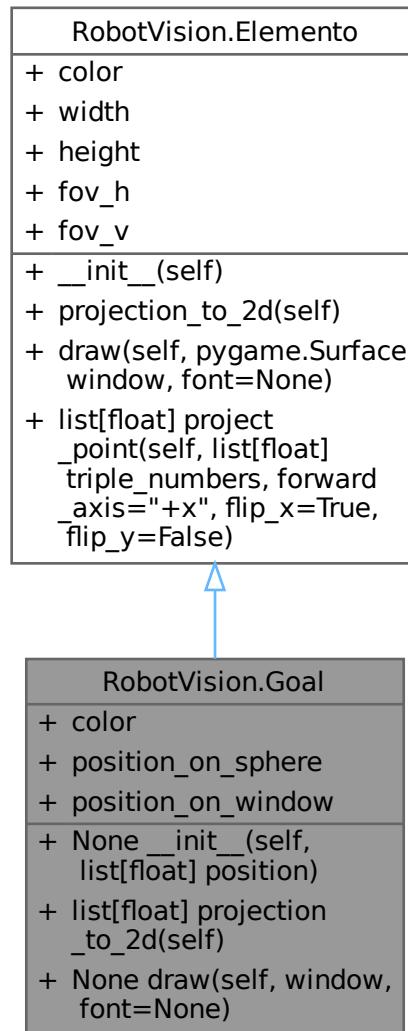
- src/cpp/environment/[Environment.hpp](#)

6.8 RobotVision.Goal Class Reference

Inheritance diagram for RobotVision.Goal:



Collaboration diagram for RobotVision.Goal:



Public Member Functions

- None [__init__](#) (self, list[float] position)
Construtor responsável por inicializar bola no e prover.
- list[float] [projection_to_2d](#) (self)
- None [draw](#) (self, window, font=None)

Public Member Functions inherited from [RobotVision.Elemento](#)

- list[float] [project_point](#) (self, list[float] triple_numbers, forward_axis="+x", flip_x=[True](#), flip_y=[False](#))

Public Attributes

- [color](#)
- [position_on_sphere](#)
- [position_on_window](#)

Public Attributes inherited from [RobotVision.Elemento](#)

- [color](#)
- [width](#)
- [height](#)
- [fov_h](#)
- [fov_v](#)

6.8.1 Detailed Description

Definition at line 120 of file [RobotVision.py](#).

6.8.2 Constructor & Destructor Documentation

6.8.2.1 `__init__()`

```
None RobotVision.Goal.__init__ (
    self,
    list[float] position )
```

Construtor responsável por inicializara bola no e prover.

Reimplemented from [RobotVision.Elemento](#).

Definition at line 121 of file [RobotVision.py](#).

6.8.3 Member Function Documentation

6.8.3.1 `draw()`

```
None RobotVision.Goal.draw (
    self,
    window,
    font = None )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 137 of file [RobotVision.py](#).

6.8.3.2 `projection_to_2d()`

```
list[float] RobotVision.Goal.projection_to_2d (
    self )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 132 of file [RobotVision.py](#).

6.8.4 Member Data Documentation

6.8.4.1 `color`

```
RobotVision.Goal.color
```

Definition at line 128 of file [RobotVision.py](#).

6.8.4.2 `position_on_sphere`

```
RobotVision.Goal.position_on_sphere
```

Definition at line 129 of file [RobotVision.py](#).

6.8.4.3 `position_on_window`

```
RobotVision.Goal.position_on_window
```

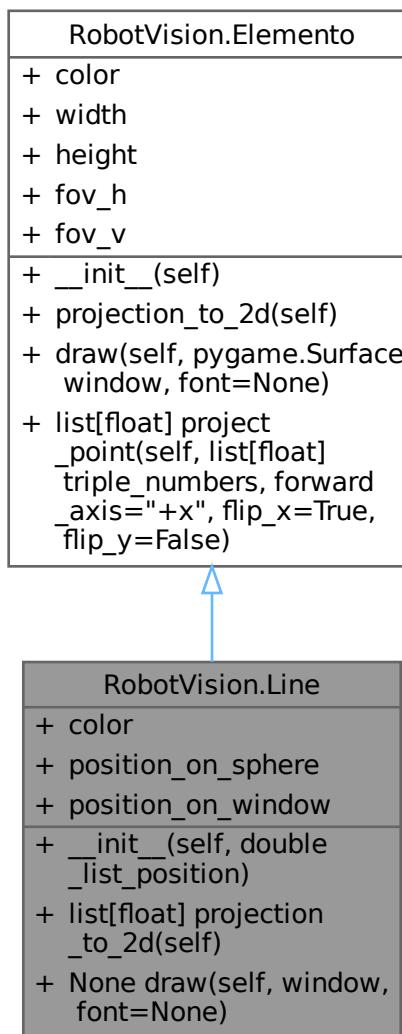
Definition at line 130 of file [RobotVision.py](#).

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

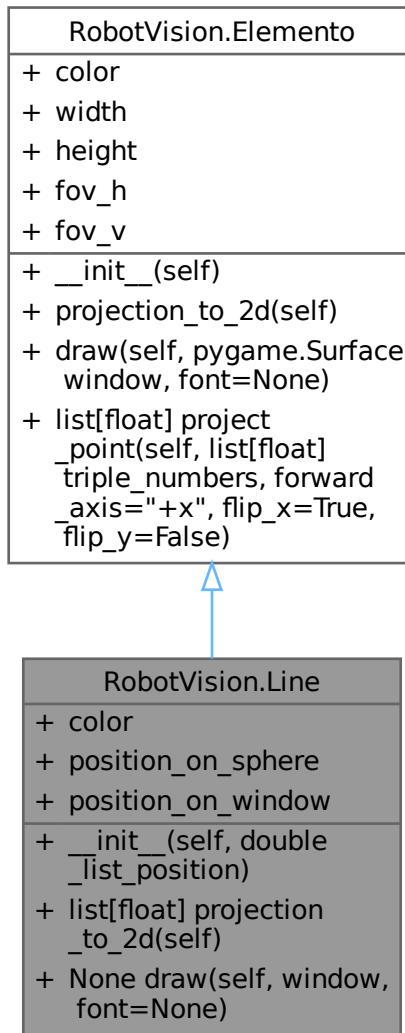
- src/utils/[RobotVision.py](#)

6.9 RobotVision.Line Class Reference

Inheritance diagram for RobotVision.Line:



Collaboration diagram for RobotVision.Line:



Public Member Functions

- `__init__` (self, double_list_position)
- list[float] `projection_to_2d` (self)
- None `draw` (self, window, font=None)

Public Member Functions inherited from `RobotVision.Elemento`

- list[float] `project_point` (self, list[float] triple_numbers, forward_axis="`+x`", flip_x=True, flip_y=False)

Public Attributes

- `color`
- `position_on_sphere`
- `position_on_window`

Public Attributes inherited from [RobotVision.Elemento](#)

- [color](#)
- [width](#)
- [height](#)
- [fov_h](#)
- [fov_v](#)

6.9.1 Detailed Description

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

6.9.2 Constructor & Destructor Documentation

6.9.2.1 `__init__()`

```
RobotVision.Line.__init__ (
    self,
    double_list_position )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 148 of file [RobotVision.py](#).

6.9.3 Member Function Documentation

6.9.3.1 `draw()`

```
None RobotVision.Line.draw (
    self,
    window,
    font = None )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 159 of file [RobotVision.py](#).

6.9.3.2 `projection_to_2d()`

```
list[float] RobotVision.Line.projection_to_2d (
    self )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 155 of file [RobotVision.py](#).

6.9.4 Member Data Documentation

6.9.4.1 color

RobotVision.Line.color

Definition at line 151 of file [RobotVision.py](#).

6.9.4.2 position_on_sphere

RobotVision.Line.position_on_sphere

Definition at line 152 of file [RobotVision.py](#).

6.9.4.3 position_on_window

RobotVision.Line.position_on_window

Definition at line 153 of file [RobotVision.py](#).

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

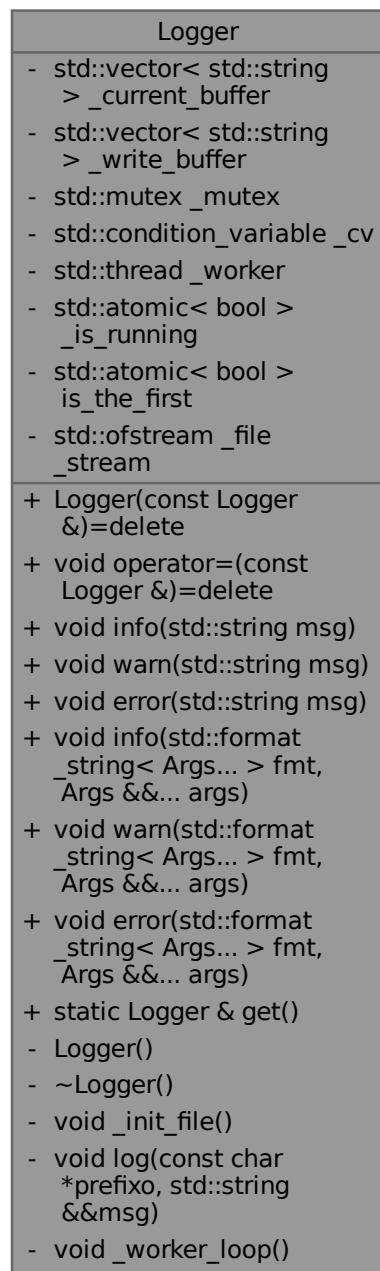
- [src/utils/RobotVision.py](#)

6.10 Logger Class Reference

Singleton para logging assíncrono.

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

Collaboration diagram for Logger:



Public Member Functions

- `Logger (const Logger &)=delete`
- `void operator= (const Logger &)=delete`
- `void info (std::string msg)`
Adiciona log nível INFO.
- `void warn (std::string msg)`

- `void error (std::string msg)`

Adiciona log nível WARN.
- `template<typename... Args> void info (std::format_string<Args... > fmt, Args &&... args)`

Log INFO usando C++20 std::format (Alta Performance).
- `template<typename... Args> void warn (std::format_string<Args... > fmt, Args &&... args)`

Log WARN usando C++20 std::format.
- `template<typename... Args> void error (std::format_string<Args... > fmt, Args &&... args)`

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

Static Public Member Functions

- `static Logger & get ()`

Acesso à instância única.

Private Member Functions

- `Logger ()`

Construtor privado: Inicializa arquivo e thread.
- `~Logger ()`

Destruitor: Sinaliza parada e espera thread terminar.
- `void _init_file ()`

Responsável por criar ambiente de .log.
- `void log (const char *prefixo, std::string &&msg)`

Responsável por providenciar genérica chamada de impressão em .log.
- `void _worker_loop ()`

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

Private Attributes

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

6.10.1 Detailed Description

Singleton para logging assíncrono.

Focada em performance utiliza uma lógica de fila de mensagens.

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

6.10.2 Constructor & Destructor Documentation

6.10.2.1 `Logger()` [1/2]

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

6.10.2.2 `Logger()` [2/2]

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

Construtor privado: Inicializa arquivo e thread.

Reservará 1000 slots para evitarmos realocações

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

6.10.2.3 `~Logger()`

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

Destrutor: Sinaliza parada e espera thread terminar.

< Informa a thread da condição de encerramento

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

6.10.3 Member Function Documentation

6.10.3.1 `_init_file()`

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

Responsável por criar ambiente de .log.

Possui uma lógica para garantir que logs sejam únicos.

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

6.10.3.2 `_worker_loop()`

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

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

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

< Agora escrevemos no disco SEM bloquear quem quer adicionar logs

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

6.10.3.3 error() [1/2]

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

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

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

6.10.3.4 error() [2/2]

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

Adiciona log nível ERROR.

Parameters

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

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

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

6.10.3.5 get()

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

Acesso à instância única.

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

6.10.3.6 info() [1/2]

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

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

Parameters

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

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

6.10.3.7 info() [2/2]

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

Adiciona log nível INFO.

Parameters

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

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

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

6.10.3.8 log()

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

Responsável por providenciar genérica chamada de impressão em .log.

Parameters

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

< Esse lock_guard trava enquanto estiver nesse escopo

Definition at line 149 of file [Logger.hpp](#).

6.10.3.9 operator=()

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

6.10.3.10 warn() [1/2]

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

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

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

6.10.3.11 warn() [2/2]

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

Adiciona log nível WARN.

Parameters

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

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

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

6.10.4 Member Data Documentation

6.10.4.1 _current_buffer

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

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

6.10.4.2 _cv

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

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

6.10.4.3 _file_stream

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

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

6.10.4.4 _is_running

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

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

6.10.4.5 _mutex

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

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

6.10.4.6 _worker

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

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

6.10.4.7 `_write_buffer`

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

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

6.10.4.8 `is_the_first`

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

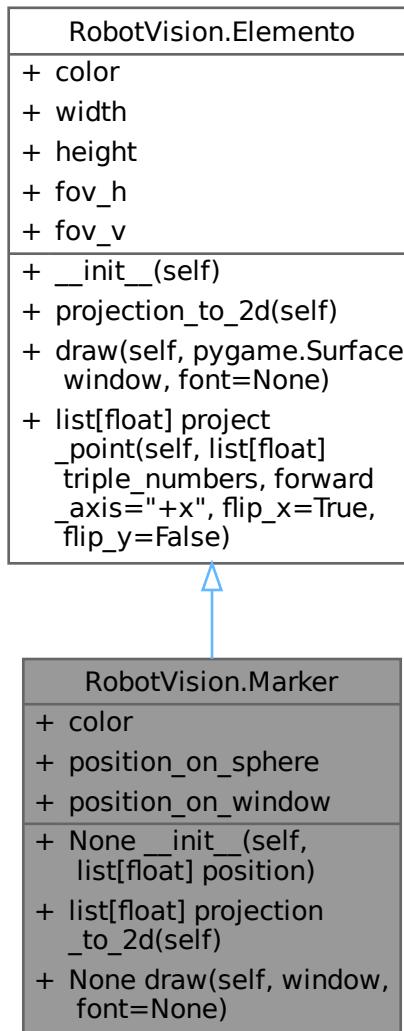
Definition at line 96 of file [Logger.hpp](#).

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

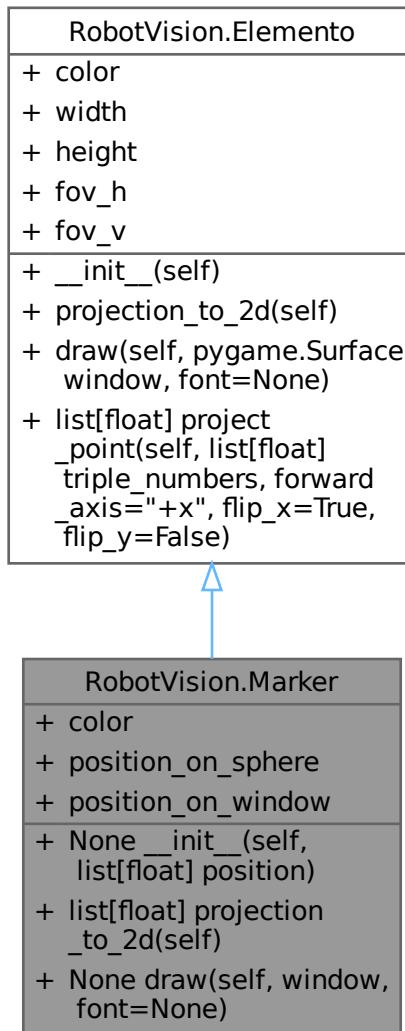
- src/cpp/logger/[Logger.hpp](#)

6.11 RobotVision.Marker Class Reference

Inheritance diagram for RobotVision.Marker:



Collaboration diagram for RobotVision.Marker:



Public Member Functions

- None [__init__](#) (self, list[float] position)
Construtor responsável por inicializar bola no e prover.
- list[float] [projection_to_2d](#) (self)
- None [draw](#) (self, window, font=None)

Public Member Functions inherited from [RobotVision.Elemento](#)

- list[float] [project_point](#) (self, list[float] triple_numbers, forward_axis="+x", flip_x=True, flip_y=False)

Public Attributes

- [color](#)
- [position_on_sphere](#)
- [position_on_window](#)

Public Attributes inherited from [RobotVision.Elemento](#)

- [color](#)
- [width](#)
- [height](#)
- [fov_h](#)
- [fov_v](#)

6.11.1 Detailed Description

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

6.11.2 Constructor & Destructor Documentation

6.11.2.1 `__init__()`

```
None RobotVision.Marker.__init__ (
    self,
    list[float] position )
```

Construtor responsável por inicializar bola no e prover.

Reimplemented from [RobotVision.Elemento](#).

Definition at line 95 of file [RobotVision.py](#).

6.11.3 Member Function Documentation

6.11.3.1 `draw()`

```
None RobotVision.Marker.draw (
    self,
    window,
    font = None )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 111 of file [RobotVision.py](#).

6.11.3.2 `projection_to_2d()`

```
list[float] RobotVision.Marker.projection_to_2d (
    self )
```

Reimplemented from [RobotVision.Elemento](#).

Definition at line 106 of file [RobotVision.py](#).

6.11.4 Member Data Documentation

6.11.4.1 `color`

```
RobotVision.Marker.color
```

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

6.11.4.2 `position_on_sphere`

```
RobotVision.Marker.position_on_sphere
```

Definition at line 103 of file [RobotVision.py](#).

6.11.4.3 `position_on_window`

```
RobotVision.Marker.position_on_window
```

Definition at line 104 of file [RobotVision.py](#).

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

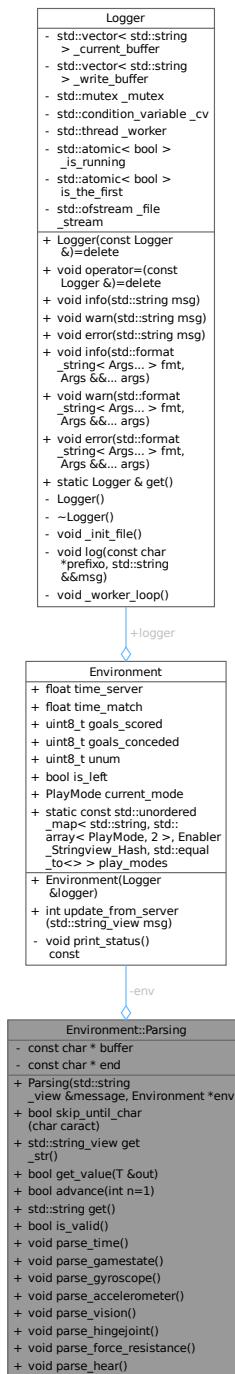
- [src/utils/RobotVision.py](#)

6.12 Environment::Parsing Class Reference

Responsável por prover ferramentas de auxílio de parsing.

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

Collaboration diagram for Environment::Parsing:



Public Member Functions

- **Parsing** (std::string_view &message, Environment *env)

Construtor do Parsing dedica à interpretação.
- **bool skip_until_char** (char caract)

Anançará até encontrar um determinado caractere de parada, pulando-o em seguida.
- **std::string_view get_str** ()

Ignorando eventuais ' ', '(' e ')', obterá a próxima string, encerrando apenas a encontrar ' '. Pulando este último caractere.
- template<typename T >
bool get_value (T &out)

Fará a conversão de caracteres em inteiro ou float, dependendo do tipo de referência dado.
- **bool advance** (int n=1)

Anançará o cursor uma determinada quantidade.
- **std::string get** ()
- **bool is_valid** ()
- **void parse_time** ()

Responsável pela interpretação da mensagem de 'time'.
- **void parse_gamestate** ()

Responsável pela interpretação da mensagem de 'GS'.
- **void parse_gyroscope** ()

Responsável pela interpretação da mensagem de 'GS'.
- **void parse_accelerometer** ()

Responsável pela interpretação da mensagem de 'ACC'.
- **void parse_vision** ()

Responsável pela interpretação da mensagem de 'See'.
- **void parse_hingejoint** ()

Responsável pela interpretação da mensagem de 'HJ'.
- **void parse_force_resistance** ()

Responsável pela interpretação da mensagem de 'FRP'.
- **void parse_hear** ()

Responsável pela interpretação da mensagem de 'hear'. Bem mais Complexo.

Private Attributes

- **const char * buffer = nullptr**

Permitirá-nos saber o ponto da mensagem que estamos.
- **const char * end = nullptr**

Permitirá-nos saber o ponto final.
- **Environment * env = nullptr**

Permitirá-nos modificar atributos.

6.12.1 Detailed Description

Responsável por prover ferramentas de auxílio de parsing.

Centralizará todas as funções inerentes ao parsing das mensagens.

Definition at line 99 of file [Environment.hpp](#).

6.12.2 Constructor & Destructor Documentation

6.12.2.1 Parsing()

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

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

Parameters

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

Returns

Atualização de todas as variáveis de ambiente.

Definition at line 113 of file [Environment.hpp](#).

6.12.3 Member Function Documentation

6.12.3.1 advance()

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

Avançará o cursor uma determinada quantidade.

Parameters

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

Returns

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

Definition at line 170 of file [Environment.hpp](#).

6.12.3.2 get()

```
std::string Environment::Parsing::get () [inline]
```

< Vamos pegar alguns endereços antes e alguns depois.

Definition at line 173 of file [Environment.hpp](#).

6.12.3.3 get_str()

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

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

Returns

`String_view` da string.

Definition at line 142 of file [Environment.hpp](#).

6.12.3.4 get_value()

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

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

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

Parameters

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

Returns

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

Definition at line 158 of file [Environment.hpp](#).

6.12.3.5 is_valid()

```
bool Environment::Parsing::is_valid ( ) [inline]
```

Definition at line 178 of file [Environment.hpp](#).

6.12.3.6 parse_accelerometer()

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

Responsável pela interpretação da mensagem de 'ACC'.

Recebe o vetor aceleração linear do centro do torso. Há toda uma lógica de sentido aqui, mas acredito que ainda não é importante.

Definition at line 273 of file [Environment.hpp](#).

6.12.3.7 parse_force_resistance()

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

Responsável pela interpretação da mensagem de 'FRP'.

Estes sensores estão embaixo de cada pé, este representado por lf ou rf. O primeiro vetor representa o ponto de contato do pé, medido em relação ao centro do mesmo. O segundo vetor representa a força(kg m/s²) total neste ponto.

Definition at line 399 of file [Environment.hpp](#).

6.12.3.8 parse_gamestate()

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

Responsável pela interpretação da mensagem de 'GS'.

