Reference Manual

Generated by Doxygen 1.7.3

Thu Oct 20 2011 21:48:26

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 Action Class Reference

Public Member Functions

- **Action** (Memory mem)
- void **setMem** (Memory mem)
- boolean **isBallInKickRange** ()
- void **findBall** (RoboClient rc) throws Exception
- boolean **isInInterceptDistance** ()
- void **interceptBall** (RoboClient rc) throws Exception

Public Attributes

- Memory **mem**

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

- Action.java

4.2 Brain Class Reference

Public Member Functions

- Brain ()
- Brain (Mode currentMode)
- Mode getCurrentMode ()
- void setDefensive ()
- void setOffensive ()
- String getMarked_team ()

- void setMarked_team (String marked_team)
- String getMarked_unum ()
- void setMarked_unum (String marked_unum)

4.2.1 Detailed Description

The brain serves as a place to store the Player modes, marked players for various functions, and a set of strategies for player actions.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Brain::Brain ( ) [inline]

Default constructor

4.2.2.2 Brain::Brain (Mode *currentMode*) [inline]

Constructor

Parameters

currentMode	

4.2.3 Member Function Documentation

4.2.3.1 Mode Brain::getCurrentMode ( ) [inline]

Returns

the currentMode

4.2.3.2 String Brain::getMarked_team ( ) [inline]

Returns

the marked_team

4.2.3.3 String Brain::getMarked_unum ( ) [inline]

Returns

the marked_unum

4.2.3.4 void Brain::setDefensive ( ) [inline]

Sets the player mode to defensive

4.2.3.5 void Brain::setMarked_team (String *marked_team*) [inline]

Parameters

marked_team	the marked_team to set

4.2.3.6 void Brain::setMarked_unum (String *marked_unum*) [inline]

Parameters

marked_unum	the marked_unum to set

4.2.3.7 void Brain::setOffensive ( ) [inline]

Sets the player mode to be offensive

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

- Brain.java

4.3 Field Class Reference

Public Member Functions

- Field (String side)

Public Attributes

- ArrayList< Pos > **posList** = new ArrayList()

4.3.1 Detailed Description

This creates an ArrayList that holds all the coordinates for the fixed points on the field. As the orientation of the axes depends on the side of the field the starts on, there are two sets of coordinates, each with opposite signs.

Author

Grant Hays

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Field::Field (String *side*) [inline]

Field constructor

Parameters

side	The side of the field the player's team starts on

Precondition

The side needs to be parsed from the server's (init) message and passed as the argument

Postcondition

A new Field will be created with access to an array list of all the field's fixed points

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

- Field.java

4.4 Game Class Reference

Static Public Member Functions

- static void **main** (String args[ ]) throws Exception

Package Attributes

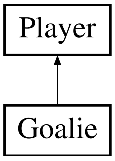
- BufferedReader **readin** = new BufferedReader(new InputStreamReader(System.in))
- String **command** = new String()

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

- Game.java

4.5 Goalie Class Reference

Inheritance diagram for Goalie:



Public Member Functions

- void initGoalie () throws SocketException, UnknownHostException
- void catchball (int dir) throws UnknownHostException

4.5.1 Detailed Description

The Goalie class inherits from the Player class. The Goalie is a specialized type of Player that may catch the ball under certain conditions and defends the goal from the opposing team.

4.5.2 Member Function Documentation

4.5.2.1 void Goalie::catchball (int *dir*) throws UnknownHostException [inline]

Causes the Goalie to catch the ball.

Precondition

> Playmode is play-on, ball is within goalkeeper zone and in the catchable area.

Postcondition

> The Goalie has caught the ball.

4.5.2.2 void Goalie::initGoalie ( ) throws SocketException, UnknownHostException [inline]

Initializes the Player with the RoboCup server as a goalie.

Precondition

> A RoboCup server is available.

Postcondition

> The Player has been initialized to the correct team as a goalie.

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

- Goalie.java

4.6 MathHelp Class Reference

Public Member Functions

- Pos getCartesian (double r, double t)

- Pos getPos (Polar p)
- Polar getPolar (double x, double y)
- Polar getPolar (Pos p)
- Pos vAdd (Pos p1, Pos p2)
- Pos vSub (Pos p2, Pos p1)
- double mag (Pos p)
- Pos norm (Pos p)
- Pos norm (double dist, Pos a)

4.6.1 Member Function Documentation

4.6.1.1 Pos MathHelp::getCartesian (double *r*, double *t*) [inline]

Polar to Cartesian converter

Parameters

r	the length of the Polar arm
t	the angle, in degrees, of the arm from the x-axis

Returns

A new Cartesian Pos converted from the r and t of a Polar vector

4.6.1.2 Polar MathHelp::getPolar (Pos *p*) [inline]

Cartesian to polar wrapper

This is just a wrapper, so you can pass in a Pos instead of extracting it's x and y and passing them in.