Atualizará o instante de tempo da partida, o modo de jogo a cada ciclo e pontuações. Caso seja a primeira vez que receba, atualizará dados de número de uniforme, lado de campo. < Obteremos as subtags

< Poderá ser 'sl', 'sr'

< Há apenas 'pm'

< Há 'time' e 'team'

< Há apenas o 'u'

Definition at line 207 of file [Environment.hpp](#).

6.12.3.9 parse_gyroscope()

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

Responsável pela interpretação da mensagem de 'GS'.

Os números dados representam os incrementos e decrementos dos ângulos de rotação durante o ciclo, como uma espécie de velocidade. Em termos gerais, é o vetor velocidade angular no último ciclo em degraus por segundo do torso do robô.

Definition at line 257 of file [Environment.hpp](#).

6.12.3.10 parse_hear()

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

Responsável pela interpretação da mensagem de 'hear'. Bem mais Complexo.

Definition at line 418 of file [Environment.hpp](#).

6.12.3.11 parse_hingejoint()

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

Responsável pela interpretação da mensagem de 'HJ'.

Recebemos o nome abreviado da junta e o ângulo instantâneo do eixo em degraus.

Definition at line 381 of file [Environment.hpp](#).

6.12.3.12 parse_time()

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

Responsável pela interpretação da mensagem de 'time'.

Informará o instante de tempo do servidor. < Vamos ter fé que nunca será diferente.

< Sairemos da tag 'time'

Definition at line 188 of file [Environment.hpp](#).

6.12.3.13 parse_vision()

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

Responsável pela interpretação da mensagem de 'See'.

Recebe diversas(MUITAS) informações a partir de pontos em coordenadas esféricas. < Estamos vendo um jogador. Há outras lowers tags a serem verificadas.

< Informação de 'team' do jogador visto

< Saberemos o unum do jogador visto

< Obviamente, a bola.

< Linhas Vistas

Definition at line 287 of file [Environment.hpp](#).

6.12.3.14 skip_until_char()

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

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

Parameters

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

Returns

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

Definition at line 128 of file [Environment.hpp](#).

6.12.4 Member Data Documentation

6.12.4.1 buffer

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

Permitirá-nos saber o ponto da mensagem que estamos.

Definition at line 101 of file [Environment.hpp](#).

6.12.4.2 end

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

Permitirá-nos saber o ponto final.

Definition at line 102 of file [Environment.hpp](#).

6.12.4.3 env

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

Permitirá-nos modificar atributos.

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

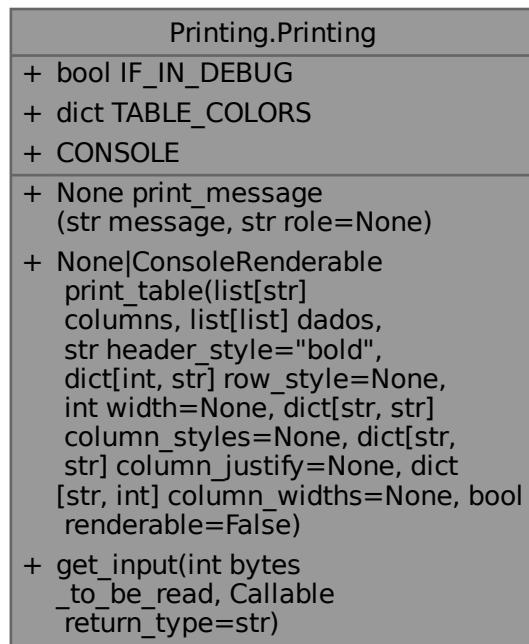
The documentation for this class was generated from the following file:

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

6.13 Printing.Printing Class Reference

Responsável pela comunicação usuário - terminal.

Collaboration diagram for Printing.Printing:



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()

6.13.1 Detailed Description

Responsável pela comunicação usuário - terminal.

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

6.13.2 Member Function Documentation

6.13.2.1 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](#).

6.13.2.2 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](#).

6.13.2.3 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>dados</i>	Lista de listas com os valores de linhas

Assume os seguintes parâmetros de personalização: *columns*: Lista de nomes das colunas *dados*: 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](#).

6.13.3 Member Data Documentation

6.13.3.1 CONSOLE

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

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

6.13.3.2 IF_IN_DEBUG

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

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

6.13.3.3 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](#).

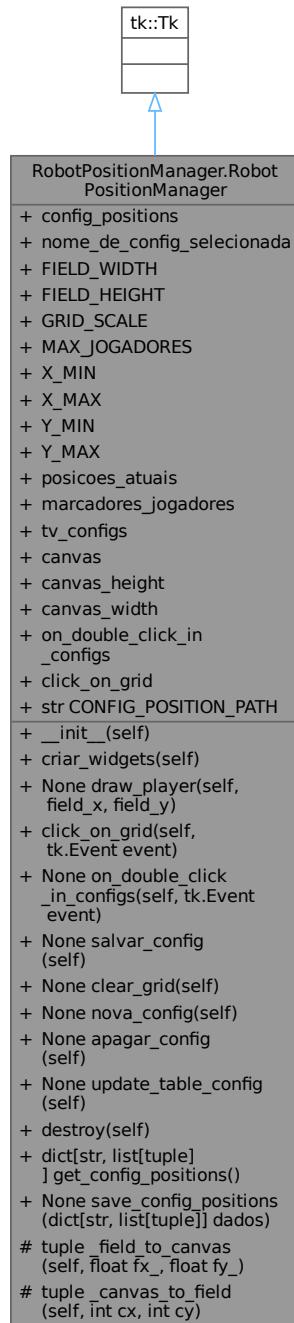
The documentation for this class was generated from the following file:

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

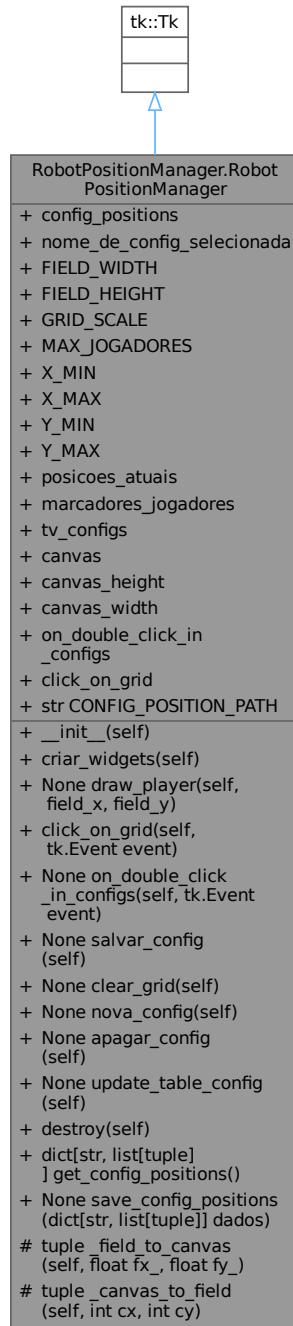
6.14 RobotPositionManager.RobotPositionManager Class Reference

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.

6.14.1 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](#).

6.14.2 Constructor & Destructor Documentation

6.14.2.1 [__init__\(\)](#)

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

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

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

6.14.3 Member Function Documentation

6.14.3.1 [_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](#).

6.14.3.2 `_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

<code>fx</code>	Coordenada real em x
<codefy< code=""></codefy<>	Coordenada real em y

Returns

Coordenadas corrigidas para o grid

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

6.14.3.3 `apagar_config()`

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

Apagará uma configuração selecionada.

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

6.14.3.4 `clear_grid()`

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

Responsável por limpar as posições e a grade.

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

6.14.3.5 `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](#).

6.14.3.6 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](#).

6.14.3.7 destroy()

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

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

6.14.3.8 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](#).

6.14.3.9 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](#).

6.14.3.10 nova_config()

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

Prepará uma nova configuração para ser criada.

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

6.14.3.11 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](#).

6.14.3.12 salvar_config()

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

Salvará uma configuração selecionada.

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

6.14.3.13 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](#).

6.14.3.14 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](#).

6.14.4 Member Data Documentation

6.14.4.1 canvas

```
RobotPositionManager.RobotPositionManager.canvas
```

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

6.14.4.2 canvas_height

```
RobotPositionManager.RobotPositionManager.canvas_height
```

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

6.14.4.3 canvas_width

```
RobotPositionManager.RobotPositionManager.canvas_width
```

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

6.14.4.4 click_on_grid

```
RobotPositionManager.RobotPositionManager.click_on_grid
```

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

6.14.4.5 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](#).

6.14.4.6 config_positions

RobotPositionManager.RobotPositionManager.config_positions

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

6.14.4.7 FIELD_HEIGHT

RobotPositionManager.RobotPositionManager.FIELD_HEIGHT

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

6.14.4.8 FIELD_WIDTH

RobotPositionManager.RobotPositionManager.FIELD_WIDTH

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

6.14.4.9 GRID_SCALE

RobotPositionManager.RobotPositionManager.GRID_SCALE

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

6.14.4.10 marcadores_jogadores

RobotPositionManager.RobotPositionManager.marcadores_jogadores

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

6.14.4.11 MAX_JOGADORES

RobotPositionManager.RobotPositionManager.MAX_JOGADORES

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

6.14.4.12 nome_de_config_selecionada

RobotPositionManager.RobotPositionManager.nome_de_config_selecionada

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

6.14.4.13 on_double_click_in_configs

RobotPositionManager.RobotPositionManager.on_double_click_in_configs

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

6.14.4.14 `posicoes_atuais`

`RobotPositionManager.RobotPositionManager.posicoes_atuais`

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

6.14.4.15 `tv_configs`

`RobotPositionManager.RobotPositionManager.tv_configs`

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

6.14.4.16 `X_MAX`

`RobotPositionManager.RobotPositionManager.X_MAX`

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

6.14.4.17 `X_MIN`

`RobotPositionManager.RobotPositionManager.X_MIN`

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

6.14.4.18 `Y_MAX`

`RobotPositionManager.RobotPositionManager.Y_MAX`

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

6.14.4.19 `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/utils/RobotPositionManager.py](#)

6.15 RobotVision.RobotVision Class Reference

Classe responsável por gerir a aplicação principal.

Collaboration diagram for RobotVision.RobotVision:

RobotVision.RobotVision
+ frames
+ current_index
+ need_to_update
+ objects
+ str FRAMES_VISION_PATH
+ __init__(self)
+ None load_frames_from_file(self)
+ None parse_frame(self)
+ None mainloop(self)
+ draw_legend(screen, items, font, padding =10, line_height=20)
str None _get_only_tag_See(self)

Public Member Functions

- [__init__](#) (self)
Abrirá um arquivo que conterá os frames recebidos pelo agente.
- None [load_frames_from_file](#) (self)
Interpreta uma mensagem 'See' do Simspark/Rcssserver3d.
- None [parse_frame](#) (self)
Desenhará a legenda das cores.
- None [mainloop](#) (self)

Static Public Member Functions

- [draw_legend](#) (screen, items, font, padding=10, line_height=20)
Desenhará a legenda das cores.

Public Attributes

- [frames](#)
- [current_index](#)
- [need_to_update](#)
- [objects](#)

Static Public Attributes

- str `FRAMES_VISION_PATH` = "frames_vision.txt"

Protected Member Functions

- str|None `_get_only_tag_See` (self)
Buscará no frame principal o bloco referente ao conjunto See.

6.15.1 Detailed Description

Classe responsável por gerir a aplicação principal.

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

6.15.2 Constructor & Destructor Documentation

6.15.2.1 `__init__()`

```
RobotVision.RobotVision.__init__ (
    self )
```

Definition at line 177 of file [RobotVision.py](#).

6.15.3 Member Function Documentation

6.15.3.1 `_get_only_tag_See()`

```
str | None RobotVision.RobotVision._get_only_tag_See (
    self ) [protected]
```

Buscará no frame principal o bloco referente ao conjunto See.

Returns

String referente ao bloco See. None caso não exista.

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

6.15.3.2 `draw_legend()`

```
RobotVision.RobotVision.draw_legend (
    screen,
    items,
    font,
    padding = 10,
    line_height = 20 ) [static]
```

Desenhará a legenda das cores.

Definition at line 341 of file [RobotVision.py](#).

6.15.3.3 load_frames_from_file()

```
None RobotVision.RobotVision.load_frames_from_file (
    self )
```

Abrirá um arquivo que conterá os frames recebidos pelo agente.

Pode ser aprimorada para permitir observação em tempo real.

Definition at line 183 of file [RobotVision.py](#).

6.15.3.4 mainloop()

```
None RobotVision.RobotVision.mainloop (
    self )
```

Definition at line 358 of file [RobotVision.py](#).

6.15.3.5 parse_frame()

```
None RobotVision.RobotVision.parse_frame (
    self )
```

Interpreta uma mensagem 'See' do Simspark/Rcssserver3d.

Divide a responsabilidade com subfunções.

Definition at line 232 of file [RobotVision.py](#).

6.15.4 Member Data Documentation

6.15.4.1 current_index

```
RobotVision.RobotVision.current_index
```

Definition at line 179 of file [RobotVision.py](#).

6.15.4.2 frames

```
RobotVision.RobotVision.frames
```

Definition at line 178 of file [RobotVision.py](#).

6.15.4.3 FRAMES_VISION_PATH

```
str RobotVision.RobotVision.FRAMES_VISION_PATH = "frames_vision.txt" [static]
```

Definition at line 175 of file [RobotVision.py](#).

6.15.4.4 need_to_update

RobotVision.RobotVision.need_to_update

Definition at line 180 of file [RobotVision.py](#).

6.15.4.5 objects

RobotVision.RobotVision.objects

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

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

- [src/utils/RobotVision.py](#)

6.16 ServerComm.ServerComm Class Reference

Responsável pela comunicação com servidor.

Collaboration diagram for ServerComm.ServerComm:

ServerComm.ServerComm	
+ buffer_size	
+ buffer	
+ socket	
+ message_queue	
+ unum	
+ env	
+ __init__(self, list [list[str]] creation options, 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 __receive_async (self, list other_players)	

Public Member Functions

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

Private Member Functions

- None `__receive_async` (self, list other_players)
Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de execução.

6.16.1 Detailed Description

Responsável pela comunicação com servidor.

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

6.16.2 Constructor & Destructor Documentation

6.16.2.1 `__init__()`

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

6.16.3 Member Function Documentation

6.16.3.1 `__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 153 of file [ServerComm.py](#).

6.16.3.2 `clear_queue()`

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

Limpará a fila de commits.

Definition at line 225 of file [ServerComm.py](#).

6.16.3.3 `close()`

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

Responsável por fazer o encerramento dos canais de comunicação.

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

6.16.3.4 `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 190 of file [ServerComm.py](#).

6.16.3.5 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 232 of file [ServerComm.py](#).

6.16.3.6 receive()

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

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

Definition at line 95 of file [ServerComm.py](#).

6.16.3.7 send()

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

Enviará ao servidor todas as mensagens commitadas.

Definition at line 205 of file [ServerComm.py](#).

6.16.3.8 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 80 of file [ServerComm.py](#).

6.16.4 Member Data Documentation

6.16.4.1 buffer

`ServerComm.ServerComm.buffer`

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

6.16.4.2 buffer_size

`ServerComm.ServerComm.buffer_size`

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

6.16.4.3 env

`ServerComm.ServerComm.env`

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

6.16.4.4 message_queue

`ServerComm.ServerComm.message_queue`

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

6.16.4.5 socket

`ServerComm.ServerComm.socket`

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

6.16.4.6 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 Penálti
00025         - Proxy
00026         - Modo de Debug
00027     """
00028
00029     self.unum = creation_options[4][1]
00030     creation_options[5][1] = (0,1,1,1,2,3,3,3,4,4,4)[self.unum - 1]
00031
00032     super().__init__(creation_options)
00033

```

7.3 src/agent/AgentPenalty.py File Reference

Implementação de Lógica de Goleiro.

Namespaces

- namespace [AgentPenalty](#)

7.3.1 Detailed Description

Implementação de Lógica de Goleiro.

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

7.4 AgentPenalty.py

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

```

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

```

7.5 src/agent/BaseAgent.py File Reference

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

Classes

- class [BaseAgent](#).[BaseAgent](#)

Classe que agrupará todas as funcionalidades comuns a qualquer agente.

Namespaces

- namespace [BaseAgent](#)

7.5.1 Detailed Description

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

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

7.6 BaseAgent.py

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

```

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

```

7.7 src/communication/ServerComm.py File Reference

Implementação da Comunicação com Servidor.

Classes

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

Namespaces

- namespace [ServerComm](#)

7.7.1 Detailed Description

Implementação da Comunicação com Servidor.

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

7.8 ServerComm.py

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

```

00001 """
00002 @file ServerComm.py
00003 @brief Implementação da Comunicação com Servidor
00004 """
00005 import socket
00006 from time import sleep
00007 from term.Printing import Printing
00008 from select import select
00009 from pathlib import Path
00010
00011 class ServerComm:
00012     """
00013     @brief Responsável pela comunicação com servidor.
00014     """
00015
00016     def __init__(self, creation_options: list[list[str]], environment, other_players: list):
00017         """
00018         @brief Construtor da classe, inicializando buffers e a conexão de cada agente com servidor.
00019         @param creation_options Lista de parâmetros de criação, self ainda não foi incluído na lista.
00020         @param environment
00021         @param other_players
00022         """
00023
00024         # Características da comunicação
00025         self.buffer_size = 4096 # Posteriormente, devemos analisar se realmente vale a pena ter um
00026         buffer com este comprimento
00027         self.buffer = bytearray(self.buffer_size)
00028         self.socket = socket.socket(
00029             socket.AF_INET,
00030             socket.SOCK_STREAM # TCP
00031         )
00032         self.socket.settimeout(2)
00033
00034         # Características alheias
00035         self.message_queue = []
00036         self.unum = creation_options[4][1]
00037         self.env = environment
00038
00039         # Fazemos a conexão com servidor
00040         Printing.print_message(f"Tentando conexão do jogador {self.unum}", "info")
00041         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
00066 # Aqui podem ser realizados testes de execução de quaisquer funções do ServerComm
00067
00068 for _ in range(3):
00069     self.send_immediate(b'(syn)')
00070     for p in other_players:
00071         p.scom.send_immediate(b'(syn)')
00072     for p in other_players:
00073         p.scom.receive()
00074     self.receive()
00075
00076
00077 # self.close()
00078
00079 # Métodos Mínimos da Classe de Comunicação com servidor
00080 def send_immediate(self, message: bytes) -> None:
00081     """
00082     @brief Envia uma mensagem instantânea ao servidor, verificando se a conexão continua ativa
00083     @param message String em forma de bytes para ser transmitida
00084     @details
00085     Coloca-se na frente uma informação de tamanho da mensagem dentro de 4 bytes.
00086     """
00087
00088     try:
00089         self.socket.send(
00090             len(message).to_bytes(4, byteorder="big") + message
00091         )
00092     except BrokenPipeError:
00093         Printing.print_message("Error: socket foi fechado por rcssserver3d", "error")
00094
00095 def receive(self) -> None:
00096     """
00097     @brief Receberá informações diretamente do servidor, fazendo todas as verificações
00098     necessárias.
00099     """
00100     msg_size = None
00101     while True:
00102         try:
00103             # Verificamos se há 4 bytes no cabeçalho e nos preparamos para ler.
00104             if self.socket.recv_into(
00105                 self.buffer, nbytes=4
00106             ) != 4:
00107                 raise ConnectionResetError
00108
00109             # Lemos o comprimento total da mensagem
00110             msg_size = int.from_bytes(
00111                 self.buffer[:4], # Garantimos leitura de apenas 4 bytes
00112                 byteorder="big", # ordem de significativo
00113                 signed=False # se tem sinal
00114             )
00115
00116             # Lemos o restante da mensagem
00117             if(
00118                 self.socket.recv_into(
00119                     self.buffer,
00120                     nbytes=msg_size
00121                 )
00122             ) != msg_size:
00123                 raise ConnectionResetError
00124
00125     except ConnectionResetError:
00126         Printing.print_message("\nError: socket foi fechado pelo rcssserver3d.", "error")
00127         exit()
00128

```