Parameters

p	the Cartesian vector

Returns

A new Polar vector converted from the Cartesian vector

4.6.1.3 Polar MathHelp::getPolar (double *x*, double *y*) [inline]

Cartesian to polar converter

Parameters

x	the x coordinate of the Cartesian vector
y	the y coordinate of the Cartesian vector

Returns

A new Polar vector converted from the Cartesian vector

4.6.1.4 Pos MathHelp::getPos (Polar *p*) [inline]

Polar to Cartesian wrapper

This allows you to pass a whole polar in, instead of extracting it's r and t variables and passing them in

Parameters

p	The polar coordinates you want to convert

Returns

A new Pos with the Cartesian version of your Polar vector

4.6.1.5 double MathHelp::mag (Pos *p*) [inline]

Magnitude Calculates the Magnitude of a vector, same as r in a Polar vector

Parameters

p	the Pos of the vector

Returns

A double containing the magnitude of the vector

4.6.1.6 Pos MathHelp::norm (Pos *p*) [inline]

A normalizer

Parameters

p	the vector to find the normal of

Returns

a Pos of the unit vector of p

4.6.1.7 Pos MathHelp::norm (double *dist*, Pos *a*) [inline]

A normalizer

Parameters

dist	the magnitude of the vector
a	the vector to be normalized

Returns

a Pos of the unit vector of p

4.6.1.8 Pos MathHelp::vAdd (Pos *p1*, Pos *p2*) `[inline]`

Vector Addition

Parameters

p1	first position
p2	second position

Returns

New position with the sum of the two arguments

4.6.1.9 Pos MathHelp::vSub (Pos *p2*, Pos *p1*) `[inline]`

Vector Subtraction

Parameters

p2	final position
p1	initial position

Returns

new Pos with the difference between p2 and p1

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

- MathHelp.java

4.7 Memory Class Reference

Public Member Functions

- Memory ()
- void **setField** (String side)
- ObjInfo getObj (int i)
- int getObjMemorySize ()
- boolean isObjVisible (String name)

- ObjBall getBall ()
- ObjFlag getFlag (String name)
- ObjGoal getGoal ()
- ObjPlayer getPlayer ()
- ArrayList< ObjPlayer > **getPlayers** ()
- ObjLine getClosestLine ()
- boolean **timeCheck** (int t)
- void **setLocation** (double x, double y)
- ObjFlag **getClosestFlag** ()
- ObjFlag **getClosestBoundary** ()
- Pos **getFlagPos** (String flagName)
- double **getStamina** ()
- double **getRecovery** ()
- double **getEffort** ()
- double **getAmountOfSpeed** ()
- double **getDirectionOfSpeed** ()
- double **getHeadDirection** ()
- String **getPlayMode** ()

Public Attributes

- Field **f**
- Pos **pos**
- ObjMemory ObjMem
- SenseMemory SenMem
- String playMode
- String **side**
- Double **uNum**
- Pos **OppGoal**

4.7.1 Detailed Description

Author

Grant Hays The Memory class stores instances of ObjMemory and SenseMemory and supplies methods to access their innards.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 Memory::Memory ( ) `[inline]`

The default constructor for the Memory.

This creates new, empty ArrayList for the ObjMemory and SenseMemory, initiates the time at 0 for both, and creates an ObjMemory and SenseMemory with the new ArrayLists and time as parameters.

4.7.3 Member Function Documentation

4.7.3.1 ObjBall Memory::getBall ( ) [inline]

The Ball Getter

****** Make sure you check visibility first! ******* If you don't, you will get a null object, and nobody wants that.

Returns

ObjBall containing the ball

4.7.3.2 ObjLine Memory::getClosestLine ( ) [inline]

****** Make sure you check visibility first! ******* If you don't, you will get a null object, and nobody wants that.

This will get the ObjLine closest to the player.

Returns

ObjLine

4.7.3.3 ObjFlag Memory::getFlag (String *name*) [inline]

The Flag Getter

****** Make sure you check visibility first! ******* If you don't, you will get a null object, and nobody wants that.

If you're looking for a specific flag, this is you're guy. You need to pass in the FlagName (i.e. flb30) into it, and out pops the ObjFlag with that FlagName attached to it.

Parameters

name	

Returns

ObjFlag containing the FlagName you input

4.7.3.4 ObjGoal Memory::getGoal ( ) [inline]

The Goal Getter

****** Make sure you check visibility first! ******* If you don't, you will get a null object, and nobody wants that.

This will get the ObjGoal closest to you.

Returns

ObjGoal containing the goal closest to you

4.7.3.5 ObjInfo Memory::getObj (int *i*) [inline]

The ObjInfo getter

This fetches the ObjInfo at index i of the ArrayList ObjArray in ObjMemory, and returns it as an ObjInfo.

Parameters

i	the index number of the location of the desired ObjInfo in ObjArray

Returns

ObjInfo the ObjInfo at location i of the ObjArray

4.7.3.6 int Memory::getObjMemorySize ( ) [inline]

The ObjMemory size

A getter to quickly retrieve the number of ObjInfo in ObjMemory

Returns

size of ObjMemory

4.7.3.7 ObjPlayer Memory::getPlayer ( ) [inline]

The Player Getter

****** Make sure you check visibility first! ******* If you don't, you will get a null object, and nobody wants that.

This will get the ObjPlayer of the first player you see.