```

00129     except TimeoutError:
00130         pass
00131
00132     if len(
00133         select( # Monitora sockets/arquivos para I/O
00134             [self.socket], # Lista de sockets/arquivos para verificar leitura
00135             [], # Lista vazia para escrita
00136             [], # Lista vazia para exceções
00137             0.0 # timeout zero (não bloqueante)
00138         )[0] # Pegamos o primeiro socket para leitura
00139     ) == 0: # Logo, não há dados disponíveis para leitura
00140         break
00141
00142     # Como há algo para ser lido, devemos aplicar o parser
00143     self.env.update_from_server(self.buffer[:msg_size])
00144
00145     # if self.buffer[:msg_size].find(b' (See') != -1:
00146     #     with open(
00147     #         Path(__file__).resolve().parents[1] / "utils" / "frames_vision.txt",
00148     #         "a"
00149     #     ) as f:
00150     #         f.write(self.buffer[:msg_size].decode())
00151     #         f.write("\n")
00152
00153 def __receive_async(self, other_players: list) -> None:
00154 """
00155     @brief Responsável por esperar resposta do servidor de forma assíncrona, sem impedir fluxo de
00156     execução
00157     @details
00158     Essa função foi criada com o único propósito de impedir que a espera por resposta
00159     do servidor interrompa o fluxo de execução. Não deve ser executada posteriormente.
00160     @param other_players Lista de jogadores de mesmo time presentes na partida
00161
00162     # Caso não haja ninguém além dele
00163     if not other_players:
00164         # Sem isso, um loop infinito existiria
00165         return self.receive()
00166
00167     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00168     self.socket.setblocking(False)
00169
00170     while True:
00171         try:
00172             Printing.print_message(".")
00173             self.receive()
00174             break
00175         except BlockingIOError:
00176             pass
00177
00178         # Força que todos estejam em condições
00179         for p in other_players:
00180             p.scom.send_immediate(b"(syn)")
00181
00182         for p in other_players:
00183             p.scom.receive()
00184
00185         # Voltamos ao padrão
00186         self.socket.setblocking(True)
00187
00188     return None
00189
00190 def commit(self, message: bytes) -> None:
00191 """
00192     @brief Responsável por adicionar uma nova mensagem à fila de mensagens
00193     @param message String em bytes a ser adicionada à fila
00194 """
00195     assert isinstance(message, bytes), "Mensagem deve estar em bytes"
00196     self.message_queue.append(message)
00197
00198 def close(self) -> None:
00199 """
00200     @brief Responsável por fazer o encerramento dos canais de comunicação
00201 """
00202
00203     self.socket.close()
00204
00205 def send(self) -> None:
00206 """
00207     @brief Enviará ao servidor todas as mensagens commitadas.
00208 """
00209
00210     if len(
00211         select(
00212             [self.socket],
00213             [],
00214             [],
00215             0.0
00216         )[0]
00217     ) == 0:
00218         self.socket.sendall(self.message_queue.get())
00219
00220     self.message_queue.task_done()
00221
00222     if self.message_queue.empty():
00223         self.close()
00224
00225     return None
00226
00227     # Caso não haja ninguém além dele
00228     if not other_players:
00229         # Sem isso, um loop infinito existiria
00230         return self.receive()
00231
00232     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00233     self.socket.setblocking(False)
00234
00235     while True:
00236         try:
00237             Printing.print_message(".")
00238             self.receive()
00239             break
00240         except BlockingIOError:
00241             pass
00242
00243         # Força que todos estejam em condições
00244         for p in other_players:
00245             p.scom.send_immediate(b"(syn)")
00246
00247         for p in other_players:
00248             p.scom.receive()
00249
00250         # Voltamos ao padrão
00251         self.socket.setblocking(True)
00252
00253     return None
00254
00255 def __receive(self) -> bytes:
00256 """
00257     @brief Responsável por receber mensagens do servidor
00258     @return bytes
00259 """
00260
00261     if self.message_queue.empty():
00262         self.socket.setblocking(True)
00263
00264         while True:
00265             try:
00266                 self.receive()
00267             except BlockingIOError:
00268                 break
00269
00270         self.socket.setblocking(False)
00271
00272     return self.message_queue.get()
00273
00274     # Caso não haja ninguém além dele
00275     if not other_players:
00276         # Sem isso, um loop infinito existiria
00277         return self.receive()
00278
00279     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00280     self.socket.setblocking(False)
00281
00282     while True:
00283         try:
00284             Printing.print_message(".")
00285             self.receive()
00286             break
00287         except BlockingIOError:
00288             pass
00289
00290         # Força que todos estejam em condições
00291         for p in other_players:
00292             p.scom.receive()
00293
00294         for p in other_players:
00295             p.scom.send_immediate(b"(syn)")
00296
00297         # Voltamos ao padrão
00298         self.socket.setblocking(True)
00299
00300     return self.message_queue.get()
00301
00302     # Caso não haja ninguém além dele
00303     if not other_players:
00304         # Sem isso, um loop infinito existiria
00305         return self.receive()
00306
00307     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00308     self.socket.setblocking(False)
00309
00310     while True:
00311         try:
00312             Printing.print_message(".")
00313             self.receive()
00314             break
00315         except BlockingIOError:
00316             pass
00317
00318         # Força que todos estejam em condições
00319         for p in other_players:
00320             p.scom.receive()
00321
00322         for p in other_players:
00323             p.scom.send_immediate(b"(syn)")
00324
00325         # Voltamos ao padrão
00326         self.socket.setblocking(True)
00327
00328     return self.message_queue.get()
00329
00330     # Caso não haja ninguém além dele
00331     if not other_players:
00332         # Sem isso, um loop infinito existiria
00333         return self.receive()
00334
00335     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00336     self.socket.setblocking(False)
00337
00338     while True:
00339         try:
00340             Printing.print_message(".")
00341             self.receive()
00342             break
00343         except BlockingIOError:
00344             pass
00345
00346         # Força que todos estejam em condições
00347         for p in other_players:
00348             p.scom.receive()
00349
00350         for p in other_players:
00351             p.scom.send_immediate(b"(syn)")
00352
00353         # Voltamos ao padrão
00354         self.socket.setblocking(True)
00355
00356     return self.message_queue.get()
00357
00358     # Caso não haja ninguém além dele
00359     if not other_players:
00360         # Sem isso, um loop infinito existiria
00361         return self.receive()
00362
00363     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00364     self.socket.setblocking(False)
00365
00366     while True:
00367         try:
00368             Printing.print_message(".")
00369             self.receive()
00370             break
00371         except BlockingIOError:
00372             pass
00373
00374         # Força que todos estejam em condições
00375         for p in other_players:
00376             p.scom.receive()
00377
00378         for p in other_players:
00379             p.scom.send_immediate(b"(syn)")
00380
00381         # Voltamos ao padrão
00382         self.socket.setblocking(True)
00383
00384     return self.message_queue.get()
00385
00386     # Caso não haja ninguém além dele
00387     if not other_players:
00388         # Sem isso, um loop infinito existiria
00389         return self.receive()
00390
00391     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00392     self.socket.setblocking(False)
00393
00394     while True:
00395         try:
00396             Printing.print_message(".")
00397             self.receive()
00398             break
00399         except BlockingIOError:
00400             pass
00401
00402         # Força que todos estejam em condições
00403         for p in other_players:
00404             p.scom.receive()
00405
00406         for p in other_players:
00407             p.scom.send_immediate(b"(syn)")
00408
00409         # Voltamos ao padrão
00410         self.socket.setblocking(True)
00411
00412     return self.message_queue.get()
00413
00414     # Caso não haja ninguém além dele
00415     if not other_players:
00416         # Sem isso, um loop infinito existiria
00417         return self.receive()
00418
00419     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00420     self.socket.setblocking(False)
00421
00422     while True:
00423         try:
00424             Printing.print_message(".")
00425             self.receive()
00426             break
00427         except BlockingIOError:
00428             pass
00429
00430         # Força que todos estejam em condições
00431         for p in other_players:
00432             p.scom.receive()
00433
00434         for p in other_players:
00435             p.scom.send_immediate(b"(syn)")
00436
00437         # Voltamos ao padrão
00438         self.socket.setblocking(True)
00439
00440     return self.message_queue.get()
00441
00442     # Caso não haja ninguém além dele
00443     if not other_players:
00444         # Sem isso, um loop infinito existiria
00445         return self.receive()
00446
00447     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00448     self.socket.setblocking(False)
00449
00450     while True:
00451         try:
00452             Printing.print_message(".")
00453             self.receive()
00454             break
00455         except BlockingIOError:
00456             pass
00457
00458         # Força que todos estejam em condições
00459         for p in other_players:
00460             p.scom.receive()
00461
00462         for p in other_players:
00463             p.scom.send_immediate(b"(syn)")
00464
00465         # Voltamos ao padrão
00466         self.socket.setblocking(True)
00467
00468     return self.message_queue.get()
00469
00470     # Caso não haja ninguém além dele
00471     if not other_players:
00472         # Sem isso, um loop infinito existiria
00473         return self.receive()
00474
00475     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00476     self.socket.setblocking(False)
00477
00478     while True:
00479         try:
00480             Printing.print_message(".")
00481             self.receive()
00482             break
00483         except BlockingIOError:
00484             pass
00485
00486         # Força que todos estejam em condições
00487         for p in other_players:
00488             p.scom.receive()
00489
00490         for p in other_players:
00491             p.scom.send_immediate(b"(syn)")
00492
00493         # Voltamos ao padrão
00494         self.socket.setblocking(True)
00495
00496     return self.message_queue.get()
00497
00498     # Caso não haja ninguém além dele
00499     if not other_players:
00500         # Sem isso, um loop infinito existiria
00501         return self.receive()
00502
00503     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00504     self.socket.setblocking(False)
00505
00506     while True:
00507         try:
00508             Printing.print_message(".")
00509             self.receive()
00510             break
00511         except BlockingIOError:
00512             pass
00513
00514         # Força que todos estejam em condições
00515         for p in other_players:
00516             p.scom.receive()
00517
00518         for p in other_players:
00519             p.scom.send_immediate(b"(syn)")
00520
00521         # Voltamos ao padrão
00522         self.socket.setblocking(True)
00523
00524     return self.message_queue.get()
00525
00526     # Caso não haja ninguém além dele
00527     if not other_players:
00528         # Sem isso, um loop infinito existiria
00529         return self.receive()
00530
00531     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00532     self.socket.setblocking(False)
00533
00534     while True:
00535         try:
00536             Printing.print_message(".")
00537             self.receive()
00538             break
00539         except BlockingIOError:
00540             pass
00541
00542         # Força que todos estejam em condições
00543         for p in other_players:
00544             p.scom.receive()
00545
00546         for p in other_players:
00547             p.scom.send_immediate(b"(syn)")
00548
00549         # Voltamos ao padrão
00550         self.socket.setblocking(True)
00551
00552     return self.message_queue.get()
00553
00554     # Caso não haja ninguém além dele
00555     if not other_players:
00556         # Sem isso, um loop infinito existiria
00557         return self.receive()
00558
00559     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00560     self.socket.setblocking(False)
00561
00562     while True:
00563         try:
00564             Printing.print_message(".")
00565             self.receive()
00566             break
00567         except BlockingIOError:
00568             pass
00569
00570         # Força que todos estejam em condições
00571         for p in other_players:
00572             p.scom.receive()
00573
00574         for p in other_players:
00575             p.scom.send_immediate(b"(syn)")
00576
00577         # Voltamos ao padrão
00578         self.socket.setblocking(True)
00579
00580     return self.message_queue.get()
00581
00582     # Caso não haja ninguém além dele
00583     if not other_players:
00584         # Sem isso, um loop infinito existiria
00585         return self.receive()
00586
00587     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00588     self.socket.setblocking(False)
00589
00590     while True:
00591         try:
00592             Printing.print_message(".")
00593             self.receive()
00594             break
00595         except BlockingIOError:
00596             pass
00597
00598         # Força que todos estejam em condições
00599         for p in other_players:
00600             p.scom.receive()
00601
00602         for p in other_players:
00603             p.scom.send_immediate(b"(syn)")
00604
00605         # Voltamos ao padrão
00606         self.socket.setblocking(True)
00607
00608     return self.message_queue.get()
00609
00610     # Caso não haja ninguém além dele
00611     if not other_players:
00612         # Sem isso, um loop infinito existiria
00613         return self.receive()
00614
00615     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00616     self.socket.setblocking(False)
00617
00618     while True:
00619         try:
00620             Printing.print_message(".")
00621             self.receive()
00622             break
00623         except BlockingIOError:
00624             pass
00625
00626         # Força que todos estejam em condições
00627         for p in other_players:
00628             p.scom.receive()
00629
00630         for p in other_players:
00631             p.scom.send_immediate(b"(syn)")
00632
00633         # Voltamos ao padrão
00634         self.socket.setblocking(True)
00635
00636     return self.message_queue.get()
00637
00638     # Caso não haja ninguém além dele
00639     if not other_players:
00640         # Sem isso, um loop infinito existiria
00641         return self.receive()
00642
00643     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00644     self.socket.setblocking(False)
00645
00646     while True:
00647         try:
00648             Printing.print_message(".")
00649             self.receive()
00650             break
00651         except BlockingIOError:
00652             pass
00653
00654         # Força que todos estejam em condições
00655         for p in other_players:
00656             p.scom.receive()
00657
00658         for p in other_players:
00659             p.scom.send_immediate(b"(syn)")
00660
00661         # Voltamos ao padrão
00662         self.socket.setblocking(True)
00663
00664     return self.message_queue.get()
00665
00666     # Caso não haja ninguém além dele
00667     if not other_players:
00668         # Sem isso, um loop infinito existiria
00669         return self.receive()
00670
00671     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00672     self.socket.setblocking(False)
00673
00674     while True:
00675         try:
00676             Printing.print_message(".")
00677             self.receive()
00678             break
00679         except BlockingIOError:
00680             pass
00681
00682         # Força que todos estejam em condições
00683         for p in other_players:
00684             p.scom.receive()
00685
00686         for p in other_players:
00687             p.scom.send_immediate(b"(syn)")
00688
00689         # Voltamos ao padrão
00690         self.socket.setblocking(True)
00691
00692     return self.message_queue.get()
00693
00694     # Caso não haja ninguém além dele
00695     if not other_players:
00696         # Sem isso, um loop infinito existiria
00697         return self.receive()
00698
00699     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00700     self.socket.setblocking(False)
00701
00702     while True:
00703         try:
00704             Printing.print_message(".")
00705             self.receive()
00706             break
00707         except BlockingIOError:
00708             pass
00709
00710         # Força que todos estejam em condições
00711         for p in other_players:
00712             p.scom.receive()
00713
00714         for p in other_players:
00715             p.scom.send_immediate(b"(syn)")
00716
00717         # Voltamos ao padrão
00718         self.socket.setblocking(True)
00719
00720     return self.message_queue.get()
00721
00722     # Caso não haja ninguém além dele
00723     if not other_players:
00724         # Sem isso, um loop infinito existiria
00725         return self.receive()
00726
00727     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00728     self.socket.setblocking(False)
00729
00730     while True:
00731         try:
00732             Printing.print_message(".")
00733             self.receive()
00734             break
00735         except BlockingIOError:
00736             pass
00737
00738         # Força que todos estejam em condições
00739         for p in other_players:
00740             p.scom.receive()
00741
00742         for p in other_players:
00743             p.scom.send_immediate(b"(syn)")
00744
00745         # Voltamos ao padrão
00746         self.socket.setblocking(True)
00747
00748     return self.message_queue.get()
00749
00750     # Caso não haja ninguém além dele
00751     if not other_players:
00752         # Sem isso, um loop infinito existiria
00753         return self.receive()
00754
00755     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00756     self.socket.setblocking(False)
00757
00758     while True:
00759         try:
00760             Printing.print_message(".")
00761             self.receive()
00762             break
00763         except BlockingIOError:
00764             pass
00765
00766         # Força que todos estejam em condições
00767         for p in other_players:
00768             p.scom.receive()
00769
00770         for p in other_players:
00771             p.scom.send_immediate(b"(syn)")
00772
00773         # Voltamos ao padrão
00774         self.socket.setblocking(True)
00775
00776     return self.message_queue.get()
00777
00778     # Caso não haja ninguém além dele
00779     if not other_players:
00780         # Sem isso, um loop infinito existiria
00781         return self.receive()
00782
00783     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00784     self.socket.setblocking(False)
00785
00786     while True:
00787         try:
00788             Printing.print_message(".")
00789             self.receive()
00790             break
00791         except BlockingIOError:
00792             pass
00793
00794         # Força que todos estejam em condições
00795         for p in other_players:
00796             p.scom.receive()
00797
00798         for p in other_players:
00799             p.scom.send_immediate(b"(syn)")
00800
00801         # Voltamos ao padrão
00802         self.socket.setblocking(True)
00803
00804     return self.message_queue.get()
00805
00806     # Caso não haja ninguém além dele
00807     if not other_players:
00808         # Sem isso, um loop infinito existiria
00809         return self.receive()
00810
00811     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00812     self.socket.setblocking(False)
00813
00814     while True:
00815         try:
00816             Printing.print_message(".")
00817             self.receive()
00818             break
00819         except BlockingIOError:
00820             pass
00821
00822         # Força que todos estejam em condições
00823         for p in other_players:
00824             p.scom.receive()
00825
00826         for p in other_players:
00827             p.scom.send_immediate(b"(syn)")
00828
00829         # Voltamos ao padrão
00830         self.socket.setblocking(True)
00831
00832     return self.message_queue.get()
00833
00834     # Caso não haja ninguém além dele
00835     if not other_players:
00836         # Sem isso, um loop infinito existiria
00837         return self.receive()
00838
00839     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00840     self.socket.setblocking(False)
00841
00842     while True:
00843         try:
00844             Printing.print_message(".")
00845             self.receive()
00846             break
00847         except BlockingIOError:
00848             pass
00849
00850         # Força que todos estejam em condições
00851         for p in other_players:
00852             p.scom.receive()
00853
00854         for p in other_players:
00855             p.scom.send_immediate(b"(syn)")
00856
00857         # Voltamos ao padrão
00858         self.socket.setblocking(True)
00859
00860     return self.message_queue.get()
00861
00862     # Caso não haja ninguém além dele
00863     if not other_players:
00864         # Sem isso, um loop infinito existiria
00865         return self.receive()
00866
00867     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00868     self.socket.setblocking(False)
00869
00870     while True:
00871         try:
00872             Printing.print_message(".")
00873             self.receive()
00874             break
00875         except BlockingIOError:
00876             pass
00877
00878         # Força que todos estejam em condições
00879         for p in other_players:
00880             p.scom.receive()
00881
00882         for p in other_players:
00883             p.scom.send_immediate(b"(syn)")
00884
00885         # Voltamos ao padrão
00886         self.socket.setblocking(True)
00887
00888     return self.message_queue.get()
00889
00890     # Caso não haja ninguém além dele
00891     if not other_players:
00892         # Sem isso, um loop infinito existiria
00893         return self.receive()
00894
00895     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00896     self.socket.setblocking(False)
00897
00898     while True:
00899         try:
00900             Printing.print_message(".")
00901             self.receive()
00902             break
00903         except BlockingIOError:
00904             pass
00905
00906         # Força que todos estejam em condições
00907         for p in other_players:
00908             p.scom.receive()
00909
00910         for p in other_players:
00911             p.scom.send_immediate(b"(syn)")
00912
00913         # Voltamos ao padrão
00914         self.socket.setblocking(True)
00915
00916     return self.message_queue.get()
00917
00918     # Caso não haja ninguém além dele
00919     if not other_players:
00920         # Sem isso, um loop infinito existiria
00921         return self.receive()
00922
00923     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00924     self.socket.setblocking(False)
00925
00926     while True:
00927         try:
00928             Printing.print_message(".")
00929             self.receive()
00930             break
00931         except BlockingIOError:
00932             pass
00933
00934         # Força que todos estejam em condições
00935         for p in other_players:
00936             p.scom.receive()
00937
00938         for p in other_players:
00939             p.scom.send_immediate(b"(syn)")
00940
00941         # Voltamos ao padrão
00942         self.socket.setblocking(True)
00943
00944     return self.message_queue.get()
00945
00946     # Caso não haja ninguém além dele
00947     if not other_players:
00948         # Sem isso, um loop infinito existiria
00949         return self.receive()
00950
00951     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00952     self.socket.setblocking(False)
00953
00954     while True:
00955         try:
00956             Printing.print_message(".")
00957             self.receive()
00958             break
00959         except BlockingIOError:
00960             pass
00961
00962         # Força que todos estejam em condições
00963         for p in other_players:
00964             p.scom.receive()
00965
00966         for p in other_players:
00967             p.scom.send_immediate(b"(syn)")
00968
00969         # Voltamos ao padrão
00970         self.socket.setblocking(True)
00971
00972     return self.message_queue.get()
00973
00974     # Caso não haja ninguém além dele
00975     if not other_players:
00976         # Sem isso, um loop infinito existiria
00977         return self.receive()
00978
00979     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
00980     self.socket.setblocking(False)
00981
00982     while True:
00983         try:
00984             Printing.print_message(".")
00985             self.receive()
00986             break
00987         except BlockingIOError:
00988             pass
00989
00990         # Força que todos estejam em condições
00991         for p in other_players:
00992             p.scom.receive()
00993
00994         for p in other_players:
00995             p.scom.send_immediate(b"(syn)")
00996
00997         # Voltamos ao padrão
00998         self.socket.setblocking(True)
00999
01000     return self.message_queue.get()
01001
01002     # Caso não haja ninguém além dele
01003     if not other_players:
01004         # Sem isso, um loop infinito existiria
01005         return self.receive()
01006
01007     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01008     self.socket.setblocking(False)
01009
01010     while True:
01011         try:
01012             Printing.print_message(".")
01013             self.receive()
01014             break
01015         except BlockingIOError:
01016             pass
01017
01018         # Força que todos estejam em condições
01019         for p in other_players:
01020             p.scom.receive()
01021
01022         for p in other_players:
01023             p.scom.send_immediate(b"(syn)")
01024
01025         # Voltamos ao padrão
01026         self.socket.setblocking(True)
01027
01028     return self.message_queue.get()
01029
01030     # Caso não haja ninguém além dele
01031     if not other_players:
01032         # Sem isso, um loop infinito existiria
01033         return self.receive()
01034
01035     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01036     self.socket.setblocking(False)
01037
01038     while True:
01039         try:
01040             Printing.print_message(".")
01041             self.receive()
01042             break
01043         except BlockingIOError:
01044             pass
01045
01046         # Força que todos estejam em condições
01047         for p in other_players:
01048             p.scom.receive()
01049
01050         for p in other_players:
01051             p.scom.send_immediate(b"(syn)")
01052
01053         # Voltamos ao padrão
01054         self.socket.setblocking(True)
01055
01056     return self.message_queue.get()
01057
01058     # Caso não haja ninguém além dele
01059     if not other_players:
01060         # Sem isso, um loop infinito existiria
01061         return self.receive()
01062
01063     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01064     self.socket.setblocking(False)
01065
01066     while True:
01067         try:
01068             Printing.print_message(".")
01069             self.receive()
01070             break
01071         except BlockingIOError:
01072             pass
01073
01074         # Força que todos estejam em condições
01075         for p in other_players:
01076             p.scom.receive()
01077
01078         for p in other_players:
01079             p.scom.send_immediate(b"(syn)")
01080
01081         # Voltamos ao padrão
01082         self.socket.setblocking(True)
01083
01084     return self.message_queue.get()
01085
01086     # Caso não haja ninguém além dele
01087     if not other_players:
01088         # Sem isso, um loop infinito existiria
01089         return self.receive()
01090
01091     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01092     self.socket.setblocking(False)
01093
01094     while True:
01095         try:
01096             Printing.print_message(".")
01097             self.receive()
01098             break
01099         except BlockingIOError:
01100             pass
01101
01102         # Força que todos estejam em condições
01103         for p in other_players:
01104             p.scom.receive()
01105
01106         for p in other_players:
01107             p.scom.send_immediate(b"(syn)")
01108
01109         # Voltamos ao padrão
01110         self.socket.setblocking(True)
01111
01112     return self.message_queue.get()
01113
01114     # Caso não haja ninguém além dele
01115     if not other_players:
01116         # Sem isso, um loop infinito existiria
01117         return self.receive()
01118
01119     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01120     self.socket.setblocking(False)
01121
01122     while True:
01123         try:
01124             Printing.print_message(".")
01125             self.receive()
01126             break
01127         except BlockingIOError:
01128             pass
01129
01130         # Força que todos estejam em condições
01131         for p in other_players:
01132             p.scom.receive()
01133
01134         for p in other_players:
01135             p.scom.send_immediate(b"(syn)")
01136
01137         # Voltamos ao padrão
01138         self.socket.setblocking(True)
01139
01140     return self.message_queue.get()
01141
01142     # Caso não haja ninguém além dele
01143     if not other_players:
01144         # Sem isso, um loop infinito existiria
01145         return self.receive()
01146
01147     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01148     self.socket.setblocking(False)
01149
01150     while True:
01151         try:
01152             Printing.print_message(".")
01153             self.receive()
01154             break
01155         except BlockingIOError:
01156             pass
01157
01158         # Força que todos estejam em condições
01159         for p in other_players:
01160             p.scom.receive()
01161
01162         for p in other_players:
01163             p.scom.send_immediate(b"(syn)")
01164
01165         # Voltamos ao padrão
01166         self.socket.setblocking(True)
01167
01168     return self.message_queue.get()
01169
01170     # Caso não haja ninguém além dele
01171     if not other_players:
01172         # Sem isso, um loop infinito existiria
01173         return self.receive()
01174
01175     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01176     self.socket.setblocking(False)
01177
01178     while True:
01179         try:
01180             Printing.print_message(".")
01181             self.receive()
01182             break
01183         except BlockingIOError:
01184             pass
01185
01186         # Força que todos estejam em condições
01187         for p in other_players:
01188             p.scom.receive()
01189
01190         for p in other_players:
01191             p.scom.send_immediate(b"(syn)")
01192
01193         # Voltamos ao padrão
01194         self.socket.setblocking(True)
01195
01196     return self.message_queue.get()
01197
01198     # Caso não haja ninguém além dele
01199     if not other_players:
01200         # Sem isso, um loop infinito existiria
01201         return self.receive()
01202
01203     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01204     self.socket.setblocking(False)
01205
01206     while True:
01207         try:
01208             Printing.print_message(".")
01209             self.receive()
01210             break
01211         except BlockingIOError:
01212             pass
01213
01214         # Força que todos estejam em condições
01215         for p in other_players:
01216             p.scom.receive()
01217
01218         for p in other_players:
01219             p.scom.send_immediate(b"(syn)")
01220
01221         # Voltamos ao padrão
01222         self.socket.setblocking(True)
01223
01224     return self.message_queue.get()
01225
01226     # Caso não haja ninguém além dele
01227     if not other_players:
01228         # Sem isso, um loop infinito existiria
01229         return self.receive()
01230
01231     # Desabilitamos o bloqueio do fluxo de execução por espera de dados no socket
01232     self.socket.setblocking(False)
01233
01234     while True:
01235         try:
01236             Printing.print_message(".")
01237             self.receive()
01238             break
01239         except BlockingIOError:
01240             pass
01241
01242         # Força que todos estejam em condições
01243         for p in other_players:
01244             p.scom.receive()
01245
01246         for p in other_players:
01247             p.scom.send_immediate(b"(syn)")
01248
01249         # Voltamos ao padrão
01250         self.socket.setblocking(True)
01251
01252     return self.message_queue.get()
01253
01254     # Caso não haja ninguém além dele
01255     if not other_players:
01256         # Sem isso, um loop infinito existiria
01257         return self.receive()
01258
01259     # Desabilitamos o bloqueio do fluxo de execução por espera de dados
```

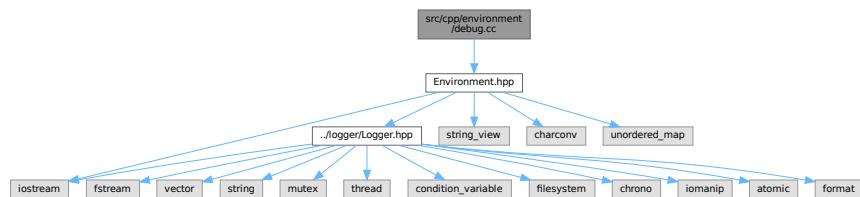
```

00215         ) [0]
00216     ) == 0:
00217         # Se não há nenhum socket para ler neste momento, enviarei
00218         self.message_queue.append(b"(syn)")
00219         self.send_immediate(b"".join(self.message_queue))
00220     else:
00221         Printing.print_message("\nHavia sockets de leitura disponíveis enquanto tentava enviar
00222         fila de mensagens commitadas.", "warning")
00223
00224         self.message_queue.clear() # Limpamos buffer
00225
00226     def clear_queue(self) -> None:
00227         """
00228             @brief Limpará a fila de commits.
00229         """
00230         self.message_queue.clear() # Assim usamos o mesmo ponteiro
00231
00232     # Métodos Derivados
00233     def commit_beam(self, vector_position2d: list, rotation: float):
00234         """
00235             @brief Comando de beam oficial do agente
00236             @param vector_position2d Sequência de dois valores, x e y finais do agente
00237             @param rotation Valor de rotação a ser dado ao robô
00238
00239             assert len(vector_position2d) == 2, "O beam oficial permite apenas posições 2D."
00240             self.commit(
00241                 f"(beam {vector_position2d[0]} {vector_position2d[1]} {rotation})".encode()
00242
00243
00244
00245
00246
00247
00248
00249
00250
00251
00252
00253

```

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 (team left) (unum 1) (sl 3) (sr 2) (t 5.12) (pm BeforeKick← Off))(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))"
```

- int `size` = 1836
- 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` = 1795

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 (team left) (unum 1) (sl 3) (sr 2) (t 5.12) (pm
BeforeKickOff)) (GYR (n torso) (rt 0.01 -0.00 0.00))(ACC (n torso) (a -0.00 -0.00 0.01)) (HJ
(n hj1) (ax 0.00)) (HJ (n hj2) (ax -0.00)) (See (P (team RoboIME) (id 1) (rlowerarm (pol 0.18
-35.30 -22.17)) (llowerarm (pol 0.18 36.49 -21.66))) (G2R (pol 30.92 -19.31 0.55)) (G1R (pol
30.30 -15.73 0.47)) (F1R (pol 29.27 1.62 -1.01)) (F2R (pol 34.87 -33.26 -0.82)) (B (pol 16.91
-32.71 -1.64)) (L (pol 23.88 -53.55 -1.53) (pol 14.22 3.30 -2.23)) (L (pol 34.95 -33.18 -0.←
98) (pol 29.18 1.37 -1.25)) (L (pol 29.20 1.45 -1.09) (pol 1.07 59.96 -29.70)) (L (pol 34.98
-33.31 -0.90) (pol 22.18 -60.01 -1.25)) (L (pol 28.07 -12.48 -0.97) (pol 29.94 -23.73 -1.00))
```

```
(L (pol 28.07 -12.88 -1.02) (pol 29.83 -11.92 -1.07)) (L (pol 29.99 -23.90 -1.00) (pol 31.←
66 -22.86 -0.96)) (L (pol 18.62 -29.50 -1.68) (pol 17.73 -26.93 -1.76)) (L (pol 17.76 -26.80
-1.58) (pol 16.53 -26.27 -1.95)) (L (pol 16.52 -26.24 -1.94) (pol 15.44 -28.34 -2.03)) (L (pol
15.42 -28.55 -1.86) (pol 14.92 -32.55 -1.98)) (L (pol 14.90 -32.54 -2.25) (pol 15.26 -37.←
08 -1.89)) (L (pol 15.28 -37.21 -2.06) (pol 16.31 -39.67 -1.78)) (L (pol 16.28 -39.55 -1.←
64) (pol 17.54 -39.17 -1.67)) (L (pol 17.55 -39.31 -1.67) (pol 18.51 -36.89 -1.61)) (L (pol
18.55 -36.88 -1.69) (pol 18.93 -33.46 -1.78)) (L (pol 18.93 -33.32 -1.51) (pol 18.64 -29.59
-1.54))) (HJ (n raj1) (ax 0.00)) (HJ (n raj2) (ax 0.00)) (HJ (n raj3) (ax 0.00)) (HJ (n raj4) (ax
0.00)) (HJ (n laj1) (ax 0.00)) (HJ (n laj2) (ax -0.00)) (HJ (n laj3) (ax 0.00)) (HJ (n laj4) (ax
-0.00)) (HJ (n rlj1) (ax 0.00)) (HJ (n rlj2) (ax -0.00)) (HJ (n rlj3) (ax -0.00)) (HJ (n rlj4) (ax
-0.00)) (HJ (n rlj5) (ax -0.00)) (HJ (n rlj6) (ax -0.00)) (HJ (n llj1) (ax 0.00)) (HJ (n llj2) (ax
0.00)) (HJ (n llj3) (ax -0.00)) (HJ (n llj4) (ax -0.00)) (HJ (n llj5) (ax -0.00)) (HJ (n llj6) (ax
0.00))"
```

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

7.9.2.2 example1

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

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

7.9.2.3 size

```
int size = 1836
```

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

7.9.2.4 size1

```
int size1 = 1795
```

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

7.10 debug.cc

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

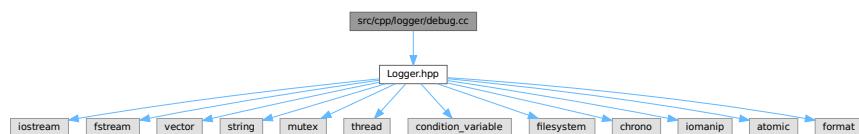
```

00001 #include "Environment.hpp"
00002
00003 const char* example = "(time (now 10.06))(GS (team left) (unum 1) (sl 3) (sr 2) (t 5.12) (pm
BeforeKickOff))(GYR (n torso) (rt 0.01 -0.00 0.00))(ACC (n torso) (a -0.00 -0.00 0.01))(HJ (n hj1) (ax
0.00))(HJ (n hj2) (ax -0.00))(See (P (team RoboIME) (id 1) (rlowerarm (pol 0.18 -35.30 -22.17))
(llowerarm (pol 0.18 36.49 -21.66))) (G2R (pol 30.92 -19.31 0.55)) (G1R (pol 30.30 -15.73 0.47)) (F1R
(pol 29.27 1.62 -1.01)) (F2R (pol 34.87 -33.26 -0.82)) (B (pol 16.91 -32.71 -1.64)) (L (pol 23.88
-53.55 -1.53)) (pol 14.22 3.30 -2.23)) (L (pol 34.95 -33.18 -0.98)) (pol 29.18 1.37 -1.25)) (L (pol
29.20 1.45 -1.09) (pol 1.07 59.96 -29.70)) (L (pol 34.98 -33.31 -0.90)) (pol 22.18 -60.01 -1.25)) (L
(pol 28.07 -12.48 -0.97) (pol 29.94 -23.73 -1.00)) (L (pol 28.07 -12.88 -1.02) (pol 29.83 -11.92
-1.07)) (L (pol 29.99 -23.90 -1.00) (pol 31.66 -22.86 -0.96)) (L (pol 18.62 -29.50 -1.68) (pol 17.73
-26.93 -1.76)) (L (pol 17.76 -26.80 -1.58) (pol 16.53 -26.27 -1.95)) (L (pol 16.52 -26.24 -1.94) (pol
15.44 -28.34 -2.03)) (L (pol 15.42 -28.55 -1.86) (pol 14.92 -32.55 -1.98)) (L (pol 14.90 -32.54 -2.25)
(pol 15.26 -37.08 -1.89)) (L (pol 15.28 -37.21 -2.06) (pol 16.31 -39.67 -1.78)) (L (pol 16.28 -39.55
-1.64) (pol 17.54 -39.17 -1.67)) (L (pol 17.55 -39.31 -1.67) (pol 18.51 -36.89 -1.61)) (L (pol 18.55
-36.88 -1.69) (pol 18.93 -33.46 -1.78)) (L (pol 18.93 -33.32 -1.51) (pol 18.64 -29.59 -1.54)))(HJ (n
raj1) (ax 0.00))(HJ (n raj2) (ax 0.00))(HJ (n raj3) (ax 0.00))(HJ (n raj4) (ax 0.00))(HJ (n laj1) (ax
0.00))(HJ (n laj2) (ax 0.00))(HJ (n laj3) (ax 0.00))(HJ (n laj4) (ax 0.00))(HJ (n r1j1) (ax
0.00))(HJ (n r1j2) (ax 0.00))(HJ (n r1j3) (ax 0.00))(HJ (n r1j4) (ax 0.00))(HJ (n r1j5) (ax
0.00))(HJ (n r1j6) (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 = 1836;
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 r1j3) (ax
0.01))(HJ (n r1j4) (ax 0.00))(HJ (n r1j5) (ax 0.00))(FRP (n rf) (c -0.02 -0.00 -0.02) (f -0.02 -0.17
22.52))(HJ (n r1j6) (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 = 1795;
00008
00009 int
0010 main(){
0011
0012     std::string_view message_from_server(example1, size1);
0013     Environment ex = Environment(Logger::get());
0014     ex.update_from_server(message_from_server);
0015
0016     return 0;
0017 }

```

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

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



Functions

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

7.11.1 Function Documentation

7.11.1.1 main()

```
int main ( )
```

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

7.11.1.2 tarefaPesada()

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

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

7.12 debug.cc

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

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

```

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

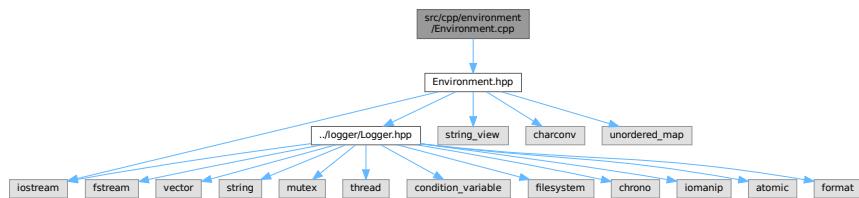
```

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

```

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

#include "Environment.hpp"
Include dependency graph for Environment.cpp:



7.14 Environment.cpp

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

```

00001 #include "Environment.hpp"
00002
00003 /* --- Definições de Estáticos Não Literais --- */
00004 const std::unordered_map<
00005     std::string,
00006     std::array<Environment::PlayMode, 2>,
00007     Environment::Enabler_Stringview_Hash,
00008     std::equal_to<>>
00009 > Environment::play_modes = {
00010     // --- Neutros (LEFT e RIGHT veem o mesmo modo) ---
00011     {"BeforeKickOff", {Environment::PlayMode::BEFORE_KICKOFF, Environment::PlayMode::BEFORE_KICKOFF}},
00012     {"GameOver", {Environment::PlayMode::GAME_OVER, Environment::PlayMode::GAME_OVER}},
00013     {"PlayOn", {Environment::PlayMode::PLAY_ON, Environment::PlayMode::PLAY_ON}},
00014
00015     // --- LEFT Kick Events (LEFT é o nosso time, RIGHT é o time deles) ---
00016
00017     // LEFT_PLAY_MODE_TO_ID: KickOff_Left -> OUR_KICKOFF (0)
00018     // RIGHT_PLAY_MODE_TO_ID: KickOff_Left -> THEIR_KICKOFF (9)
00019     {"KickOff_Left", {Environment::PlayMode::OUR_KICKOFF,
00020                      Environment::PlayMode::THEIR_KICKOFF}},
00021
00022     // LEFT: OUR_KICK_IN (1); RIGHT: THEIR_KICK_IN (10)
00023     {"KickIn_Left", {Environment::PlayMode::OUR_KICK_IN,
00024                      Environment::PlayMode::THEIR_KICK_IN}},
00025
00026     // LEFT: OUR_CORNER_KICK (2); RIGHT: THEIR_CORNER_KICK (11)
00027     {"corner_kick_left", {Environment::PlayMode::OUR_CORNER_KICK,
00028                           Environment::PlayMode::THEIR_CORNER_KICK}},
00029
00030     // LEFT: OUR_GOAL_KICK (3); RIGHT: THEIR_GOAL_KICK (12)

```

```

00028     {"goal_kick_left",           {Environment::PlayMode::OUR_GOAL_KICK,
00029      Environment::PlayMode::THEIR_GOAL_KICK}},
00030     // LEFT: OUR_FREE_KICK (4); RIGHT: THEIR_FREE_KICK (13)
00031     {"free_kick_left",          {Environment::PlayMode::OUR_FREE_KICK,
00032      Environment::PlayMode::THEIR_FREE_KICK}},
00033     // LEFT: OUR_PASS (5); RIGHT: THEIR_PASS (14)
00034     {"pass_left",              {Environment::PlayMode::OUR_PASS,
00035      Environment::PlayMode::THEIR_PASS}},
00036     // LEFT: OUR_DIR_FREE_KICK (6); RIGHT: THEIR_DIR_FREE_KICK (15)
00037     {"direct_free_kick_left",   {Environment::PlayMode::OUR_DIR_FREE_KICK,
00038      Environment::PlayMode::THEIR_DIR_FREE_KICK}},
00039     // LEFT: OUR_GOAL (7); RIGHT: THEIR_GOAL (16)
00040     {"Goal_Left",               {Environment::PlayMode::OUR_GOAL,
00041      Environment::PlayMode::THEIR_GOAL}},
00042     // LEFT: OUR_OFSIDE (8); RIGHT: THEIR_OFSIDE (17)
00043     {"offside_left",            {Environment::PlayMode::OUR_OFSIDE,
00044      Environment::PlayMode::THEIR_OFSIDE}},
00045     // --- RIGHT Kick Events (RIGHT é o nosso time, LEFT é o time deles) ---
00046
00047     // LEFT_PLAY_MODE_TO_ID: KickOff_Right -> THEIR_KICKOFF (9)
00048     // RIGHT_PLAY_MODE_TO_ID: KickOff_Right -> OUR_KICKOFF (0)
00049     {"KickOff_Right",           {Environment::PlayMode::THEIR_KICKOFF,
00050      Environment::PlayMode::OUR_KICKOFF}},
00051     // LEFT: THEIR_KICK_IN (10); RIGHT: OUR_KICK_IN (1)
00052     {"KickIn_Right",            {Environment::PlayMode::THEIR_KICK_IN,
00053      Environment::PlayMode::OUR_KICK_IN}},
00054     // LEFT: THEIR_CORNER_KICK (11); RIGHT: OUR_CORNER_KICK (2)
00055     {"corner_kick_right",       {Environment::PlayMode::THEIR_CORNER_KICK,
00056      Environment::PlayMode::OUR_CORNER_KICK}},
00057     // LEFT: THEIR_GOAL_KICK (12); RIGHT: OUR_GOAL_KICK (3)
00058     {"goal_kick_right",         {Environment::PlayMode::THEIR_GOAL_KICK,
00059      Environment::PlayMode::OUR_GOAL_KICK}},
00060     // LEFT: THEIR_FREE_KICK (13); RIGHT: OUR_FREE_KICK (4)
00061     {"free_kick_right",         {Environment::PlayMode::THEIR_FREE_KICK,
00062      Environment::PlayMode::OUR_FREE_KICK}},
00063     // LEFT: THEIR_PASS (14); RIGHT: OUR_PASS (5)
00064     {"pass_right",              {Environment::PlayMode::THEIR_PASS,
00065      Environment::PlayMode::OUR_PASS}},
00066     // LEFT: THEIR_DIR_FREE_KICK (15); RIGHT: OUR_DIR_FREE_KICK (6)
00067     {"direct_free_kick_right",  {Environment::PlayMode::THEIR_DIR_FREE_KICK,
00068      Environment::PlayMode::OUR_DIR_FREE_KICK}},
00069     // LEFT: THEIR_GOAL (16); RIGHT: OUR_GOAL (7)
00070     {"Goal_Right",               {Environment::PlayMode::THEIR_GOAL,
00071      Environment::PlayMode::OUR_GOAL}},
00072     // LEFT: THEIR_OFSIDE (17); RIGHT: OUR_OFSIDE (8)
00073     {"offside_right",            {Environment::PlayMode::THEIR_OFSIDE,
00074      Environment::PlayMode::OUR_OFSIDE}};
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086

```

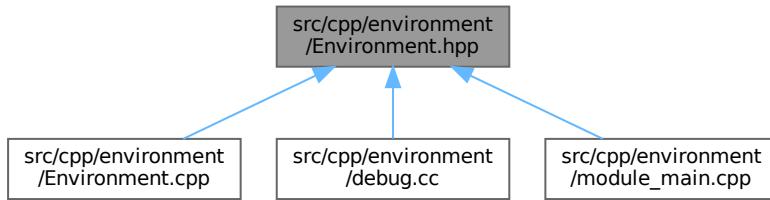
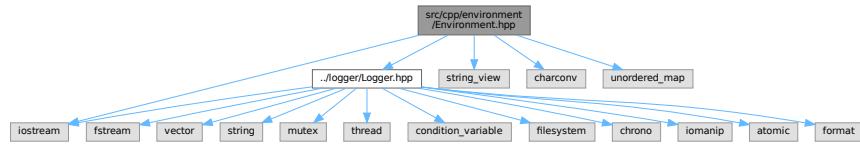
7.15 src/cpp/environment/Environment.hpp File Reference

```

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

```

```
#include <charconv>
#include <unordered_map>
Include dependency graph for Environment.hpp:
```



Classes

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

Macros

- #define [True](#) true
< std::from_chars
- #define [False](#) false

7.15.1 Macro Definition Documentation

7.15.1.1 [False](#)

```
#define False false
```

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

7.15.1.2 True

```
#define True true
```

```
< std::from_chars
```

Definition at line 9 of file Environment.hpp.