Returns

ObjPlayer

4.7.3.8 boolean Memory::isObjVisible (String *name*) [inline]

Is this ObjInfo visible?

Parameters

name	the ObjName of the ObjInfo we're detecting visibility of

Returns

true if the ball is in the ObjMemory, false if it is not or if the the ObjMemory is empty

4.7.4 Member Data Documentation

4.7.4.1 ObjMemory Memory::ObjMem

The memory that stores all parsed ObjInfo

4.7.4.2 String Memory::playMode

The play mode as told by the referee

4.7.4.3 SenseMemory Memory::SenMem

The memory that stores all parsed SenseInfo

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

- Memory.java

4.8 Mode Class Reference

Public Member Functions

- Mode (String modename, double timeinmode)
- String getModename ()
- void setModename (String modename)
- double getTimeinmode ()
- void setTimeinmode (double timeinmode)

4.8.1 Detailed Description

The Mode class is a basic data structure to store the parameters for the player modes.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 Mode::Mode (String *modename*, double *timeinmode*) `[inline]`

Parameters

modename	
timeinmode	

4.8.3 Member Function Documentation

4.8.3.1 String Mode::getModename ( ) `[inline]`

Returns

the modename

4.8.3.2 double Mode::getTimeinmode ( ) `[inline]`

Returns

the timeinmode

4.8.3.3 void Mode::setModename (String *modename*) `[inline]`

Parameters

modename	the modename to set

4.8.3.4 void Mode::setTimeinmode (double *timeinmode*) `[inline]`

Parameters

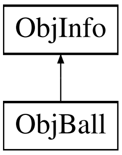
timeinmode	the timeinmode to set

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

- Mode.java

4.9 ObjBall Class Reference

Inheritance diagram for ObjBall:



4.9.1 Detailed Description

container for the ball ObjInfo,

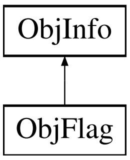
container for the flag ObjInfo,

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

- ObjInfo.java

4.10 ObjFlag Class Reference

Inheritance diagram for ObjFlag:



Public Member Functions

- ObjFlag (String name)
- String getFlagType ()
- void setFlagType (String flagType)
- String getFlagName ()
- void setFlagName (String name)
- String getX_pos ()
- void setX_pos (String x_pos)
- String getY_pos ()
- void setY_pos (String y_pos)
- String getYard ()
- void setYard (String yard)

4.10.1 Constructor & Destructor Documentation

4.10.1.1 ObjFlag::ObjFlag (String *name*) [inline]

Constructor of flag with flag name

4.10.2 Member Function Documentation

4.10.2.1 String ObjFlag::getFlagName ( ) [inline]

The Flag Name getter

Returns

The name of the flag, as given by the server but with no spaces (e.g. flt20 for boundary flag left, top, 20 yard line)

4.10.2.2 String ObjFlag::getFlagType ( ) [inline]

The Flag Type getter

Returns

The type of flag depending on it's location: "b" - outer boundary "g" - goal post "p" - penalty box "c" - center of field "l" - border line

4.10.2.3 String ObjFlag::getX_pos ( ) [inline]

The X position getter

Returns

Either "l" for left, "r" for right, or "c" for center

4.10.2.4 String ObjFlag::getY_pos ( ) [inline]

The Y position getter

Returns

Either "t" for top, "b" for bottom, or "c" for center

4.10.2.5 String ObjFlag::getYard ( ) [inline]

The yard getter

Returns

the yard is a String of a number for boundaries

4.10.2.6 void ObjFlag::setFlagName (String *name*) [inline]

The Flag Name setter

4.10.2.7 void ObjFlag::setFlagType (String *flagType*) [inline]

The Flag Type setter

4.10.2.8 void ObjFlag::setX_pos (String *x_pos*) [inline]

The X position setter

4.10.2.9 void ObjFlag::setY_pos (String *y_pos*) [inline]

The Y position setter

4.10.2.10 void ObjFlag::setYard (String *yard*) [inline]

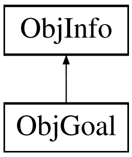
The yard setter

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

- ObjInfo.java

4.11 ObjGoal Class Reference

Inheritance diagram for ObjGoal:



4.11.1 Detailed Description

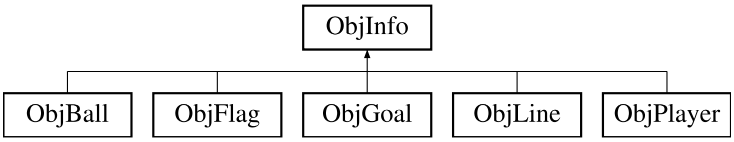
container for the goal ObjInfo,

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

- ObjInfo.java

4.12 ObjInfo Class Reference

Inheritance diagram for ObjInfo:



Public Member Functions

- ObjInfo ()
- ObjInfo (String name)
- String getObjName ()
- void setObjName (String name)
- String getSide ()
- void setSide (String objSide)
- double getDistance ()
- void setDistance (double distance)
- double getDirection ()
- void setDirection (double direction)
- double getDistChng ()
- void setDistChng (double distChng)
- double getDirChng ()
- void setDirChng (double dirChng)

4.12.1 Detailed Description

A container for items in the Player's vision

4.12.2 Constructor & Destructor Documentation

4.12.2.1 ObjInfo::ObjInfo ( ) [inline]

The Default constructor

4.12.2.2 ObjInfo::ObjInfo (String *name*) [inline]

The ObjInfo constructor

This initializes all the variables to 0.0 and sets the name

Parameters

name	The type of ObjInfo, either ball, player, goal, line, or flag

4.12.3 Member Function Documentation

4.12.3.1 double ObjInfo::getDirChng ( ) [inline]

The direction change getter

Returns

the approximate direction change (direction of velocity) of ObjInfo

4.12.3.2 double ObjInfo::getDirection ( ) [inline]

The direction getter

Returns

the approximate direction of ObjInfo

4.12.3.3 double ObjInfo::getDistance ( ) [inline]

The distance getter

Returns

the approximate distance to the object

4.12.3.4 double ObjInfo::getDistChng ( ) [inline]

The distance change getter

Returns

the approximate distance change (magnitude of velocity) of ObjInfo

4.12.3.5 String ObjInfo::getObjName ( ) [inline]

The ObjName getter

4.12.3.6 String ObjInfo::getSide ( ) [inline]

The side getter

4.12.3.7 void ObjInfo::setDirChng (double *dirChng*) [inline]

The distance change setter

4.12.3.8 void ObjInfo::setDirection (double *direction*) [inline]

The direction setter

4.12.3.9 void ObjInfo::setDistance (double *distance*) [inline]

The distance setter

4.12.3.10 void ObjInfo::setDistChng (double *distChng*) [inline]

The distance change setter

4.12.3.11 void ObjInfo::setObjName (String *name*) [inline]

The ObjName setter

4.12.3.12 void ObjInfo::setSide (String *objSide*) [inline]

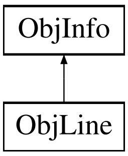
The side setter

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

- ObjInfo.java

4.13 ObjLine Class Reference

Inheritance diagram for ObjLine:



4.13.1 Detailed Description

container for line ObjInfo

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

- ObjInfo.java

4.14 ObjMemory Class Reference

Public Member Functions

- ObjMemory ()
- ObjMemory (ArrayList< ObjInfo > ObjArray, int t)
- void addInfo (ObjInfo newInfo)
- int getTime ()
- void setTime (int t)
- int getSize ()
- ObjInfo getObj (int index)
- ObjInfo getObj (String name)

Public Attributes

- ArrayList< ObjInfo > **ObjArray**

4.14.1 Detailed Description

The ObjMemory saves all the ObjInfo (and it's children) objects from a parse into ArrayList along with the time parsed.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 ObjMemory::ObjMemory ( ) [inline]

Default constructor

This initializes the time to 0

4.14.2.2 ObjMemory::ObjMemory (ArrayList< ObjInfo > *ObjArray*, int *t*) [inline]

ObjMemory constructor

Parameters

ObjArray	the ArrayList containing all the ObjInfos from the server's parsed (see) message
t	the time parsed from the server's (see) message

Precondition

This should only be called inside of the parser. It's merely a way to store ObjInfos from the (see) message into the greater Memory class

Postcondition

A new ObjMemory containing the list of visible ObjInfos and the most recent time will be availbe to add to the Memory

4.14.3 Member Function Documentation

4.14.3.1 void ObjMemory::addInfo (ObjInfo *newInfo*) [inline]

A method to add new ObjInfo to the ObjMemory

Parameters

newInfo	the ObjInfo to add tot he ObjMemory's ArrayList

Precondition

A non-null ObjInfo will be passed into the method

Postcondition

The newInfo will be added to the ObjArray

4.14.3.2 ObjInfo ObjMemory::getObj (int *index*) [inline]

An accessor of individual ObjInfo

Parameters

index	the index of the ObjInfo to retrieve

Precondition

The ObjArray should have at least one ObjInfo in it

Postcondition

The ObjInfo at the given index will be returned, this is a good way to traverse the ObjInfos visible to you

4.14.3.3 ObjInfo ObjMemory::getObj (String *name*) [inline]

A method to get an ObjInfo by name

Parameters

name	the ObjName of the ObjInfo searched for (e.g. "ball")

Precondition

The ObjInfo should be checked for visibility first, otherwise you run the risk of getting an empty ObjInfo

Postcondition

The first ObjInfo with the name will be returned. Remember, this won't return all the ObjInfos of an ObjName, if there are multiple.