7.16 Environment.hpp

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

```
00001 #pragma once
00002
00003 #include "../logger/Logger.hpp"
00004 #include <iostream>
00005 #include <string_view>
00006 #include <charconv>
00007 #include <unordered_map>
00008
00009 #define True true
00010 #define False false
00011
00018 class Environment {
00019 public:
00020
00021     Logger& logger;
00022     Environment(
00023         Logger& logger
00024     ) : logger(logger) {}
00028
00029     /* -- Definição de Ferramentas que serão amplamente Usadas -- */
00031     enum class PlayMode : uint8_t {
00032         // Ao nosso favor
00033         OUR_KICKOFF = 0,
00034         OUR_KICK_IN = 1,
00035         OUR_CORNER_KICK = 2,
00036         OUR_GOAL_KICK = 3,
00037         OUR_FREE_KICK = 4,
00038         OUR_PASS = 5,
00039         OUR_DIR_FREE_KICK = 6,
00040         OUR_GOAL = 7,
00041         OUR_OFSIDE = 8,
00042
00043         // Ao favor deles
00044         THEIR_KICKOFF = 9,
00045         THEIR_KICK_IN = 10,
00046         THEIR_CORNER_KICK = 11,
00047         THEIR_GOAL_KICK = 12,
00048         THEIR_FREE_KICK = 13,
00049         THEIR_PASS = 14,
00050         THEIR_DIR_FREE_KICK = 15,
00051         THEIR_GOAL = 16,
00052         THEIR_OFSIDE = 17,
00053
00054         // Neutros
00055         BEFORE_KICKOFF = 18,
00056         GAME_OVER = 19,
00057         PLAY_ON = 20
00058     };
00059     enum class PlayModeGroup : uint8_t {
00060         OUR_KICK = 0,           // É nossa vez de chutar parado
00061         THEIR_KICK = 1,          // É vez deles de chutar parado
00062         ACTIVE_BEAM = 2,          // Podemos usar o comando beam (teleporte)
00063         PASSIVE_BEAM = 3,          // Devemos esperar (beam passivo/goalie)
00064         OTHER = 4,              // Jogo rolando ou parado sem ação específica
00065     };
00066     struct Enabler_Stringview_Hash {
00067         using is_transparent = void;
00068         // Sobrecarga do operador para hashing de std::string
00069         ::size_t operator()(const std::string& s) const { return std::hash<std::string>{}(s); }
00070         // Sobrecarga do operador para hashing de std::string_view (para pesquisa)
00071         ::size_t operator()(std::string_view sv) const { return std::hash<std::string_view>{}(sv); }
00072     };
00073     static const
00074     std::unordered_map<std::string,
00075                         std::array<PlayMode, 2>,
00076                         Enabler_Stringview_Hash,
```

```

00077             std::equal_to<>
00078         >play_modes;
00079
00080     /* Atributos Públicos de Ambiente */
00081     float time_server;
00082     float time_match;
00083     uint8_t goals_scored;
00084     uint8_t goals_conceded;
00085     uint8_t unum;
00086     bool is_left;
00087     PlayMode current_mode;
00088
00089     /* Métodos Inerentes a Execução da Aplicação */
00090
00091     /* ----- Parser de Mensagem do Servidor ----- */
00092
00093     class Parsing {
00094     private:
00095         const char* buffer = nullptr;
00096         const char* end    = nullptr;
00097         Environment* env   = nullptr;
00098
00099     public:
00100     /* Métodos Simples de Cursor */
00101
00102     Parsing(
00103         std::string_view message,
00104         Environment* env
00105     ) :
00106         buffer(message.data()),
00107         end(message.data() + message.size()),
00108         env(env)
00109     {}
00110
00111     bool
00112     skip_until_char(char caract){
00113         while(*this->buffer != caract){
00114             if(this->buffer > this->end){ return False; }
00115             this->buffer++;
00116         }
00117         this->buffer++;
00118         return True;
00119     }
00120
00121     std::string_view
00122     get_str(){
00123         while(*this->buffer == ' ' || *this->buffer == '(' || *this->buffer == ')'){
00124             this->buffer++;
00125         }
00126         const char* value_start = this->buffer;
00127         while(*this->buffer != ' ' && *this->buffer != ')'){ this->buffer++; }
00128         return std::string_view(value_start, ::size_t(this->buffer - value_start));
00129     }
00130
00131     template<typename T>
00132     bool
00133     get_value(T& out){
00134         const char* value_start = this->buffer;
00135         while(*this->buffer != ' ' && *this->buffer != ')'){ this->buffer++; }
00136         return std::from_chars(value_start, this->buffer++, out).ec == std::errc{};
00137     }
00138
00139     bool
00140     advance(int n = 1){ if((this->buffer + n) > this->end){ return False; } this->buffer += n;
00141     return True; }
00142
00143     std::string
00144     get(){
00145         return std::string(std::string_view(this->buffer - 30, 60));
00146     }
00147
00148     bool
00149     is_valid(){ return this->buffer < this->end; }
00150
00151     /* -- Métodos de Parsing -- */
00152
00153     void
00154     parse_time(){
00155         /*
00156             Buffer está aqui.
00157             |
00158             v
00159             ' (now 10.03)'
00160             */
00161         this->advance(5);
00162         this->get_value(env->time_server);
00163         this->advance();
00164     }

```

```

00199
00206     void
00207     parse_gamestate() {
00208
00209         std::string_view lower_tag;
00210         while(True) {
00211             lower_tag = this->get_str();
00212
00213             switch(lower_tag[0]) {
00214
00215                 case 's': {
00216                     this->get_value( (lower_tag[1] == '1') ? env->goals_scored :
00217                         env->goals_conceded );
00218                     break;
00219                 }
00220
00221                 case 'p': {
00222                     // É garantido que já tenhamos tido is_left
00223                     lower_tag = this->get_str();
00224                     auto it = play_modes.find(lower_tag);
00225                     if( it != play_modes.end() ) { env->current_mode = it->second[env->is_left]; }
00226                     break;
00227                 }
00228
00229                 case 't': {
00230                     if(lower_tag[1] == 'i'){ this->get_value(env->time_match); }
00231                     else{ env->is_left = this->get_str()[0] == '1'; }
00232                     break;
00233                 }
00234
00235                 case 'u': {
00236                     this->get_value(env->unum);
00237                     break;
00238                 }
00239
00240                 default: {
00241                     env->logger.warn("[{}]{Flag Desconhecida Encontrada em 'GS': {} \t Buffer Neste
00242 momento: {}, env->unum, lower_tag, this->buffer);
00243                     break;
00244                 }
00245
00246             if(*this->buffer == ')') { break; }
00247         }
00248
00249         void
00250         parse_gyroscope() {
00251
00252             // Só há uma tag aqui. Logo, não é necessário loop e busca por tentativas.
00253             this->advance(14); // Colocamos 13, pois nunca se sabe se virá um '-' para nos atrapalhar.
00254
00255             // Devemos usar Eigen
00256             float value;
00257             for(int i = 0; i < 3; i++){ this->get_value(value); }
00258         }
00259
00260         void
00261         parse_accelerometer() {
00262
00263             this->advance(13);
00264             float value;
00265             for(int i = 0; i < 3; i++){ this->get_value(value); }
00266         }
00267
00268         void
00269         parse_vision() {
00270
00271             std::string_view lower_tag;
00272             while(True) {
00273
00274                 lower_tag = this->get_str();
00275
00276                 switch(lower_tag[0]) {
00277
00278                     case 'P':
00279                         while(True) {
00280
00281                             lower_tag = this->get_str();
00282
00283                             switch(lower_tag[0]) {
00284
00285                                 case 't': {
00286                                     this->get_str();
00287                                     break;
00288                                 }
00289
00290                             }
00291
00292                         }
00293
00294                 }
00295             }
00296         }
00297     }
00298
00299     void
00300     parse_gyro() {
00301
00302         std::string_view lower_tag;
00303
00304         switch(lower_tag[0]) {
00305
00306             case 't': {
00307                 this->get_str();
00308                 break;
00309             }
00310
00311         }
00312     }
00313
00314     void
00315     parse_imu() {
00316
00317         std::string_view lower_tag;
00318
00319         switch(lower_tag[0]) {
00320
00321             case 't': {
00322                 this->get_str();
00323                 break;
00324             }
00325
00326         }
00327     }
00328
00329     void
00330     parse_light() {
00331
00332         std::string_view lower_tag;
00333
00334         switch(lower_tag[0]) {
00335
00336             case 't': {
00337                 this->get_str();
00338                 break;
00339             }
00340
00341         }
00342     }
00343
00344     void
00345     parse_sonar() {
00346
00347         std::string_view lower_tag;
00348
00349         switch(lower_tag[0]) {
00350
00351             case 't': {
00352                 this->get_str();
00353                 break;
00354             }
00355
00356         }
00357     }
00358
00359     void
00360     parse_stick() {
00361
00362         std::string_view lower_tag;
00363
00364         switch(lower_tag[0]) {
00365
00366             case 't': {
00367                 this->get_str();
00368                 break;
00369             }
00370
00371         }
00372     }
00373
00374     void
00375     parse_touch() {
00376
00377         std::string_view lower_tag;
00378
00379         switch(lower_tag[0]) {
00380
00381             case 't': {
00382                 this->get_str();
00383                 break;
00384             }
00385
00386         }
00387     }
00388
00389     void
00390     parse_vision() {
00391
00392         std::string_view lower_tag;
00393
00394         switch(lower_tag[0]) {
00395
00396             case 't': {
00397                 this->get_str();
00398                 break;
00399             }
00400
00401         }
00402     }
00403
00404     void
00405     parse_gyroscope() {
00406
00407         std::string_view lower_tag;
00408
00409         switch(lower_tag[0]) {
00410
00411             case 't': {
00412                 this->get_str();
00413                 break;
00414             }
00415
00416         }
00417     }
00418
00419     void
00420     parse_accelerometer() {
00421
00422         std::string_view lower_tag;
00423
00424         switch(lower_tag[0]) {
00425
00426             case 't': {
00427                 this->get_str();
00428                 break;
00429             }
00430
00431         }
00432     }
00433
00434     void
00435     parse_imu() {
00436
00437         std::string_view lower_tag;
00438
00439         switch(lower_tag[0]) {
00440
00441             case 't': {
00442                 this->get_str();
00443                 break;
00444             }
00445
00446         }
00447     }
00448
00449     void
00450     parse_light() {
00451
00452         std::string_view lower_tag;
00453
00454         switch(lower_tag[0]) {
00455
00456             case 't': {
00457                 this->get_str();
00458                 break;
00459             }
00460
00461         }
00462     }
00463
00464     void
00465     parse_sonar() {
00466
00467         std::string_view lower_tag;
00468
00469         switch(lower_tag[0]) {
00470
00471             case 't': {
00472                 this->get_str();
00473                 break;
00474             }
00475
00476         }
00477     }
00478
00479     void
00480     parse_stick() {
00481
00482         std::string_view lower_tag;
00483
00484         switch(lower_tag[0]) {
00485
00486             case 't': {
00487                 this->get_str();
00488                 break;
00489             }
00490
00491         }
00492     }
00493
00494     void
00495     parse_touch() {
00496
00497         std::string_view lower_tag;
00498
00499         switch(lower_tag[0]) {
00500
00501             case 't': {
00502                 this->get_str();
00503                 break;
00504             }
00505
00506         }
00507     }
00508
00509     void
00510     parse_vision() {
00511
00512         std::string_view lower_tag;
00513
00514         switch(lower_tag[0]) {
00515
00516             case 't': {
00517                 this->get_str();
00518                 break;
00519             }
00520
00521         }
00522     }
00523
00524     void
00525     parse_gyroscope() {
00526
00527         std::string_view lower_tag;
00528
00529         switch(lower_tag[0]) {
00530
00531             case 't': {
00532                 this->get_str();
00533                 break;
00534             }
00535
00536         }
00537     }
00538
00539     void
00540     parse_accelerometer() {
00541
00542         std::string_view lower_tag;
00543
00544         switch(lower_tag[0]) {
00545
00546             case 't': {
00547                 this->get_str();
00548                 break;
00549             }
00550
00551         }
00552     }
00553
00554     void
00555     parse_imu() {
00556
00557         std::string_view lower_tag;
00558
00559         switch(lower_tag[0]) {
00560
00561             case 't': {
00562                 this->get_str();
00563                 break;
00564             }
00565
00566         }
00567     }
00568
00569     void
00570     parse_light() {
00571
00572         std::string_view lower_tag;
00573
00574         switch(lower_tag[0]) {
00575
00576             case 't': {
00577                 this->get_str();
00578                 break;
00579             }
00580
00581         }
00582     }
00583
00584     void
00585     parse_sonar() {
00586
00587         std::string_view lower_tag;
00588
00589         switch(lower_tag[0]) {
00590
00591             case 't': {
00592                 this->get_str();
00593                 break;
00594             }
00595
00596         }
00597     }
00598
00599     void
00600     parse_stick() {
00601
00602         std::string_view lower_tag;
00603
00604         switch(lower_tag[0]) {
00605
00606             case 't': {
00607                 this->get_str();
00608                 break;
00609             }
00610
00611         }
00612     }
00613
00614     void
00615     parse_touch() {
00616
00617         std::string_view lower_tag;
00618
00619         switch(lower_tag[0]) {
00620
00621             case 't': {
00622                 this->get_str();
00623                 break;
00624             }
00625
00626         }
00627     }
00628
00629     void
00630     parse_vision() {
00631
00632         std::string_view lower_tag;
00633
00634         switch(lower_tag[0]) {
00635
00636             case 't': {
00637                 this->get_str();
00638                 break;
00639             }
00640
00641         }
00642     }
00643
00644     void
00645     parse_gyroscope() {
00646
00647         std::string_view lower_tag;
00648
00649         switch(lower_tag[0]) {
00650
00651             case 't': {
00652                 this->get_str();
00653                 break;
00654             }
00655
00656         }
00657     }
00658
00659     void
00660     parse_accelerometer() {
00661
00662         std::string_view lower_tag;
00663
00664         switch(lower_tag[0]) {
00665
00666             case 't': {
00667                 this->get_str();
00668                 break;
00669             }
00670
00671         }
00672     }
00673
00674     void
00675     parse_imu() {
00676
00677         std::string_view lower_tag;
00678
00679         switch(lower_tag[0]) {
00680
00681             case 't': {
00682                 this->get_str();
00683                 break;
00684             }
00685
00686         }
00687     }
00688
00689     void
00690     parse_light() {
00691
00692         std::string_view lower_tag;
00693
00694         switch(lower_tag[0]) {
00695
00696             case 't': {
00697                 this->get_str();
00698                 break;
00699             }
00700
00701         }
00702     }
00703
00704     void
00705     parse_sonar() {
00706
00707         std::string_view lower_tag;
00708
00709         switch(lower_tag[0]) {
00710
00711             case 't': {
00712                 this->get_str();
00713                 break;
00714             }
00715
00716         }
00717     }
00718
00719     void
00720     parse_stick() {
00721
00722         std::string_view lower_tag;
00723
00724         switch(lower_tag[0]) {
00725
00726             case 't': {
00727                 this->get_str();
00728                 break;
00729             }
00730
00731         }
00732     }
00733
00734     void
00735     parse_touch() {
00736
00737         std::string_view lower_tag;
00738
00739         switch(lower_tag[0]) {
00740
00741             case 't': {
00742                 this->get_str();
00743                 break;
00744             }
00745
00746         }
00747     }
00748
00749     void
00750     parse_vision() {
00751
00752         std::string_view lower_tag;
00753
00754         switch(lower_tag[0]) {
00755
00756             case 't': {
00757                 this->get_str();
00758                 break;
00759             }
00760
00761         }
00762     }
00763
00764     void
00765     parse_gyroscope() {
00766
00767         std::string_view lower_tag;
00768
00769         switch(lower_tag[0]) {
00770
00771             case 't': {
00772                 this->get_str();
00773                 break;
00774             }
00775
00776         }
00777     }
00778
00779     void
00780     parse_accelerometer() {
00781
00782         std::string_view lower_tag;
00783
00784         switch(lower_tag[0]) {
00785
00786             case 't': {
00787                 this->get_str();
00788                 break;
00789             }
00790
00791         }
00792     }
00793
00794     void
00795     parse_imu() {
00796
00797         std::string_view lower_tag;
00798
00799         switch(lower_tag[0]) {
00800
00801             case 't': {
00802                 this->get_str();
00803                 break;
00804             }
00805
00806         }
00807     }
00808
00809     void
00810     parse_light() {
00811
00812         std::string_view lower_tag;
00813
00814         switch(lower_tag[0]) {
00815
00816             case 't': {
00817                 this->get_str();
00818                 break;
00819             }
00820
00821         }
00822     }
00823
00824     void
00825     parse_sonar() {
00826
00827         std::string_view lower_tag;
00828
00829         switch(lower_tag[0]) {
00830
00831             case 't': {
00832                 this->get_str();
00833                 break;
00834             }
00835
00836         }
00837     }
00838
00839     void
00840     parse_stick() {
00841
00842         std::string_view lower_tag;
00843
00844         switch(lower_tag[0]) {
00845
00846             case 't': {
00847                 this->get_str();
00848                 break;
00849             }
00850
00851         }
00852     }
00853
00854     void
00855     parse_touch() {
00856
00857         std::string_view lower_tag;
00858
00859         switch(lower_tag[0]) {
00860
00861             case 't': {
00862                 this->get_str();
00863                 break;
00864             }
00865
00866         }
00867     }
00868
00869     void
00870     parse_vision() {
00871
00872         std::string_view lower_tag;
00873
00874         switch(lower_tag[0]) {
00875
00876             case 't': {
00877                 this->get_str();
00878                 break;
00879             }
00880
00881         }
00882     }
00883
00884     void
00885     parse_gyroscope() {
00886
00887         std::string_view lower_tag;
00888
00889         switch(lower_tag[0]) {
00890
00891             case 't': {
00892                 this->get_str();
00893                 break;
00894             }
00895
00896         }
00897     }
00898
00899     void
00900     parse_accelerometer() {
00901
00902         std::string_view lower_tag;
00903
00904         switch(lower_tag[0]) {
00905
00906             case 't': {
00907                 this->get_str();
00908                 break;
00909             }
00910
00911         }
00912     }
00913
00914     void
00915     parse_imu() {
00916
00917         std::string_view lower_tag;
00918
00919         switch(lower_tag[0]) {
00920
00921             case 't': {
00922                 this->get_str();
00923                 break;
00924             }
00925
00926         }
00927     }
00928
00929     void
00930     parse_light() {
00931
00932         std::string_view lower_tag;
00933
00934         switch(lower_tag[0]) {
00935
00936             case 't': {
00937                 this->get_str();
00938                 break;
00939             }
00940
00941         }
00942     }
00943
00944     void
00945     parse_sonar() {
00946
00947         std::string_view lower_tag;
00948
00949         switch(lower_tag[0]) {
00950
00951             case 't': {
00952                 this->get_str();
00953                 break;
00954             }
00955
00956         }
00957     }
00958
00959     void
00960     parse_stick() {
00961
00962         std::string_view lower_tag;
00963
00964         switch(lower_tag[0]) {
00965
00966             case 't': {
00967                 this->get_str();
00968                 break;
00969             }
00970
00971         }
00972     }
00973
00974     void
00975     parse_touch() {
00976
00977         std::string_view lower_tag;
00978
00979         switch(lower_tag[0]) {
00980
00981             case 't': {
00982                 this->get_str();
00983                 break;
00984             }
00985
00986         }
00987     }
00988
00989     void
00990     parse_vision() {
00991
00992         std::string_view lower_tag;
00993
00994         switch(lower_tag[0]) {
00995
00996             case 't': {
00997                 this->get_str();
00998                 break;
00999             }
01000
01001         }
01002     }
01003
01004     void
01005     parse_gyroscope() {
01006
01007         std::string_view lower_tag;
01008
01009         switch(lower_tag[0]) {
01010
01011             case 't': {
01012                 this->get_str();
01013                 break;
01014             }
01015
01016         }
01017     }
01018
01019     void
01020     parse_accelerometer() {
01021
01022         std::string_view lower_tag;
01023
01024         switch(lower_tag[0]) {
01025
01026             case 't': {
01027                 this->get_str();
01028                 break;
01029             }
01030
01031         }
01032     }
01033
01034     void
01035     parse_imu() {
01036
01037         std::string_view lower_tag;
01038
01039         switch(lower_tag[0]) {
01040
01041             case 't': {
01042                 this->get_str();
01043                 break;
01044             }
01045
01046         }
01047     }
01048
01049     void
01050     parse_light() {
01051
01052         std::string_view lower_tag;
01053
01054         switch(lower_tag[0]) {
01055
01056             case 't': {
01057                 this->get_str();
01058                 break;
01059             }
01060
01061         }
01062     }
01063
01064     void
01065     parse_sonar() {
01066
01067         std::string_view lower_tag;
01068
01069         switch(lower_tag[0]) {
01070
01071             case 't': {
01072                 this->get_str();
01073                 break;
01074             }
01075
01076         }
01077     }
01078
01079     void
01080     parse_stick() {
01081
01082         std::string_view lower_tag;
01083
01084         switch(lower_tag[0]) {
01085
01086             case 't': {
01087                 this->get_str();
01088                 break;
01089             }
01090
01091         }
01092     }
01093
01094     void
01095     parse_touch() {
01096
01097         std::string_view lower_tag;
01098
01099         switch(lower_tag[0]) {
01100
01101             case 't': {
01102                 this->get_str();
01103                 break;
01104             }
01105
01106         }
01107     }
01108
01109     void
01110     parse_vision() {
01111
01112         std::string_view lower_tag;
01113
01114         switch(lower_tag[0]) {
01115
01116             case 't': {
01117                 this->get_str();
01118                 break;
01119             }
01120
01121         }
01122     }
01123
01124     void
01125     parse_gyroscope() {
01126
01127         std::string_view lower_tag;
01128
01129         switch(lower_tag[0]) {
01130
01131             case 't': {
01132                 this->get_str();
01133                 break;
01134             }
01135
01136         }
01137     }
01138
01139     void
01140     parse_accelerometer() {
01141
01142         std::string_view lower_tag;
01143
01144         switch(lower_tag[0]) {
01145
01146             case 't': {
01147                 this->get_str();
01148                 break;
01149             }
01150
01151         }
01152     }
01153
01154     void
01155     parse_imu() {
01156
01157         std::string_view lower_tag;
01158
01159         switch(lower_tag[0]) {
01160
01161             case 't': {
01162                 this->get_str();
01163                 break;
01164             }
01165
01166         }
01167     }
01168
01169     void
01170     parse_light() {
01171
01172         std::string_view lower_tag;
01173
01174         switch(lower_tag[0]) {
01175
01176             case 't': {
01177                 this->get_str();
01178                 break;
01179             }
01180
01181         }
01182     }
01183
01184     void
01185     parse_sonar() {
01186
01187         std::string_view lower_tag;
01188
01189         switch(lower_tag[0]) {
01190
01191             case 't': {
01192                 this->get_str();
01193                 break;
01194             }
01195
01196         }
01197     }
01198
01199     void
01200     parse_stick() {
01201
01202         std::string_view lower_tag;
01203
01204         switch(lower_tag[0]) {
01205
01206             case 't': {
01207                 this->get_str();
01208                 break;
01209             }
01210
01211         }
01212     }
01213
01214     void
01215     parse_touch() {
01216
01217         std::string_view lower_tag;
01218
01219         switch(lower_tag[0]) {
01220
01221             case 't': {
01222                 this->get_str();
01223                 break;
01224             }
01225
01226         }
01227     }
01228
01229     void
01230     parse_vision() {
01231
01232         std::string_view lower_tag;
01233
01234         switch(lower_tag[0]) {
01235
01236             case 't': {
01237                 this->get_str();
01238                 break;
01239             }
01240
01241         }
01242     }
01243
01244     void
01245     parse_gyroscope() {
01246
01247         std::string_view lower_tag;
01248
01249         switch(lower_tag[0]) {
01250
01251             case 't': {
01252                 this->get_str();
01253                 break;
01254             }
01255
01256         }
01257     }
01258
01259     void
01260     parse_accelerometer() {
01261
01262         std::string_view lower_tag;
01263
01264         switch(lower_tag[0]) {
01265
01266             case 't': {
01267                 this->get_str();
01268                 break;
01269             }
01270
01271         }
01272     }
01273
01274     void
01275     parse_imu() {
01276
01277         std::string_view lower_tag;
01278
01279         switch(lower_tag[0]) {
01280
01281             case 't': {
01282                 this->get_str();
01283                 break;
01284             }
01285
01286         }
01287     }
01288
01289     void
01290     parse_light() {
01291
01292         std::string_view lower_tag;
01293
01294         switch(lower_tag[0]) {
01295
01296             case 't': {
01297                 this->get_str();
01298                 break;
01299             }
01300
01301         }
01302     }
01303
01304     void
01305     parse_sonar() {
01306
01307         std::string_view lower_tag;
01308
01309         switch(lower_tag[0]) {
01310
01311             case 't': {
01312                 this->get_str();
01313                 break;
01314             }
01315
01316         }
01317     }
01318
01319     void
01320     parse_stick() {
01321
01322         std::string_view lower_tag;
01323
01324         switch(lower_tag[0]) {
01325
01326             case 't': {
01327                 this->get_str();
01328                 break;
01329             }
01330
01331         }
01332     }
01333
01334     void
01335     parse_touch() {
01336
01337         std::string_view lower_tag;
01338
01339         switch(lower_tag[0]) {
01340
01341             case 't': {
01342                 this->get_str();
01343                 break;
01344             }
01345
01346         }
01347     }
01348
01349     void
01350     parse_vision() {
01351
01352         std::string_view lower_tag;
01353
01354         switch(lower_tag[0]) {
01355
01356             case 't': {
01357                 this->get_str();
01358                 break;
01359             }
01360
01361         }
01362     }
01363
01364     void
01365     parse_gyroscope() {
01366
01367         std::string_view lower_tag;
01368
01369         switch(lower_tag[0]) {
01370
01371             case 't': {
01372                 this->get_str();
01373                 break;
01374             }
01375
01376         }
01377     }
01378
01379     void
01380     parse_accelerometer() {
01381
01382         std::string_view lower_tag;
01383
01384         switch(lower_tag[0]) {
01385
01386             case 't': {
01387                 this->get_str();
01388                 break;
01389             }
01390
01391         }
01392     }
01393
01394     void
01395     parse_imu() {
01396
01397         std::string_view lower_tag;
01398
01399         switch(lower_tag[0]) {
01400
01401             case 't': {
01402                 this->get_str();
01403                 break;
01404             }
01405
01406         }
01407     }
01408
01409     void
01410     parse_light() {
01411
01412         std::string_view lower_tag;
01413
01414         switch(lower_tag[0]) {
01415
01416             case 't': {
01417                 this->get_str();
01418                 break;
01419             }
01420
01421         }
01422     }
01423
01424     void
01425     parse_sonar() {
01426
01427         std::string_view lower_tag;
01428
01429         switch(lower_tag[0]) {
01430
01431             case 't': {
01432                 this->get_str();
01433                 break;
01434             }
01435
01436         }
01437     }
01438
01439     void
01440     parse_stick() {
01441
01442         std::string_view lower_tag;
01443
01444         switch(lower_tag[0]) {
01445
01446             case 't': {
01447                 this->get_str();
01448                 break;
01449             }
01450
01451         }
01452     }
01453
01454     void
01455     parse_touch() {
01456
01457         std::string_view lower_tag;
01458
01459         switch(lower_tag[0]) {
01460
01461             case 't': {
01462                 this->get_str();
01463                 break;
01464             }
01465
01466         }
01467     }
01468
01469     void
01470     parse_vision() {
0
```

```

00307
00308         case 'i': {
00309             uint8_t value;
00310             this->get_value(value);
00311             break;
00312         }
00313
00314         // Após essas, qualquer informação dada será da parte do corpo dele.
00315         case 'h': {
00316             }
00317         case 'r': {
00318             }
00319         case 'l': {
00320             // Vamos apenas pular as informações
00321             this->advance(5);
00322             float value;
00323             for(int i = 0; i < 3; i++){ this->get_value(value); }
00324             break;
00325         }
00326
00327     }
00328
00329     default:
00330         env->logger.warn("[{}]\tFlag Desconhecida dentro de 'See:P': {}.\n\t"
00331         Buffer Neste momento: {}, env->unum, lower_tag, this->buffer);
00332         break;
00333     }
00334
00335     if(*this->buffer == ')'){ this->advance(1); if(*this->buffer == ')'){ 
00336         break; } }
00337         }
00338         break;
00339         case 'B': {
00340             }
00341             // Landmarks
00342             case 'G': {
00343                 }
00344             case 'F': {
00345                 this->advance(5);
00346                 float value;
00347                 for(int i = 0; i < 3; i++){ this->get_value(value); }
00348                 break;
00349             }
00350
00351             case 'L': {
00352                 this->advance(5);
00353                 // Precisamos pegar ambos pontos da linha
00354                 float value;
00355                 for(int i = 0; i < 3; i++){ this->get_value(value); }

00356                 this->advance(6);
00357                 for(int i = 0; i < 3; i++){ this->get_value(value); }

00358                 break;
00359             }
00360             default:
00361                 env->logger.warn("[{}]\tFlag Desconhecida dentro de 'See': {}.\n\t"
00362                 Buffer Neste momento: {}, env->unum, lower_tag, this->buffer);
00363                 break;
00364             }
00365         }
00366
00367         if(*this->buffer == ')'){ this->advance(1); if(*this->buffer == ')'{ break; } }
00368     }
00369     }
00370     if(*this->buffer == ')'){ this->advance(1); if(*this->buffer == ')'{ break; } }
00371     }
00372 }
00373 }
00374
00375 void
00376 parse_hingejoint(){
00377
00378     // Dado que será sempre o mesmo padrão. É possível:
00379     this->advance(3);
00380     std::string_view nome_da_junta = this->get_str();
00381     this->advance(5);
00382     float value;
00383     this->get_value(value);
00384 }
00385
00386 void
00387 parse_force_resistance(){
00388
00389     // Dado que será sempre o mesmo padrão, é possível:
00390     this->advance(3);
00391 }
```

```

00403     this->get_str();
00404
00405     this->advance(4);
00406     // Começamos a pegar o vetor
00407     float value;
00408     for(int i = 0; i < 3; i++){ this->get_value(value); }
00409
00410     this->advance(4);
00411     for(int i = 0; i < 3; i++){ this->get_value(value); }
00412 }
00413
00414 void
00415     parse_hear(){
00416     // sanha
00417 }
00418
00419 int
00420     update_from_server(
00421         std::string_view msg
00422     ) {
00423
00424     Parsing cursor(msg, this);
00425     std::string_view upper_tag;
00426     while(True){
00427
00428         if(
00429             !cursor.skip_until_char('(')
00430         ){ this->print_status(); return 0; }
00431
00432         upper_tag = cursor.get_str();
00433         switch(upper_tag[0]){
00434             case 't': {
00435                 cursor.parse_time();
00436                 break;
00437             }
00438             case 'G': {
00439                 if(upper_tag[1] == 'S'){
00440                     cursor.parse_gamestate();
00441                 }
00442                 else if(upper_tag[1] == 'Y'){
00443                     cursor.parse_gyroscope();
00444                 }
00445                 else{
00446                     this->logger.warn("[{}]\nTag Superior Desconhecida: [{}]", this->unum,
00447                         upper_tag);
00448                 }
00449                 break;
00450             }
00451             case 'A': {
00452                 if(upper_tag[1] == 'C'){ cursor.parse_accelerometer(); }
00453                 break;
00454             }
00455             case 'S': {
00456                 if(upper_tag[1] == 'e'){ cursor.parse_vision(); }
00457                 else{ this->logger.warn("[{}]\nTag Superior Desconhecida: [{}]\n\tBuffer neste
00458                 momento: [{}]", this->unum, upper_tag, cursor.get()); }
00459                 break;
00460             }
00461             case 'H': {
00462                 cursor.parse_hingejoint();
00463                 break;
00464             }
00465             case 'F': {
00466                 cursor.parse_force_resistance();
00467                 break;
00468             }
00469             default: {
00470                 if(!cursor.is_valid()){ return 2; }
00471                 this->logger.warn("[{}]\nTag Superior Desconhecida: [{}]\n\tBuffer neste momento:
00472                 [{}]", this->unum, upper_tag, cursor.get());
00473                 break;
00474             }
00475         }
00476     }
00477
00478     private:
00479     void
00480         print_status() const {

```

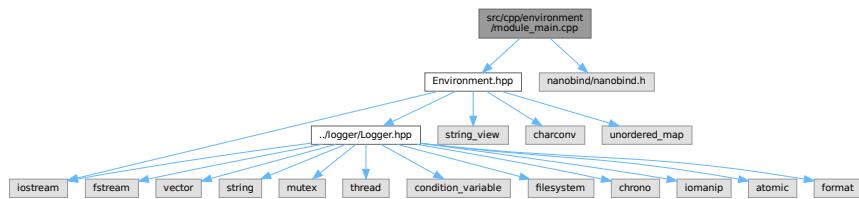
```

00500     return;
00501     printf("\n==== Environment State ====\n");
00502     printf("time_server : %.3f\n", time_server);
00503     printf("time_match : %.3f\n", time_match);
00504     printf("goals_scored : %d\n", goals_scored);
00505     printf("goals_conceded : %d\n", goals_conceded);
00506     printf("is_left : %d\n", is_left);
00507     printf("playmode : %d\n", static_cast<uint8_t>(current_mode));
00508 }
00509 };
00510
00511
00512
00513

```

7.17 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)

7.17.1 Function Documentation

7.17.1.1 NB_MODULE()

```
NB_MODULE (
    environment ,
    m )
```

Definition at line 6 of file [module_main.cpp](#).

7.18 module_main.cpp

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

```

00001 #include "Environment.hpp"
00002 #include <nanobind/nanobind.h>
00003
00004 namespace nb = nanobind;
00005
00006 NB_MODULE(
00007     environment,
00008     m

```

```

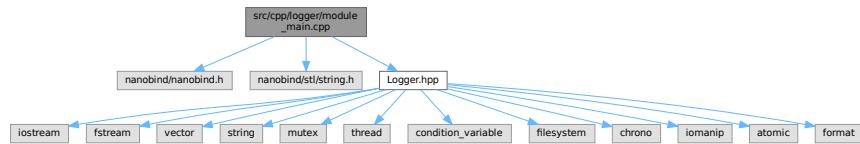
00009 ) {
00010     nb::class_<Environment>(
00011         m,
00012         "Environment",
00013         "Responsável por representar o ambiente externo ao robô.\n"
00014         "Agrupará todos os métodos de interpretação do mundo."
00015     )
00016     .def(
00017         nb::init<Logger&>(),
00018         nb::arg("Logger"),
00019         R"(Classe Logger para imprimirmos possíveis erros.)"
00020     )
00021
00022     /* -- Definição de Atributos da Classe -- */
00023     .def_ro("time_server", &Environment::time_server,
00024             "Instante de Tempo do Servidor, útil apenas para sincronização entre agentes")
00025
00026     .def_ro("time_match", &Environment::time_match,
00027             "Instante de Tempo de Partida")
00028
00029     .def_ro("goals_scored", &Environment::goals_scored,
00030             "Nossos Gols, pode ser útil para mudarmos de tática conforme o jogo avança")
00031
00032     .def_ro("goals_conceded", &Environment::goals_conceded,
00033             "Gols adversários, pode ser útil para mudarmos de tática conforme o jogo avança")
00034
00035     .def_ro("is_left", &Environment::is_left,
00036             "True caso sejamos o time da esquerda, False caso contrário.")
00037
00038     .def_ro("current_mode", &Environment::current_mode,
00039             "True caso sejamos o time da esquerda, False caso contrário.")
00040
00041
00042
00043     /* -- Métodos da Classe -- */
00044     .def(
00045         "update_from_server",
00046             // A função anônima é apenas para convertermos os tipos
00047         [](
00048             Environment &self,
00049             const nb::bytearray& from_server
00050         ){
00051             return self.update_from_server(std::string_view(reinterpret_cast<const
00052                 char*>(from_server.data()), from_server.size()));
00053         },
00054         nb::arg("from_server"),
00055         "Função responsável por atualizar o estado de ambiente a partir de mensagens do servidor."
00056     );
00057
00058     nb::enum_<Environment::PlayMode>(m, "PlayMode")
00059         .value("OUR_KICKOFF", Environment::PlayMode::OUR_KICKOFF)
00060         .value("OUR_KICK_IN", Environment::PlayMode::OUR_KICK_IN)
00061         .value("OUR_CORNER_KICK", Environment::PlayMode::OUR_CORNER_KICK)
00062         .value("OUR_GOAL_KICK", Environment::PlayMode::OUR_GOAL_KICK)
00063         .value("OUR_FREE_KICK", Environment::PlayMode::OUR_FREE_KICK)
00064         .value("OUR_PASS", Environment::PlayMode::OUR_PASS)
00065         .value("OUR_DIR_FREE_KICK", Environment::PlayMode::OUR_DIR_FREE_KICK)
00066         .value("OUR_GOAL", Environment::PlayMode::OUR_GOAL)
00067         .value("OUR_OFSIDE", Environment::PlayMode::OUR_OFSIDE)
00068
00069         .value("THEIR_KICKOFF", Environment::PlayMode::THEIR_KICKOFF)
00070         .value("THEIR_KICK_IN", Environment::PlayMode::THEIR_KICK_IN)
00071         .value("THEIR_CORNER_KICK", Environment::PlayMode::THEIR_CORNER_KICK)
00072         .value("THEIR_GOAL_KICK", Environment::PlayMode::THEIR_GOAL_KICK)
00073         .value("THEIR_FREE_KICK", Environment::PlayMode::THEIR_FREE_KICK)
00074         .value("THEIR_PASS", Environment::PlayMode::THEIR_PASS)
00075         .value("THEIR_DIR_FREE_KICK", Environment::PlayMode::THEIR_DIR_FREE_KICK)
00076         .value("THEIR_GOAL", Environment::PlayMode::THEIR_GOAL)
00077         .value("THEIR_OFSIDE", Environment::PlayMode::THEIR_OFSIDE)
00078
00079         .value("BEFORE_KICKOFF", Environment::PlayMode::BEFORE_KICKOFF)
00080         .value("GAME_OVER", Environment::PlayMode::GAME_OVER)
00081         .value("PLAY_ON", Environment::PlayMode::PLAY_ON)
00082     .export_values();
00083 }
00084

```

7.19 src/cpp/logger/module_main.cpp File Reference

```
#include <nanobind/nanobind.h>
#include <nanobind/stl/string.h>
```

```
#include "Logger.hpp"
Include dependency graph for module_main.cpp:
```



Functions

- `NB_MODULE(logger, m)`
< Necessário para converter std::string <-> str automaticamente

7.19.1 Function Documentation

7.19.1.1 NB_MODULE()

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

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

Definition at line 7 of file [module_main.cpp](#).

7.20 module_main.cpp

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

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

```

00031      "      msg (str): Mensagem a ser imprimida.\n\n"
00032      "Details:\n"
00033      "      Recebe por valor para permitir std::move (otimização de r-values).")
00034 .def("error", [] (Logger& self, std::string msg) {
00035                                     self.error(std::move(msg));
00036                                     },
00037                                     nb::arg("msg"),
00038                                     "Adiciona log nível ERROR.\n\n"
00039                                     "Args:\n"
00040                                     "      msg (str): Mensagem a ser imprimida.\n\n"
00041                                     "Details:\n"
00042                                     "      Recebe por valor para permitir std::move (otimização de r-values).");
00043 }
00044

```

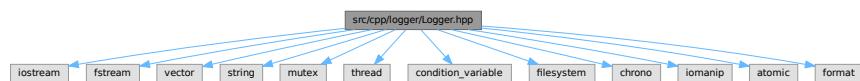
7.21 src/cpp/logger/Logger.hpp File Reference

```

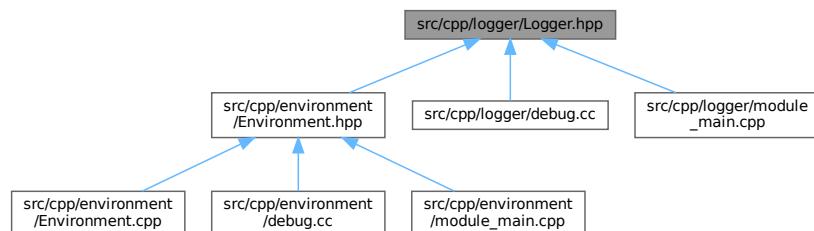
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <mutex>
#include <thread>
#include <condition_variable>
#include <filesystem>
#include <chrono>
#include <iomanip>
#include <atomic>
#include <format>

```

Include dependency graph for Logger.hpp:



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



Classes

- class [Logger](#)

Singleton para logging assíncrono.

Macros

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

7.21.1 Macro Definition Documentation

7.21.1.1 False

```
#define False false
```

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

7.21.1.2 True

```
#define True true
```

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

7.22 Logger.hpp

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

```
00001 #pragma once
00002
00003 #include <iostream>
00004 #include <fstream>
00005 #include <vector>
00006 #include <string>
00007 #include <mutex>
00008 #include <thread>
00009 #include <condition_variable>
00010 #include <filesystem>
00011 #include <chrono>
00012 #include <iomanip>
00013 #include <atomic>
00014 #include <format>
00015
00016 namespace fs = std::filesystem;
00017
00018 #define True true
00019 #define False false
00020
00021
00022 class Logger {
00023 public:
00024     static Logger& get(){ static Logger instance; return instance; }
00025
00026     Logger(const Logger&) = delete;
00027     void operator=(const Logger&) = delete;
00028
00029     void
00030         info(std::string msg){ this->log("[INFO] ", std::move(msg)); }
00031
00032     void
00033         warn(std::string msg){ this->log("[WARN] ", std::move(msg)); }
00034
00035     void
00036         error(std::string msg){ this->log("[ERROR] ", std::move(msg)); }
00037
00038     template<typename... Args>
00039     void info(std::format_string<Args...> fmt, Args&... args) {
00040         // std::format gera a std::string final de forma otimizada.
00041         // std::forward garante que não haja cópias desnecessárias dos argumentos.
00042         this->log("[INFO] ", std::format(fmt, std::forward<Args>(args)...));
00043     }
00044
00045     template<typename... Args>
```

```

00075     void warn(std::format_string<Args...> fmt, Args&&... args) {
00076         this->log("[WARN] ", std::format(fmt, std::forward<Args>(args)...));
00077     }
00078
00082     template<typename... Args>
00083     void error(std::format_string<Args...> fmt, Args&&... args) {
00084         this->log("[ERROR] ", std::format(fmt, std::forward<Args>(args)...));
00085     }
00086
00087 private:
00088     // Buffers para técnica de Double Buffering
00089     std::vector<std::string> _current_buffer;
00090     std::vector<std::string> _write_buffer;
00091
00092     std::mutex _mutex;
00093     std::condition_variable _cv;
00094     std::thread _worker;
00095     std::atomic<bool> _is_running;
00096     std::atomic<bool> is_the_first = True;
00097     std::ofstream _file_stream;
00098
00099     Logger() : _is_running(True) {
00100         // Reserva memória prévia para evitar realocações frequentes no vetor
00101         this->_current_buffer.reserve(30);
00102         this->_write_buffer.reserve(30);
00103     }
00104
00112     ~Logger() {
00113         this->_is_running = false;
00114         this->_cv.notify_one();
00115
00116         if(this->_worker.joinable()) { this->_worker.join(); }
00117         if(this->_file_stream.is_open()) { this->_file_stream.close(); }
00118     }
00119
00124     void
00125     _init_file() {
00126         if(!fs::exists("logs")){ fs::create_directory("logs"); }
00127
00128         auto now = std::chrono::system_clock::now();
00129         auto in_time_t = std::chrono::system_clock::to_time_t(now);
00130
00131         std::stringstream ss;
00132         ss << "logs/" << std::put_time(std::localtime(&in_time_t), "%Y-%m-%d_%H-%M-%S") << ".log";
00133
00134         // std::ios::app não é necessário se o arquivo é único por execução
00135         // mas útil se reiniciarmos o logger no mesmo segundo -> Impossível?
00136         this->_file_stream.open(ss.str(), std::ios::out | std::ios::app);
00137
00138         // Desabilita sincronização automática com stdio para performance
00139         std::ios_base::sync_with_stdio(false);
00140     }
00141
00148     void
00149     log(const char* prefixo, std::string&& msg) {
00150
00151         // --- INÍCIO DA ADIÇÃO DO TIMESTAMP ---
00152         auto now = std::chrono::system_clock::now();
00153         auto in_time_t = std::chrono::system_clock::to_time_t(now);
00154
00155         std::stringstream ss_time;
00156         // Formato: [YYYY-MM-DD HH:MM:SS]
00157         ss_time << std::put_time(std::localtime(&in_time_t), "[%Y-%m-%d %H:%M:%S] ");
00158         // --- FIM DA ADIÇÃO DO TIMESTAMP ---
00159
00161         {
00162             std::lock_guard<std::mutex> lock(this->_mutex);
00163             // Constrói a string final na memória RAM
00164             this->_current_buffer.emplace_back(ss_time.str() + prefixo + msg);
00165
00166             if( this->is_the_first ){ this->_init_file();
00167                 this->_worker = std::thread(&Logger::_worker_loop, this);
00168                 this->is_the_first = False;
00169             }
00170         }
00171
00172         // Notifica a thread de escrita que há dados
00173         _cv.notify_one();
00174     }
00175
00180     void
00181     _worker_loop() {
00182
00183         while(
00184             _is_running || !_current_buffer.empty()
00185         ){
00186

```

```

00187         std::unique_lock<std::mutex> lock(_mutex);
00188
00189         /*
00190         A thread fica bloqueada pelo sistema operacional, sem consumir CPU.
00191         Pesquise, isso é muito foda.
00192         */
00193         _cv.wait(
00194             lock,
00195             [this](){ return !this->_current_buffer.empty() || !this->_is_running; }
00196         );
00197
00198         if( this->_current_buffer.empty() && !this->_is_running ){ break; }
00199
00200         // --- A MÁGICA DA PERFORMANCE (SWAP) ---
00201         // Trocamos o vetor cheio pelo vazio instantaneamente.
00202         // O Mutex é liberado logo depois disso.
00203         std::swap(this->_current_buffer, this->_write_buffer);
00204         lock.unlock();
00205
00206
00207         if(this->_file_stream.is_open()) {
00208             for(const auto& line : this->_write_buffer){ this->_file_stream << line << "\n"; }
00209             // Flush manual apenas após lote grande
00210             this->_file_stream.flush();
00211         }
00212
00213         // Limpa o buffer de escrita para ser reusado no próximo swap
00214         this->_write_buffer.clear();
00215     }
00216 }
00217 }
00218 };
00219

```

7.23 src/exec_booting.py File Reference

Namespaces

- namespace [exec_booting](#)

7.24 exec_booting.py

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

```

00001 from term.Booting import Booting
00002
00003 Booting()

```

7.25 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`

7.26 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.beam()
00014     p.scom.send()
00015
00016 for p in players:
00017     p.scom.receive()
00018
00019
00020 print("Em condições.")
00021 while True:
00022     for p in players:
00023         p.scom.send()
00024
00025     for p in players:
00026         p.scom.receive()
```

7.27 src/run_player.py File Reference

Namespaces

- namespace [run_player](#)

Variables

- [run_player.boot](#) = Booting()
- [run_player.p](#) = Agent(boot.options)

7.28 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 p = Agent(boot.options)
00007
00008 while True:
00009     p.scom.send()
00010     p.scom.receive()
```

7.29 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](#)

7.29.1 Detailed Description

Implementação do [Booting](#) do time.