4.14.3.4 int ObjMemory::getSize ( ) `[inline]`

Returns the size of the ObjArray

4.14.3.5 int ObjMemory::getTime ( ) `[inline]`

A method to access the time the message was parsed, provided by the server's (see) message

4.14.3.6 void ObjMemory::setTime (int *t*) `[inline]`

The time setter

Parameters

t	the time integer from the server's latest (see) parse

Postcondition

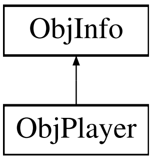
the time will be set and ready to access

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

- ObjMemory.java

4.15 ObjPlayer Class Reference

Inheritance diagram for ObjPlayer:



Public Member Functions

- String getTeam ()
- void setTeam (String team)
- int getuNum ()

- void setuNum (int uNum)
- boolean isGoalie ()
- void setGoalie (boolean goalie)
- double getHeadDir ()
- void setHeadDir (double headDir)
- double getBodyDir ()
- void setBodyDir (double bodyDir)

4.15.1 Detailed Description

container for player ObjInfo

4.15.2 Member Function Documentation

4.15.2.1 double ObjPlayer::getBodyDir ( ) [inline]

A getter for the player's body direction

Returns

> a double of the angle, in degrees, of the direction of the player's body relative to your own. The angle is 0 if their bodies are both facing each other.

4.15.2.2 double ObjPlayer::getHeadDir ( ) [inline]

A getter for the player's head direction

Returns

> a double of the angle, in degrees, of the direction of the player's head relative to your own. The angle is 0 if they are both facing each other.

4.15.2.3 String ObjPlayer::getTeam ( ) [inline]

The Team Name getter

Returns

> the name of the team the player is on, if they're close enough to see the team

4.15.2.4 int ObjPlayer::getuNum ( ) [inline]

The Uniform Number getter

Returns

> the Uniform Number on the player's shirt, if they're close enough to see it

4.15.2.5 boolean ObjPlayer::isGoalie ( ) [inline]

A check to see if the player is a goalie or field player

Returns

true if the player is the goalie, false if s/he is not

4.15.2.6 void ObjPlayer::setBodyDir (double *bodyDir*) [inline]

The body direction setter

4.15.2.7 void ObjPlayer::setGoalie (boolean *goalie*) [inline]

The goalie check setter

4.15.2.8 void ObjPlayer::setHeadDir (double *headDir*) [inline]

The head direction setter

4.15.2.9 void ObjPlayer::setTeam (String *team*) [inline]

The Team Name setter

4.15.2.10 void ObjPlayer::setuNum (int *uNum*) [inline]

The Uniform Number getter

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

- ObjInfo.java

4.16 Parser Class Reference

Public Member Functions

- Parser ()
- void initParse (String inputPacket, Memory mem)
- void Parse (String inputPacket, Memory InfoMem)

Public Attributes

- String input

4.16.1 Detailed Description

This class takes in the the messages sent by the parser and parses them into information that can be stored in Memory and used by Players.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 Parser::Parser ( ) [inline]

Default constructor

4.16.3 Member Function Documentation

4.16.3.1 void Parser::initParse (String *inputPacket*, Memory *mem*) [inline]

This parses the (init) message, the first message sent by the server, directly after a new Player is initialized.

Parameters

inputPacket	The init message from the server
mem	the player's memory

Precondition

> A memory must be created for the information to be stored in, and this must be called directly after an (init) is sent to the server.

Postcondition

> Vital information about the Player will be saved, such as the side of the field the player starts on, the Player's uniform number and the play mode, which is "before_kickoff."

4.16.3.2 void Parser::Parse (String *inputPacket*, Memory *InfoMem*) [inline]

The actual message Parsing method

Parameters

inputPacket	the incoming String message from the server
InfoMem	the Memory to store all the information in

Precondition

> A Memory must be created and passed in, along with the message from the server

Postcondition

> The message will be parsed and stored either as SenseInfos from the (sense_body) message, or ObjInfos from the (see) message, or the playMode from the referee (hear) message

4.16.4 Member Data Documentation

4.16.4.1 String **Parser::input**

The String of the incoming message

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

- Parser.java

4.17 ParserTest Class Reference

Static Public Member Functions

- static void **main** (String args[ ]) throws Exception

Package Attributes

- BufferedReader **readin** = new BufferedReader(new InputStreamReader(System.in))
- String **command** = new String()

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

- ParserTest.java

4.18 ParserTest2 Class Reference

Static Public Member Functions

- static void **main** (String args[ ]) throws Exception

Package Attributes

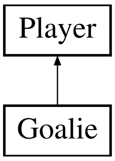
- BufferedReader **readin** = new BufferedReader(new InputStreamReader(System.in))
- String **command** = new String()

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

- ParserTest2.java

4.19 Player Class Reference

Inheritance diagram for Player:



Public Member Functions

- Player (RoboClient rc, Memory m, ObjInfo i, Parser p, Brain b, int time)
- Brain getBrain ()
- void setBrain (Brain b)
- RoboClient getRoboClient ()
- void setRoboclient (RoboClient rc)
- Memory getMem ()
- void setMem (Memory m)
- ObjInfo getObjInfo ()
- void setObjInfo (ObjInfo i)
- Parser getParser ()
- void setParser (Parser p)
- int getTime ()
- void setTime (int time)
- void initPlayer () throws SocketException, UnknownHostException
- void receiveInput () throws InterruptedException
- void move (int x, int y) throws UnknownHostException, InterruptedException
- void kick (double power, double dir) throws UnknownHostException, InterruptedException
- void **dash** (double power) throws Exception
- void turn (double moment) throws UnknownHostException, InterruptedException
- void say (String message) throws UnknownHostException, InterruptedException
- void **markOpponent** (String team, String number)
- void **runDefense** ()

Protected Attributes

- RoboClient **rc** = new RoboClient()

4.19.1 Detailed Description

The Player class defines all objects and methods used for the Player within the RoboCup match. The Player establishes a connection to the server, initializes itself on the team, and performs all actions related to a RoboCup soccer player such as (but not limited to) kicking, dashing, dribbling, passing and scoring. The Player class has a Memory for storing the current RoboCup worldstate. It reacts to stimuli based on strategies provided by the Brain (TBD).

4.19.2 Constructor & Destructor Documentation

4.19.2.1 Player::Player (RoboClient *rc*, Memory *m*, ObjInfo *i*, Parser *p*, Brain *b*, int *time*) `[inline]`

Parameters

rc	
m	
i	
p	
b	
time	

4.19.3 Member Function Documentation

4.19.3.1 Brain Player::getBrain ( ) `[inline]`

Returns

the b

4.19.3.2 Memory Player::getMem ( ) `[inline]`

Returns

The Memory for this Player.

4.19.3.3 ObjInfo Player::getObjInfo ( ) `[inline]`

Returns

The ObjInfo for this Player.

4.19.3.4 Parser Player::getParser ( ) `[inline]`

Returns

The Parser for this Player.

4.19.3.5 RoboClient Player::getRoboClient ( ) [inline]

Returns

The RoboClient object for this Player.

4.19.3.6 int Player::getTime ( ) [inline]

Returns

the time

4.19.3.7 void Player::initPlayer ( ) throws SocketException, UnknownHostException [inline]

Initializes the Player with the RoboCup server.

Precondition

A RoboCup server is available.

Postcondition

The Player has been initialized to the correct team.

4.19.3.8 void Player::kick (double *power*, double *dir*) throws UnknownHostException, InterruptedException [inline]

Causes Player to kick the ball.

Parameters

dir	The direction in which to kick the ball in the form of a decimal value. representing the angle in degrees in relation go the player.
power	The power with which to kick the ball in the form of a decimal value.

Exceptions

InterruptedException	

Precondition

Playmode is play_on, ball is in kickable range.

Postcondition

The ball has been kicked in the specified direction and power.

4.19.3.9 void Player::move (int *x*, int *y*) throws UnknownHostException, InterruptedException [inline]

Teleports the Player to the specified coordinates.

Parameters

x	x-coordinate of the point to move the player to.
y	y-coordinate of the point to move the player to.

Exceptions

InterruptedException	

Precondition

Playmode is before-kickoff, goal-scored, free-kick.

Postcondition

The Player has been moved to the correct position.

4.19.3.10 void Player::receiveInput ( ) throws InterruptedException [inline]

Receives worldstate data from the RoboCup server.

Precondition

A RoboCup server is available.

Postcondition

The current worldstate has been stored in the Memory.

4.19.3.11 void Player::say (String *message*) throws UnknownHostException, InterruptedException [inline]

Causes Player to say the given message. It has a limitation of 512 characters by default.

Parameters

message	The string to be spoken by the player.

Exceptions

InterruptedException	

Precondition

None

Postcondition

The player has spoken the message.

4.19.3.12 void Player::setBrain (Brain *b*) [inline]

Parameters

b	the b to set

4.19.3.13 void Player::setMem (Memory *m*) [inline]

Parameters

m	The Memory to set.

4.19.3.14 void Player::setObjInfo (ObjInfo *i*) [inline]

Parameters

i	The ObjInfo to set.

4.19.3.15 void Player::setParser (Parser *p*) [inline]

Parameters

p	The Parser to set.

4.19.3.16 void Player::setRoboclient (RoboClient *rc*) [inline]

Parameters

rc	The RoboClient to set.

4.19.3.17 void Player::setTime (int *time*) [inline]

Parameters

time	the time to set

4.19.3.18 void Player::turn (double *moment*) throws UnknownHostException, InterruptedException [inline]

Causes Player to turn according to a specified turn moment.

Parameters

moment	The turn angle in degrees.

Exceptions

InterruptedException	

Precondition

Playmode is play_on, ball is in kickable range.

Postcondition

The ball has been kicked in the specified direction and power.

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

- Player.java

4.20 Polar Class Reference

Public Member Functions

- Polar ()
- Polar (double r, double t)

Public Attributes

- double **r**
- double **t**

4.20.1 Detailed Description

A container for polar coordinates. It holds distance (r) and direction (t) of an object with respect to the player.

Author

Grant Hays

Date

10/14/ll

Version

1

4.20.2 Constructor & Destructor Documentation

4.20.2.1 Polar::Polar ( ) `[inline]`

Default constructor

Postcondition

initializes distance and angle to 0.0

4.20.2.2 Polar::Polar (double *r*, double *t*) `[inline]`

Constructor with parameters

Parameters

r	The length of the distance to the object
t	The angle of the object from the players line of sight

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

- Polar.java

4.21 Pos Class Reference

Public Member Functions

- Pos ()
- Pos (String name, double x, double y)
- Pos (double x, double y)

Public Attributes

- String **name**
- double **x**
- double **y**

4.21.1 Detailed Description

This class holds the information for Cartesian coordinate versions of positions of players and objects

4.21.2 Constructor & Destructor Documentation

4.21.2.1 Pos::Pos ( ) [inline]

Default constructor

Postcondition

> initializes x and y to 0 and name to space, so as not to have a pointer error

4.21.2.2 Pos::Pos (String *name*, double *x*, double *y*) [inline]

Constructor with name

This is a constructor for coordinates that are given a name. It is mostly used for the positions of the flags in the Field class

Parameters

name	The name associated with the Pos, for easier searching
x	x-coordinate
y	y-coordinate

4.21.2.3 Pos::Pos (double *x*, double *y*) [inline]

Constructor with no name

This is a constructor for positions that aren't given a name. Used for positions that change often.

Parameters

x	x-coordinate
y	y-coordinate

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

- Pos.java

4.22 RoboClient Class Reference

Public Member Functions

- void send (String message) throws UnknownHostException
- String receive ()
- void init (Parser p, Memory m) throws UnknownHostException
- void initGoalie () throws UnknownHostException

- void dash (double power) throws Exception
- void kick (double power, double dir) throws UnknownHostException
- void turn (double moment) throws UnknownHostException
- void move (int x, int y) throws UnknownHostException
- void catchball (int dir) throws UnknownHostException
- void say (String message) throws UnknownHostException

Public Attributes

- DatagramSocket **dsock**

Package Attributes

- String **reply**

4.22.1 Detailed Description

The RoboClient class operates as a client for the RoboCup session. It is mainly designed to be used by the Player class to handle all client-server communication. The connection protocol is UDP.

4.22.2 Member Function Documentation

4.22.2.1 void RoboClient::catchball (int *dir*) throws UnknownHostException [inline]

This function causes the active player to catch the ball. It can only be used by a Goalie type player.

Parameters

dir	An integer value representing the direction from which to catch the ball.

Precondition

Playmode is play_on or goal_kick, ball is in catchable area.

Postcondition

The player has caught the ball.

Exceptions

UnknownHostException	

4.22.2.2 void RoboClient::dash (double *power*) throws Exception `[inline]`

This function sends the dash command to the server.

Parameters

power,:	a double representing the power of the dash.

Precondition

The RoboCup server is available, client has been initialized.

Postcondition

The player has dashed according to the given power.

Returns

None

4.22.2.3 void RoboClient::init (Parser *p*, Memory *m*) throws UnknownHostException `[inline]`

This function initializes the client with the RoboCup server.

Precondition

The RoboCup server is hosting connections.

Postcondition

The client has been initialized.

4.22.2.4 void RoboClient::initGoalie ( ) throws UnknownHostException `[inline]`

This function initializes the client as a goalie with the RoboCup server.

Parameters

message,:	none

Precondition

The RoboCup server is hosting connections.

Postcondition

The goalie has been initialized.

Returns

None

4.22.2.5 void RoboClient::kick (double *power,* double *dir*) throws UnknownHostException [inline]

This function causes the active player to kick.

Parameters

power,:	a double representing the power of the kick.
dir,:	a double representing the direction of the kick.

Precondition

The RoboCup server is available, team has been initialized.

Postcondition

The player has kicked according to the given power and direction.

Returns

None

4.22.2.6 void RoboClient::move (int *x,* int *y*) throws UnknownHostException [inline]

This function causes the active player to be teleported to a given set of coordinates within the soccer field.

Parameters

x,:	an integer value for the x-coordinate to move to.
y,:	an integer value for the y-coordinate to move to.

Precondition

The RoboCup server is available, team has been initialized, kickoff has not yet occurred.

Postcondition

The player has moved to the given coordinates.

Returns

None

4.22.2.7 String RoboClient::receive ( ) [inline]

This function receives a UDP packet from the RoboCup server, and converts it to a String.

Precondition

The RoboCup server is available.

Postcondition

The packet from the RoboCup server has been processed.

Returns

String

4.22.2.8 void RoboClient::say (String *message*) throws UnknownHostException [inline]

This function causes the active player to speak the given message.

Parameters

message	A string representing the message to be spoken by the player.

Precondition

None

Postcondition

The player has spoken the message.

Exceptions

UnknownHostException	

4.22.2.9 void RoboClient::send (String *message*) throws UnknownHostException [inline]

This function reads in a message string, and sends it to the RoboCup server. It primarily serves as a method to send commands to the server to control server and player actions.

Parameters

message,:	A String.

Precondition

message is a valid String value, the RoboCup server is available.