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

7.30 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             for idx in range(0, len(config_team_params)):
00062                 # Somente o IP Server e Team Name são palavras
00063                 if idx not in {0, 3}:
00064                     config_team_params[idx][1] = int(config_team_params[idx][1])
00065
00066
00067             config_team_params = [
00068                 ["IP Server",      "localhost"],
00069                 ["Agent Port",     3100], # Onde nos conectaremos com rcssserver3d
00070                 ["Monitor Port",   3200], # Onde nos conectaremos com Roboviz
00071                 ["Team Name",       "RoboIME"],
00072                 ["Uniform Number", 1],
00073                 ["Robot Type",     1],
00074                 ["Penalty Shootout", 0],
00075                 ["MagmaFatProxy",  0],
00076                 ["Debug Mode",      1]
00077             ]
00078         ]
00079
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
00109     @staticmethod
00110     def cpp_builder() -> None:
00111         """
00112             @brief Responsável por buildar os arquivos .cpp presentes na pasta cpp.
00113             @return Funcionalidades C++ em condições de interoperabilidade.
00114         """
00115
00116         # Vamos verificar quais arquivos .cpp estão disponíveis para buildar
00117         cpp_path = Path(__file__).resolve().parents[1] / "cpp"
00118         cpp_modules = [
00119             module for module in os.listdir(
00120                 cpp_path
00121             ) if os.path.isdir(os.path.join(cpp_path, module))
00122         ]
00123
00124         if not cpp_modules:
00125             return None # Não há nenhum para construirmos
00126
00127         # Servirá para verificarmos quais binários estão atualizados com a versão
00128         python_cmd = f"python{sys.version_info.major}.{sys.version_info.minor}"
00129
00130         # -- Os includes que serão necessários
00131         nb_root = os.path.dirname(nanobind.__file__)
00132         py_inc = sysconfig.get_path("include") # Python.h
00133         nb_inc = nanobind.include_dir() # nanobind.h
00134         robin_inc = os.path.join(nb_root, "ext", "robin_map", "include") # robin_map.h
00135         nb_src = os.path.join(nb_root, "src", "nb_combined.cpp")
00136         n_proc = str(os.cpu_count())
00137         command_chain = [
00138             "make",

```

```

00139         f"-j{n_proc}",
00140         f"PY_INC={py_inc}",
00141         f"NB_INC={nb_inc}",
00142         f"ROBIN_INC={robin_inc}",
00143         f"NB_SRC={nb_src}"
00144     ]
00145
00146     first = True
00147     for cpp_module in cpp_modules:
00148         cpp_module_path = os.path.join(cpp_path, cpp_module)
00149
00150         # Verificamos se já existe um binário pronto
00151         if os.path.isfile(
00152             os.path.join(
00153                 cpp_module_path,
00154                 f"{cpp_module}.so"
00155             )
00156         ):
00157             # Caso exista, devemos verificar se ele foi modificado em um limite de tempo
00158             with open(
00159                 os.path.join(
00160                     cpp_module_path,
00161                     f"{cpp_module}.cpp_info"
00162                 ),
00163                 "rb"
00164             ) as f:
00165                 info_version = pickle.load(f)
00166
00167             if info_version == python_cmd:
00168                 # Considerando que está na mesma versão, ainda devemos verificar modificações
00169
00170                 code_mod_time = max(
00171                     os.path.getmtime(
00172                         os.path.join(
00173                             cpp_module_path,
00174                             file_in_the_module
00175                         )
00176                     ) for file_in_the_module in os.listdir(
00177                         cpp_module_path
00178                     ) if file_in_the_module.endswith(".cpp") or
00179                         file_in_the_module.endswith(".hpp")
00180
00181                 bin_mod_time = os.path.getmtime(os.path.join(cpp_module_path, f"{cpp_module}.so"))
00182
00183                 if bin_mod_time + 15 > code_mod_time:
00184                     continue
00185
00186             if first:
00187                 print("\033[1;7m/* ---- Construção de Funcionalidades C++ ---- */\033[0m")
00188                 first = False
00189             msg = f"\033[1;7mConstruindo: \033[32;40m{cpp_module}\033[0m"
00190             print(f"{}{msg}{}.{<60}{", end=" ", flush=True)
00191
00192             processo = subprocess.Popen(
00193                 command_chain,
00194                 cwd=cpp_module_path,
00195                 stdout=subprocess.PIPE,
00196                 stderr=subprocess.PIPE,
00197                 text=False
00198             )
00199
00200             # Iniciamos thread de spinner
00201             running_flag = [True]
00202             worker = threading.Thread(target=Booting.show_spinner, args=(running_flag,))
00203             worker.start()
00204
00205             output, error = processo.communicate()
00206             return_code = processo.wait()
00207
00208             running_flag[0] = False
00209             worker.join()
00210
00211             if return_code == 0:
00212                 print("\033[7m\033[1mSucesso\033[0m")
00213
00214                 # Podemos construir um arquivo de fiscalização
00215                 with open(
00216                     os.path.join(cpp_module_path, f"{cpp_module}.cpp_info"),
00217                     "wb"
00218                 ) as f:
00219                     # noinspection PyTypeChecker
00220                     pickle.dump(python_cmd, f)
00221             else:
00222                 Printing.print_message("Abortando", "error")
00223                 print()
00224                 print(output.decode(), error.decode())

```

```

00225         exit()
00226
00227     subprocess.run(
00228         ["make", "clean"],
00229         stdout=subprocess.PIPE,
00230         stderr=subprocess.PIPE,
00231         cwd=cpp_module_path
00232     )
00233
00234     return None

```

7.31 src/term/Printing.py File Reference

Implementação de Interface no terminal.

Classes

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

Namespaces

- namespace [Printing](#)

7.31.1 Detailed Description

Implementação de Interface no terminal.

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

7.32 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": "\u001b[1;36m",
00020         "warning": "\u001b[1;33m",
00021         "error": "\u001b[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 Tome cuidado com a execução dessa função, pois ela é poderosa
00123             @param return_type Tipo de entrada a ser retornado
00124             @param bytes_to_be_read Quantidade de Bytes que serão lidos
00125             @return Entrada do usuário
00126         """
00127
00128         # Obtém o File Descriptor do stdin
00129         fd = sys.stdin.fileno()
00130
00131         # Guarda modo original (echo, buffering, etc) para restaurar depois
00132         old_settings = termios.tcgetattr(fd)
00133
00134         buffer = ""
00135
00136         try:
00137             # - Desativa buffering de linha (não espera Enter)
00138             # - Desativa echo (não mostra teclas na tela)
00139             # - Desativa processamento de caracteres especiais (Ctrl+C, etc)
00140             # - Captura teclas imediatamente
00141             tty.setraw(fd)
00142
00143             while len(buffer) < bytes_to_be_read:
00144                 # Verifica se há input disponível (não-bloqueante)
00145                 if select([sys.stdin], [], [], 0.5)[0]:
00146                     # Adicionamos cada caractere
00147                     buffer += sys.stdin.read(1)
00148                     if buffer[-1] in {'\r', '\n'}:
00149                         break
00150
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

```

7.33 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\(\)](#)

7.33.1 Detailed Description

Implementação de lógica organizadora de posições iniciais de partida.

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

7.34 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         # Disporemos 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         # Disporemos 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)
00149
00150         frame_botoes = ttk.Frame(upper_frame)
00151         frame_botoes.pack(side="right", fill="y", padx=10)
00152
00153         ttk.Button(frame_botoes, text="Nova Configuração", command=self.nova_config).pack(fill="x",
00154             pady=2)
00155         ttk.Button(frame_botoes, text="Salvar Atual", command=self.salvar_config).pack(fill="x",
00156             pady=2)

```

```

00153         ttk.Button(frame_botoes, text="Apagar Selecionada", command=self.apagar_config).pack(fill="x",
00154             pady=2)
00154         ttk.Button(frame_botoes, text="Limpar Grade", command=lambda: (self.clear_grid(),
00154             self.posicoes_atuais.clear())).pack(fill="x", pady=10)
00155
00156     # ----- Focando no campo
00157     frame_grid = ttk.Frame(self)
00158     frame_grid.pack(side="top", fill="both", expand=True, padx=10, pady=10)
00159
00160     # Canvas para o campo
00161     self.canvas = tk.Canvas(
00162         frame_grid,
00163         width=self.canvas_width,
00164         height=self.canvas_height,
00165         bg="#42f545" # Verde para o campo
00166     )
00167     self.canvas.pack()
00168
00169     # Bind do clique no canvas
00170     self.canvas.bind("<Button-1>", self.click_on_gridclick_on_grid)
00171
00172     self.clear_grid()
00173
00174 def draw_player(self, field_x, field_y) -> None:
00175     """
00176     @brief Desenharemos um jogador na posição especificada
00177     @param field_x Posição real em X
00178     @param field_y Posição real em Y
00179     """
00180
00181     # Converte as coordenadas do campo (ex: -14, 0) para pixels
00182     cx, cy = self._field_to_canvas(field_x, field_y)
00183
00184     r = self.GRID_SCALE / 3
00185
00186     oval_id = self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r,
00187                                         fill="yellow", outline="black", width=2)
00188
00189     self.marcadores_jogadores.append((oval_id, (field_x, field_y)))
00190
00191 # -- Métodos de Interação
00192 def click_on_grid(self, event: tk.Event):
00193     """
00194     @brief Responsável por identificar onde o usuário clicou e adicionar essa posição na lista
00195     @param event Argumento default do bind
00196     """
00197
00198     new_pos = self._canvas_to_field(event.x, event.y)
00199
00200     # Verificamos se clicamos em cima de um jogador
00201     for i, (oval_id, pos) in enumerate(self.marcadores_jogadores):
00202         if pos == new_pos:
00203             self.canvas.delete(oval_id)
00204             self.marcadores_jogadores.pop(i)
00205             self.posicoes_atuais.remove(new_pos)
00206             return
00207
00208     # Verificamos se o limite de jogadores foi atingido
00209     if len(self.posicoes_atuais) >= self.MAX_JOGADORES:
00210         messagebox.showwarning("Limite Atingido",
00211                               f"Não é possível adicionar mais de {self.MAX_JOGADORES}
jogadores.\n"
00212                               "Clique em um jogador existente para removê-lo.")
00213
00214
00215     # Caso nenhuma das opções anteriores, adicionamos
00216     self.posicoes_atuais.append(new_pos)
00217     self.draw_player(*new_pos)
00218
00219 def on_double_click_in_configs(self, event: tk.Event) -> None:
00220     """
00221     @brief Responsável por plotar a configuração de jogadores selecionada
00222     @param event Argumento Default de bind
00223     """
00224
00225     item_selecionado = self.tv_configs.focus()
00226     if not item_selecionado:
00227         return
00228
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
00257             {nome_config}?"
00258         ):
00259             # Atualizaremos
00260             self.config_positions[nome_config] = self.posicoes_atuais.copy()
00261             self.update_table_config()
00262             for item in self.tv_configs.get_children():
00263                 if self.tv_configs.item(item, 'values')[0] == nome_config: # [0] = primeira coluna
00264                     self.tv_configs.selection_set(item)
00265                     self.nome_de_config_selecionada = nome_config
00266                     break
00267
00268     def clear_grid(self) -> None:
00269         """
00270         @brief Responsável por limpar as posições e a grade
00271         """
00272
00273         self.canvas.delete("all")
00274         self.marcadores_jogadores = []
00275
00276         # Círculo central (usando a conversão de coordenadas)
00277         cx, cy = self._field_to_canvas(0,0)
00278         r = self.GRID_SCALE * 4 # Raio de 4 unidades
00279         self.canvas.create_oval(cx - r, cy - r, cx + r, cy + r, outline="white", width=2)
00280
00281         # --- Desenhar Linhas da Grade (Quadrados) ---
00282
00283         # Total de passos de 0.5
00284         n_steps_x = int(self.FIELD_WIDTH * 2) + 1
00285         n_steps_y = int(self.FIELD_HEIGHT * 2) + 1
00286
00287         # Linhas Verticais (eixo X)
00288         for i in range(n_steps_x):
00289             fx = self.X_MIN + (i * 0.5)
00290
00291             # --- Lógica das Cores (Req. 3) ---
00292             cor = "white" if fx == 0 else "#337033"
00293             largura = 2 if fx == 0 else 1
00294
00295             # Converte a coordenada X para pixel
00296             cx, _ = self._field_to_canvas(fx, 0)
00297
00298             # Desenha a linha (Req. 2 - todas as linhas são desenhadas)
00299             self.canvas.create_line(cx, 0, cx, self.canvas_height,
00300                                   fill=cor, width=largura)
00301
00302         # Linhas Horizontais (eixo Y)
00303         for i in range(n_steps_y):
00304             fy = self.Y_MIN + (i * 0.5)
00305
00306             # --- Lógica das Cores (Req. 3) ---
00307             cor = "white" if fy == 0 else "#337033"
00308             largura = 2 if fy == 0 else 1
00309
00310             # Converte a coordenada Y para pixel
00311             _, cy = self._field_to_canvas(0, fy)
00312
00313             # Desenha a linha (Req. 2)
00314             self.canvas.create_line(0, cy, self.canvas_width, cy,
00315                                   fill=cor, width=largura)
00316
00317             # Caixas do Gol Esquerda (-15 a -13 em X, 3 a -3 em Y)
00318             coords_gol_esq = (-15, 3, -13, -3)
00319
00320             # Caixas do Gol Direita (13 a 15 em X, 3 a -3 em Y)
00321             coords_gol_dir = (13, 3, 15, -3)
00322
00323             # 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
```

7.35 src/utils/RobotVision.py File Reference

Classes

- class RobotVision.Elemento
- class RobotVision.Ball
- class RobotVision.Marker
- class RobotVision.Goal
- class RobotVision.Line
- class RobotVision.RobotVision

Classe responsável por gerir a aplicação principal.

Namespaces

- namespace RobotVision

Implementação de Classe que nos permitirá ter a visão do robô

Variables

- RobotVision.WIDTH
- RobotVision.HEIGHT

7.36 RobotVision.py

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

```
00001 """
00002 @brief Implementação de Classe que nos permitirá ter a visão do robô
00003 """
00004 import pygame
00005 from time import perf_counter
00006 from math import sin, cos, radians, tan
00007
00008 WIDTH, HEIGHT = 1200, 1000
00009
00010 class Elemento:
00011
00012     def __init__(self):
00013         self.color = None
00014         self.width, self.height = WIDTH, HEIGHT # Deve-se alterar no tamanho da imagem também
00015         self.fov_h, self.fov_v = radians(120), radians(120)
00016
00017     def projection_to_2d(self):
00018         pass
00019
00020     def draw(self, window: pygame.Surface, font=None):
00021         pass
00022
00023     def project_point(self, triple_numbers: list[float], forward_axis="+x", flip_x=True, flip_y=False)
00024         -> list[float]:
00025             ah = radians(triple_numbers[1])
00026             av = radians(triple_numbers[2])
00027
00028             x = triple_numbers[0] * cos(av) * cos(ah)
00029             y = triple_numbers[0] * cos(av) * sin(ah)
00030             z = triple_numbers[0] * sin(av)
00031
00032             if forward_axis == '+x':
00033                 depth = x
00034                 cx = y
00035                 cy = z
00036             elif forward_axis == '-x':
00037                 depth = -x
00038                 cx = -y
00039                 cy = z
00040             elif forward_axis == '+z':
```

```

00040         depth = z
00041         cx = x
00042         cy = y
00043     elif forward_axis == '-z':
00044         depth = -z
00045         cx = x
00046         cy = -y
00047     else:
00048         # default
00049         depth = x
00050         cx = y
00051         cy = z
00052
00053     # Vamos definir x como profundidade?
00054
00055     fx = (self.width / 2) / tan(self.fov_h / 2)
00056     fy = (self.height / 2) / tan(self.fov_v / 2)
00057
00058     x_ndc = (cx / depth) * (-1 if flip_x else 1) # NDC horizontal (negativo corrige espelho)
00059     y_ndc = (cy / depth) * (-1 if flip_y else 1) # NDC vertical
00060
00061     u = fx * x_ndc + self.width / 2
00062     v = - fy * y_ndc + self.height / 2
00063
00064     scale = 0.5 * fx / depth
00065
00066     return [u, v, scale]
00067
00068 class Ball(Elemento):
00069
00070     def __init__(self, position: list[float]) -> None:
00071         """
00072             @brief Construtor responsável por inicializar bola no e prover
00073             """
00074
00075         super().__init__()
00076
00077         self.colorcolor = (0, 255, 0)
00078         self.position_on_sphere = position
00079         self.position_on_window = self.projection_to_2dprojection_to_2d()
00080
00081     def projection_to_2d(self) -> list[float]:
00082         # Será apenas a projeção de um ponto
00083
00084         return self.project_point(self.position_on_sphere)
00085
00086     def draw(self, window, font=None) -> None:
00087         pygame.draw.circle(
00088             window,
00089             self.colorcolor,
00090             self.position_on_window[:-1],
00091             10
00092         )
00093
00094 class Marker(Elemento):
00095     def __init__(self, position: list[float]) -> None:
00096         """
00097             @brief Construtor responsável por inicializar bola no e prover
00098             """
00099
00100         super().__init__()
00101
00102         self.colorcolor = (255, 0, 0)
00103         self.position_on_sphere = position
00104         self.position_on_window = self.projection_to_2dprojection_to_2d()
00105
00106     def projection_to_2d(self) -> list[float]:
00107         # Será apenas a projeção de um ponto
00108
00109         return self.project_point(self.position_on_sphere)
00110
00111     def draw(self, window, font=None) -> None:
00112
00113         pygame.draw.circle(
00114             window,
00115             self.colorcolor,
00116             self.position_on_window[:-1],
00117             self.position_on_window[-1]
00118         )
00119
00120 class Goal(Elemento):
00121     def __init__(self, position: list[float]) -> None:
00122         """
00123             @brief Construtor responsável por inicializar bola no e prover
00124             """
00125
00126         super().__init__()

```

```

00127
00128     self.colorcolor = (0, 0, 255)
00129     self.position_on_sphere = position
00130     self.position_on_window = self.projection_to_2dprojection_to_2d()
00131
00132     def projection_to_2d(self) -> list[float]:
00133         # Será apenas a projeção de um ponto
00134
00135         return self.project_point(self.position_on_sphere)
00136
00137     def draw(self, window, font=None) -> None:
00138
00139         pygame.draw.circle(
00140             window,
00141             self.colorcolor,
00142             self.position_on_window[:-1],
00143             self.position_on_window[-1]
00144         )
00145
00146 class Line(Elemento):
00147
00148     def __init__(self, double_list_position):
00149
00150         super().__init__()
00151         self.colorcolor = (255, 255, 255)
00152         self.position_on_sphere = double_list_position
00153         self.position_on_window = self.projection_to_2dprojection_to_2d()
00154
00155     def projection_to_2d(self) -> list[float]:
00156
00157         return self.project_point(self.position_on_sphere[:3]) +
00158         self.project_point(self.position_on_sphere[3:])
00159
00160     def draw(self, window, font=None) -> None:
00161
00162         pygame.draw.line(
00163             window,
00164             self.colorcolor,
00165             self.position_on_window[:2],
00166             self.position_on_window[3:5],
00167             2
00168         )
00169
00170 class RobotVision:
00171     """
00172         @brief Classe responsável por gerir a aplicação principal.
00173     """
00174
00175     FRAMES_VISION_PATH = "frames_vision.txt"
00176
00177     def __init__(self):
00178         self.frames = None
00179         self.current_index = 1
00180         self.need_to_update = True
00181         self.objects = []
00182
00183     def load_frames_from_file(self) -> None:
00184         """
00185             @brief Abrirá um arquivo que conterá os frames recebidos pelo agente.
00186             @details
00187             Pode ser aprimorada para permitir observação em tempo real.
00188         """
00189         with open(RobotVision.FRAMES_VISION_PATH, "r") as f:
00190             self.frames = [line for line in f]
00191
00192     def _get_only_tag_See(self) -> str | None:
00193         """
00194             @brief Buscará no frame principal o bloco referente ao conjunto See.
00195             @return String referente ao bloco See. None caso não exista.
00196         """
00197
00198         # 1. Definir o marcador de início
00199         start_marker = "(See"
00200         start_index = self.frames[self.current_index].find(start_marker)
00201
00202         # Se não houver informação visual (ex: robô cego ou mensagem de status puro)
00203         if start_index == -1:
00204             return None
00205
00206         # 2. Lógica de counter_entry de parênteses
00207         balance = 0
00208         end_index = -1
00209
00210         # Iteramos a partir do início do bloco See
00211         for i in range(start_index, len(self.frames[self.current_index])):
00212             char = self.frames[self.current_index][i]

```