Postcondition

The message has been delivered to the RoboCup server.

Returns

None

4.22.2.10 void RoboClient::turn (double *moment*) throws UnknownHostException [inline]

This function causes the active player to turn.

Parameters

moment,:	a double representing the turning angle in degrees.

Precondition

The RoboCup server is available, team has been initialized.

Postcondition

The player has turned the given number of degrees from original orientation.

Returns

None

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

- RoboClient.java

4.23 SenseMemory Class Reference

Public Member Functions

- SenseMemory ()
- SenseMemory (int time)
- int getTime ()
- void setTime (int t)
- void setTime (String[ ] seeOrSense)

Public Attributes

- double **stamina**
- double **recovery**
- double **effort**
- double **amountOfSpeed**
- double **directionOfSpeed**
- double **headDirection**

4.23.1 Detailed Description

This holds all the usable information parsed from the (sense_body) message sent from the server. It holds information about a Player's stamina, speed, and head direction angle, as well as the time parsed.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 SenseMemory::SenseMemory ( ) `[inline]`

Default constructor

Postcondition

initializes time to 0

4.23.2.2 SenseMemory::SenseMemory (int *time*) `[inline]`

Constructor with time

Parameters

time	The time the information was parsed, as told by the server.

Postcondition

A new SenseMemory with updated time

4.23.3 Member Function Documentation

4.23.3.1 int SenseMemory::getTime ( ) `[inline]`

The time getter

Returns

the time that the SenseMemory was parsed

4.23.3.2 void SenseMemory::setTime (String[] *seeOrSense*) `[inline]`

Time setter from the unparsed message sent by server

Parameters

seeOrSense	A String array with the split first argument of a (see) message from the server

4.23.3.3 void SenseMemory::setTime (int *t*) `[inline]`

The time setter

Parameters

t	the time hat the SenseMemory was parsed

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

- SenseMemory.java

4.24 TestDefensive Class Reference

Static Public Member Functions

- static void main (String[ ] args) throws Exception

4.24.1 Member Function Documentation

4.24.1.1 static void TestDefensive::main (String[ ] *args*) throws Exception [inline, static]

Parameters

args	

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

- TestDefensive.java

4.25 TestRoboClient Class Reference

Static Public Member Functions

- static void **main** (String args[ ]) throws Exception

Package Attributes

- BufferedReader **readin** = new BufferedReader(new InputStreamReader(System.in))
- String **command** = new String()

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

- TestRoboClient.java

4.26 TestSay Class Reference

Static Public Member Functions

- static void main (String[ ] args) throws Exception

4.26.1 Member Function Documentation

4.26.1.1 static void TestSay::main (String[] *args*) throws Exception `[inline, static]`

Parameters

args	

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

- TestSay.java

Chapter 5

File Documentation

5.1 Brain.java File Reference

Classes

- class Brain

5.1.1 Detailed Description

Author

Joel ∗

5.2 Field.java File Reference

Classes

- class Field

5.2.1 Detailed Description

A container for fixed points.

Author

Grant Hays

Date

10/13/11

Version

1

5.3 Goalie.java File Reference

Classes

- class Goalie

5.3.1 Detailed Description

Class file for Goalie class

Author

Joel Tanzi

Date

11 October 2011

Version

1.0

5.4 MathHelp.java File Reference

Classes

- class MathHelp

5.4.1 Detailed Description

This has functions of the math I need for calculations.

Author

granthays

Date

10/09/11

Version

135‘

5.5 Mode.java File Reference

Classes

- class Mode

5.5.1 Detailed Description

Author

Joel Tanzi∗

5.6 ObjInfo.java File Reference

Classes

- class ObjInfo
- class ObjBall
- class ObjGoal
- class ObjFlag
- class ObjPlayer
- class ObjLine

5.6.1 Detailed Description

The ObjInfo container

Author

Grant Hays

Date

09/01/11

Version

1

5.7 ObjMemory.java File Reference

Classes

- class ObjMemory

5.7.1 Detailed Description

A container for ObjInfo's visible to the player after a parse

Author

Grant Hays

Date

09/03/11

Version

1

5.8 Parser.java File Reference

Classes

- class Parser

5.8.1 Detailed Description

The server message parser.

Author

Grant Hays

Date

10/1/11

Version

2

5.9 Player.java File Reference

Classes

- class Player

5.9.1 Detailed Description

Class file for Player class

Author

Joel Tanzi

Date

11 October 2011

Version

1.0

5.10 Pos.java File Reference

Classes

- class Pos

5.10.1 Detailed Description

The Position vector for Cartesian Coordinates

Author

Grant Hays

Date

10/11/11

Version

1

5.11 RoboClient.java File Reference

Classes

- class RoboClient

5.11.1 Detailed Description

Class file for RoboClient class

Author

Joel Tanzi

Date

September 20, 2011

Version

1.2

5.12 SenseMemory.java File Reference

Classes

- class SenseMemory

5.12.1 Detailed Description

Container for parsed (sense_body) information

Author

Grant Hays

Date

09/10/11

Version

1

Index