```

00213
00214     if char == '(':
00215         balance += 1
00216     elif char == ')':
00217         balance -= 1
00218
00219     # 3. Verificação de saída
00220     # Se o balance voltou a zero, fechamos o bloco do (See ...)
00221     if balance == 0:
00222         end_index = i
00223         break
00224
00225     # 4. Retornar a substring exata
00226     # Adicionamos +1 no end_index pois o slice em Python é exclusivo no final
00227     if end_index != -1:
00228         return self.frames[self.current_index][start_index: end_index + 1]
00229
00230     return None
00231
00232 def parse_frame(self) -> None:
00233     """
00234     @brief Interpreta uma mensagem 'See' do Simspark/Rcssserver3d.
00235     @details
00236     Divide a responsabilidade com subfunções.
00237     """
00238
00239     if not self.need_to_update:
00240         return None
00241
00242     self.need_to_update = False
00243     inicio_de_interpretacao = perf_counter()
00244
00245     # -----
00246
00247     self.objects.clear()
00248     chunk_see = self._get_only_tag_See()
00249
00250     # Vamos iterar sobre essa string a fim de construirmos os elementos visuais do nosso frame.
00251
00252     # Remove o "(See " inicial e o ")" final para iterar no conteúdo
00253     chunk_see = chunk_see[5:-1]
00254
00255     counter_entry = 0
00256     buffer_objeto = ""
00257     for char in chunk_see:
00258         # 1. Controle de aninhamento
00259         if char == '(':
00260             if counter_entry == 0:
00261                 buffer_objeto = "" # Limpa buffer para novo objeto pai
00262             counter_entry += 1
00263
00264         # 2. Acumula caracteres se estivermos dentro de um objeto
00265         if counter_entry > 0:
00266             buffer_objeto += char
00267
00268         # 3. Fechamento de nível
00269         if char == ')':
00270             counter_entry -= 1
00271
00272         # Se counter_entry zerou, temos um objeto completo (ex: "(B (pol ...))")
00273         if counter_entry == 0:
00274
00275             # -- Aqui manteremos nossa lógica separada para cada flag
00276
00277             match buffer_objeto[1]:
00278                 case 'B':
00279                     # Receberemos apenas a posição dela
00280                     self.objects.append(
00281                         Ball(
00282                             list(
00283                                 map(
00284                                     float,
00285                                     buffer_objeto.split("pol")[1].replace(")", "").split()
00286
00287                             )
00288                         )
00289                     )
00290
00291                 case 'L':
00292                     buffer_objeto = buffer_objeto.replace("(", "").replace(")", "").split()
00293                     buffer_objeto.remove('L')
00294                     buffer_objeto.remove('pol')
00295                     buffer_objeto.remove('pol')
00296                     self.objects.append(
00297                         Line(
00298                             list(
00299                                 map(

```

```

00300                         float,
00301                         buffer_objeto
00302                     )
00303                 )
00304             )
00305         )
00306     )
00307     case 'F':
00308         self.objects.append(
00309             Marker(
00310                 list(
00311                     map(
00312                         float,
00313                         buffer_objeto.split("pol")[1].replace("", "").split()
00314                     )
00315                 )
00316             )
00317         )
00318     )
00319     case 'G':
00320         self.objects.append(
00321             Goal(
00322                 list(
00323                     map(
00324                         float,
00325                         buffer_objeto.split("pol")[1].replace("", "").split()
00326                     )
00327                 )
00328             )
00329         )
00330     )
00331     case _:
00332         pass
00333
00334 # -----
00335
00336 final_de_interpretacao = perf_counter()
00337 print(f"Tempo Total de Interpretação: {final_de_interpretacao - inicio_de_interpretacao}")
00338 return None
00339
00340 @staticmethod
00341 def draw_legend(screen, items, font, padding=10, line_height=20):
00342     """
00343     @brief Desenhá a legenda das cores
00344     """
00345     x = padding
00346     y = screen.get_height() - padding - line_height * len(items)
00347
00348     for name, color in items:
00349         # desenha quadradinho da cor
00350         pygame.draw.rect(screen, color, (x, y + 4, 12, 12))
00351
00352         # escreve texto
00353         text = font.render(name, True, (255, 255, 255))
00354         screen.blit(text, (x + 20, y))
00355
00356         y += line_height
00357
00358 def mainloop(self) -> None:
00359
00360     self.load_frames_from_file()
00361
00362     pygame.init()
00363     screen = pygame.display.set_mode((WIDTH, HEIGHT))
00364     pygame.display.set_caption("RobotVision")
00365
00366     font = pygame.font.SysFont(None, 25)
00367     clock = pygame.time.Clock()
00368
00369     legenda_dos_elementos = [
00370         ("Linha de Campo", (255, 255, 255)),
00371         ("Bola", (0, 255, 0)),
00372         ("Bandeira de Canto", (255, 0, 0)),
00373         ("Trave de Gol", (0, 0, 255))
00374     ]
00375
00376     running = True
00377     holding = 0
00378     move_delay = 150
00379     last_move_time = 0
00380
00381     while running:
00382         current_time = pygame.time.get_ticks()
00383         for event in pygame.event.get():
00384             if event.type == pygame.QUIT:
00385                 running = False
00386
00387             if event.type == pygame.KEYDOWN:

```

```
00387     if event.key in (pygame.K_RIGHT, pygame.K_d):
00388         holding = 1
00389         last_move_time = 0
00390
00391     if event.key in (pygame.K_LEFT, pygame.K_a):
00392         holding = -1
00393         last_move_time = 0
00394
00395
00396     if event.type == pygame.KEYUP:
00397         if event.key in (pygame.K_RIGHT, pygame.K_d, pygame.K_LEFT, pygame.K_a):
00398             holding = 0
00399
00400     if holding != 0:
00401         if current_time - last_move_time > move_delay:
00402             # Atualiza o índice
00403             self.current_index = (self.current_index + holding) % len(self.frames)
00404             self.need_to_update = True
00405
00406             # Reseta o cronômetro para o tempo atual
00407             last_move_time = current_time
00408
00409             screen.fill((0, 0, 0))
00410
00411             # --- Implementação de Lógicas
00412
00413             self.parse_frame()
00414
00415             if self.frames:
00416                 label = f"Frame {self.current_index + 1} / {len(self.frames)}"
00417                 text = font.render(label, True, (255, 255, 255))
00418                 screen.blit(text, (WIDTH - text.get_width() - 10, 10))
00419
00420
00421             for obj in self.objects:
00422                 obj: Elemento
00423                 obj.draw(screen)
00424
00425             self.draw_legend(screen, legenda_dos_elementos, font)
00426
00427
00428             pygame.display.flip()
00429             clock.tick(30)
00430
00431             pygame.quit()
00432
00433
00434
00435
00436
00437
00438
00439
00440 if __name__ == '__main__':
00441     RobotVision().mainloop()
```


Index

`__init__`
 Agent.Agent, 14
 BaseAgent.BaseAgent, 22
 Booting.Booting, 24
 RobotPositionManager.RobotPositionManager, 71
 RobotVision.Ball, 18
 RobotVision.Elemento, 27
 RobotVision.Goal, 39
 RobotVision.Line, 43
 RobotVision.Marker, 55
 RobotVision.RobotVision, 79
 ServerComm.ServerComm, 82

`_receive_async`
 ServerComm.ServerComm, 83

`_canvas_to_field`
 RobotPositionManager.RobotPositionManager, 71

`_current_buffer`
 Logger, 51

`_cv`
 Logger, 51

`_field_to_canvas`
 RobotPositionManager.RobotPositionManager, 71

`_file_stream`
 Logger, 51

`_get_only_tag_See`
 RobotVision.RobotVision, 79

`_init_file`
 Logger, 47

`_is_running`
 Logger, 51

`_mutex`
 Logger, 51

`_worker`
 Logger, 51

`_worker_loop`
 Logger, 47

`_write_buffer`
 Logger, 51

`~Logger`
 Logger, 47

`ACTIVE_BEAM`
 Environment, 34

`advance`
 Environment::Parsing, 59

`Agent`, 9

`Agent.Agent`, 13
 `__init__`, 14
 `unum`, 15

`AgentPenalty`, 9

`AGENTS_IN_THE_MATCH`
 BaseAgent.BaseAgent, 22

`apagar_config`
 RobotPositionManager.RobotPositionManager, 72

`BaseAgent`, 9

`BaseAgent.BaseAgent`, 20
 `__init__`, 22
 `AGENTS_IN_THE_MATCH`, 22
 beam, 22
 env, 22
 `init_position`, 22
 `INITIAL_POSITION`, 23
 logger, 23
 scom, 23
 unum, 23

`beam`
 BaseAgent.BaseAgent, 22

`BEFORE_KICKOFF`
 Environment, 33

`boot`
 `run_full_team`, 11
 `run_player`, 12

`Booting`, 9

`Booting.Booting`, 23
 `__init__`, 24
 `CONFIG_PATH`, 25
 `cpp_builder`, 25
 `get_team_params`, 25
 `options`, 25
 `show_spinner`, 25

`buffer`
 Environment::Parsing, 63
 ServerComm.ServerComm, 85

`buffer_size`
 ServerComm.ServerComm, 85

`canvas`
 RobotPositionManager.RobotPositionManager, 75

`canvas_height`
 RobotPositionManager.RobotPositionManager, 75

`canvas_width`
 RobotPositionManager.RobotPositionManager, 75

`clear_grid`
 RobotPositionManager.RobotPositionManager, 72

`clear_queue`
 ServerComm.ServerComm, 83

`click_on_grid`
 RobotPositionManager.RobotPositionManager, 72, 75

close
 ServerComm.ServerComm, 83

color
 RobotVision.Ball, 19
 RobotVision.Elemento, 28
 RobotVision.Goal, 40
 RobotVision.Line, 44
 RobotVision.Marker, 56

commit
 ServerComm.ServerComm, 83

commit_beam
 ServerComm.ServerComm, 84

CONFIG_PATH
 Booting.Booting, 25

CONFIG_POSITION_PATH
 RobotPositionManager.RobotPositionManager, 75

config_positions
 RobotPositionManager.RobotPositionManager, 75

CONSOLE
 Printing.Printing, 66

cpp_builder
 Booting.Booting, 25

criar_widgets
 RobotPositionManager.RobotPositionManager, 73

current_index
 RobotVision.RobotVision, 80

current_mode
 Environment, 35

debug.cc
 example, 94
 example1, 95
 main, 94, 97
 size, 95
 size1, 95
 tarefaPesada, 97

destroy
 RobotPositionManager.RobotPositionManager, 73

draw
 RobotVision.Ball, 18
 RobotVision.Elemento, 28
 RobotVision.Goal, 39
 RobotVision.Line, 43
 RobotVision.Marker, 55

draw_legend
 RobotVision.RobotVision, 79

draw_player
 RobotPositionManager.RobotPositionManager, 73

end
 Environment::Parsing, 63

env
 BaseAgent.BaseAgent, 22
 Environment::Parsing, 63
 ServerComm.ServerComm, 85

Environment, 31
 ACTIVE_BEAM, 34
 BEFORE_KICKOFF, 33
 current_mode, 35

Environment, 34
 GAME_OVER, 33
 goals_conceded, 35
 goals_scored, 35
 is_left, 35
 logger, 35
 OTHER, 34
 OUR_CORNER_KICK, 33
 OUR_DIR_FREE_KICK, 33
 OUR_FREE_KICK, 33
 OUR_GOAL, 33
 OUR_GOAL_KICK, 33
 OUR_KICK, 34
 OUR_KICK_IN, 33
 OUR_KICKOFF, 33
 OUR_OFFSIDE, 33
 OUR_PASS, 33
 PASSIVE_BEAM, 34
 play_modes, 36
 PLAY_ON, 33
 PlayMode, 33
 PlayModeGroup, 33
 print_status, 34
 THEIR_CORNER_KICK, 33
 THEIR_DIR_FREE_KICK, 33
 THEIR_FREE_KICK, 33
 THEIR_GOAL, 33
 THEIR_GOAL_KICK, 33
 THEIR_KICK, 34
 THEIR_KICK_IN, 33
 THEIR_KICKOFF, 33
 THEIR_OFFSIDE, 33
 THEIR_PASS, 33
 time_match, 36
 time_server, 36
 unum, 36
 update_from_server, 34

Environment.hpp
 False, 101
 True, 101

Environment::Enabler_Stringview_Hash, 29
 is_transparent, 30
 operator(), 30

Environment::Parsing, 57
 advance, 59
 buffer, 63
 end, 63
 env, 63
 get, 59
 get_str, 59
 get_value, 60
 is_valid, 60
 parse_accelerometer, 60
 parse_force_resistance, 60
 parse_gamestate, 61
 parse_gyroscope, 61
 parse_hear, 61
 parse_hingejoint, 61

parse_time, 62
parse_vision, 62
Parsing, 59
skip_until_char, 62
error
 Logger, 47, 48
example
 debug.cc, 94
example1
 debug.cc, 95
exec_booting, 9

False
 Environment.hpp, 101
 Logger.hpp, 111
FIELD_HEIGHT
 RobotPositionManager.RobotPositionManager, 76
FIELD_WIDTH
 RobotPositionManager.RobotPositionManager, 76
fov_h
 RobotVision.Elemento, 28
fov_v
 RobotVision.Elemento, 28
frames
 RobotVision.RobotVision, 80
FRAMES_VISION_PATH
 RobotVision.RobotVision, 80

GAME_OVER
 Environment, 33
get
 Environment::Parsing, 59
 Logger, 48
get_config_positions
 RobotPositionManager.RobotPositionManager, 73
get_input
 Printing.Printing, 65
get_str
 Environment::Parsing, 59
get_team_params
 Booting.Booting, 25
get_value
 Environment::Parsing, 60
goals_conceded
 Environment, 35
goals_scored
 Environment, 35
GRID_SCALE
 RobotPositionManager.RobotPositionManager, 76

HEIGHT
 RobotVision, 11
height
 RobotVision.Elemento, 29

IF_IN_DEBUG
 Printing.Printing, 66
info
 Logger, 48

init_position
 BaseAgent.BaseAgent, 22
INITIAL_POSITION
 BaseAgent.BaseAgent, 23
is_left
 Environment, 35
is_the_first
 Logger, 52
is_transparent
 Environment::Enabler_Stringview_Hash, 30
is_valid
 Environment::Parsing, 60

load_frames_from_file
 RobotVision.RobotVision, 79
log
 Logger, 49
Logger, 44
 _current_buffer, 51
 _cv, 51
 _file_stream, 51
 _init_file, 47
 _is_running, 51
 _mutex, 51
 _worker, 51
 _worker_loop, 47
 _write_buffer, 51
 ~Logger, 47
error, 47, 48
get, 48
info, 48
is_the_first, 52
log, 49
Logger, 47
operator=, 49
warn, 49
logger
 BaseAgent.BaseAgent, 23
 Environment, 35
Logger.hpp
 False, 111
 True, 111

main
 debug.cc, 94, 97
mainloop
 RobotVision.RobotVision, 80
marcadores_jogadores
 RobotPositionManager.RobotPositionManager, 76
MAX_JOGADORES
 RobotPositionManager.RobotPositionManager, 76
message_queue
 ServerComm.ServerComm, 85
module_main.cpp
 NB_MODULE, 107, 109

NB_MODULE
 module_main.cpp, 107, 109
need_to_update

RobotVision.RobotVision, 80
 nome_de_config_selecionada
 RobotPositionManager.RobotPositionManager, 76
 nova_config
 RobotPositionManager.RobotPositionManager, 74

 objects
 RobotVision.RobotVision, 81
 on_double_click_in_configs
 RobotPositionManager.RobotPositionManager, 74, 76

 operator()
 Environment::Enabler_Stringview_Hash, 30
 operator=
 Logger, 49
 options
 Booting.Booting, 25
 OTHER
 Environment, 34
 OUR_CORNER_KICK
 Environment, 33
 OUR_DIR_FREE_KICK
 Environment, 33
 OUR_FREE_KICK
 Environment, 33
 OUR_GOAL
 Environment, 33
 OUR_GOAL_KICK
 Environment, 33
 OUR_KICK
 Environment, 34
 OUR_KICK_IN
 Environment, 33
 OUR_KICKOFF
 Environment, 33
 OUR_OFFSIDE
 Environment, 33
 OUR_PASS
 Environment, 33

 p
 run_full_team, 11
 run_player, 12
 parse_accelerometer
 Environment::Parsing, 60
 parse_force_resistance
 Environment::Parsing, 60
 parse_frame
 RobotVision.RobotVision, 80
 parse_gamestate
 Environment::Parsing, 61
 parse_gyroscope
 Environment::Parsing, 61
 parse_hear
 Environment::Parsing, 61
 parse_hingejoint
 Environment::Parsing, 61
 parse_time
 Environment::Parsing, 62

 parse_vision
 Environment::Parsing, 62
 Parsing
 Environment::Parsing, 59
 PASSIVE_BEAM
 Environment, 34
 play_modes
 Environment, 36
 PLAY_ON
 Environment, 33
 players
 run_full_team, 11
 PlayMode
 Environment, 33
 PlayModeGroup
 Environment, 33
 posicoes_atuais
 RobotPositionManager.RobotPositionManager, 76
 position_on_sphere
 RobotVision.Ball, 19
 RobotVision.Goal, 40
 RobotVision.Line, 44
 RobotVision.Marker, 56
 position_on_window
 RobotVision.Ball, 19
 RobotVision.Goal, 40
 RobotVision.Line, 44
 RobotVision.Marker, 56
 print_message
 Printing.Printing, 65
 print_status
 Environment, 34
 print_table
 Printing.Printing, 66
 Printing, 9
 Printing.Printing, 64
 CONSOLE, 66
 get_input, 65
 IF_IN_DEBUG, 66
 print_message, 65
 print_table, 66
 TABLE_COLORS, 66
 project_point
 RobotVision.Elemento, 28
 projection_to_2d
 RobotVision.Ball, 18
 RobotVision.Elemento, 28
 RobotVision.Goal, 39
 RobotVision.Line, 43
 RobotVision.Marker, 55

 receive
 ServerComm.ServerComm, 84
 RobotPositionManager, 10
 root, 10
 RobotPositionManager.RobotPositionManager, 67
 __init__, 71
 _canvas_to_field, 71
 _field_to_canvas, 71

apagar_config, 72
canvas, 75
canvas_height, 75
canvas_width, 75
clear_grid, 72
click_on_grid, 72, 75
CONFIG_POSITION_PATH, 75
config_positions, 75
criar_widgets, 73
destroy, 73
draw_player, 73
FIELD_HEIGHT, 76
FIELD_WIDTH, 76
get_config_positions, 73
GRID_SCALE, 76
marcadores_jogadores, 76
MAX_JOGADORES, 76
nome_de_config_selecionada, 76
nova_config, 74
on_double_click_in_configs, 74, 76
posicoes_atuais, 76
salvar_config, 74
save_config_positions, 74
tv_configs, 77
update_table_config, 75
X_MAX, 77
X_MIN, 77
Y_MAX, 77
Y_MIN, 77
RobotVision, 10
HEIGHT, 11
WIDTH, 11
RobotVision.Ball, 16
 __init__, 18
 color, 19
 draw, 18
 position_on_sphere, 19
 position_on_window, 19
 projection_to_2d, 18
RobotVision.Elemento, 26
 __init__, 27
 color, 28
 draw, 28
 fov_h, 28
 fov_v, 28
 height, 29
 project_point, 28
 projection_to_2d, 28
 width, 29
RobotVision.Goal, 37
 __init__, 39
 color, 40
 draw, 39
 position_on_sphere, 40
 position_on_window, 40
 projection_to_2d, 39
RobotVision.Line, 41
 __init__, 43
 color, 44
 draw, 43
 position_on_sphere, 44
 position_on_window, 44
 projection_to_2d, 43
RobotVision.Marker, 53
 __init__, 55
 color, 56
 draw, 55
 position_on_sphere, 56
 position_on_window, 56
 projection_to_2d, 55
RobotVision.RobotVision, 78
 __init__, 79
 _get_only_tag_See, 79
 current_index, 80
 draw_legend, 79
 frames, 80
 FRAMES_VISION_PATH, 80
 load_frames_from_file, 79
 mainloop, 80
 need_to_update, 80
 objects, 81
 parse_frame, 80
root
 RobotPositionManager, 10
run_full_team, 11
 boot, 11
 p, 11
 players, 11
run_player, 12
 boot, 12
 p, 12
salvar_config
 RobotPositionManager.RobotPositionManager, 74
save_config_positions
 RobotPositionManager.RobotPositionManager, 74
scom
 BaseAgent.BaseAgent, 23
send
 ServerComm.ServerComm, 84
send_immediate
 ServerComm.ServerComm, 84
ServerComm, 12
ServerComm.ServerComm, 81
 __init__, 82
 __receive_async, 83
 buffer, 85
 buffer_size, 85
 clear_queue, 83
 close, 83
 commit, 83
 commit_beam, 84
 env, 85
 message_queue, 85
 receive, 84
 send, 84
 send_immediate, 84

socket, 85
 unum, 85
show_spinner
 Booting.Booting, 25
size
 debug.cc, 95
size1
 debug.cc, 95
skip_until_char
 Environment::Parsing, 62
socket
 ServerComm.ServerComm, 85
src/agent/Agent.py, 87
src/agent/AgentPenalty.py, 88
src/agent/BaseAgent.py, 88, 89
src/communication/ServerComm.py, 90
src/cpp/environment/debug.cc, 93, 96
src/cpp/environment/Environment.cpp, 99
src/cpp/environment/Environment.hpp, 100, 102
src/cpp/environment/module_main.cpp, 107
src/cpp/logger/debug.cc, 96, 97
src/cpp/logger/Logger.hpp, 110, 111
src/cpp/logger/module_main.cpp, 108, 109
src/exec_booting.py, 113
src/run_full_team.py, 113, 114
src/run_player.py, 114
src/term/Booting.py, 114, 115
src/term/Printing.py, 118
src/utils/RobotPositionManager.py, 120, 121
src/utils/RobotVision.py, 126

TABLE_COLORS
 Printing.Printing, 66

tarefaPesada
 debug.cc, 97

THEIR_CORNER_KICK
 Environment, 33

THEIR_DIR_FREE_KICK
 Environment, 33

THEIR_FREE_KICK
 Environment, 33

THEIR_GOAL
 Environment, 33

THEIR_GOAL_KICK
 Environment, 33

THEIR_KICK
 Environment, 34

THEIR_KICK_IN
 Environment, 33

THEIR_KICKOFF
 Environment, 33

THEIR_OFFSIDE
 Environment, 33

THEIR_PASS
 Environment, 33

time_match
 Environment, 36

time_server
 Environment, 36

True
 Environment.hpp, 101
 Logger.hpp, 111

tv_configs
 RobotPositionManager.RobotPositionManager, 77

unum
 Agent.Agent, 15
 BaseAgent.BaseAgent, 23
 Environment, 36
 ServerComm.ServerComm, 85

update_from_server
 Environment, 34

update_table_config
 RobotPositionManager.RobotPositionManager, 75

warn
 Logger, 49

WIDTH
 RobotVision, 11

width
 RobotVision.Elemento, 29

X_MAX
 RobotPositionManager.RobotPositionManager, 77

X_MIN
 RobotPositionManager.RobotPositionManager, 77

Y_MAX
 RobotPositionManager.RobotPositionManager, 77

Y_MIN
 RobotPositionManager.RobotPositionManager, 77