**Macintosh HD:Users:rruzzo:Desktop:logo.pngCapstone Analytics**

Software Application Programming Guide

Version 1.0

Table of Contents

1 Preface 4

2 Introduction 4

3 LinearRegression.php 4

3.1 Variables 4

3.1.1 meanX 4

3.1.2 stddevX 4

3.1.3 sumX 4

3.1.4 sumXsqr 4

3.1.5 meanY 4

3.1.6 stddevY 4

3.1.7 sumY 4

3.1.8 sumYsqr 4

3.2 Class Functions 5

3.2.1 getMeanX() 5

3.2.2 getStdDevX() 5

3.2.3 getSumX() 5

3.2.4 getSumXSquared() 6

3.2.5 getSumXSquared() 6

3.2.6 getMeanY() 6

3.2.7 getStdDevY() 7

3.2.8 getSumX() 7

3.2.9 getSumXSquared() 7

3.2.10 getSumYSquared() 7

3.2.11 getPsum() 8

3.2.12 getCount() 8

3.2.13 getMeanX() 8

3.2.14 getStdDevX() 9

3.2.15 getSumX() 9

3.2.16 getSumXSquared() 9

3.2.16 getMeanY() 10

3.2.17 getStdDevY() 10

3.2.18 getSumY() 10

3.2.19 getSumYSquared() 10

3.2.20 getPsum() 11

3.2.21 getCount() 11

3.2.22 setMeanX() 11

3.2.23 setStdDevX() 12

3.2.24 setSumX() 12

3.2.25 setSumXSquared() 12

3.2.26 setMeanY() 13

3.2.27 setStdDevY() 13

3.2.29 setSumY() 13

3.2.30 setSumYSquared() 13

3.2.31 setPsum() 14

3.2.32 setCount() 14

3.2.33 getCorrelationCoefficient() 15

3.2.34 getSlope() 15

3.2.35 getIntercept() 15

3.2.36 getY() 16

3.2.37 toString() 16

4 LinearRegressionConstants.php 16

4.1 class LinearRegressionConstants 16

4.2 class PlayerDataPointConstants 17

5 LinearRegressionCriteria.php 17

5.1 Variables 17

5.1.1 tableX 17

5.1.2 tableY 17

5.1.3 rowX 17

5.1.4 rowY 17

5.1.5 joinX 17

5.1.6 joinY 17

5.1.7 sql 17

5.2 Class Functions 18

5.2.1 \_construct() 18

5.2.1 addInnerJoin() 18

5.2.3 getLinearRegressionSql() 18

5.2.4 getPlayerDataPointSql() 19

5.2.4 addJoins() 19

6 LinearRegressionUtils.php 19

6.1 rowToLinearRegression 19

6.2 rowsToPlayerDataPoints 20

7 PlayerDataPoint.php 20

7.1 Variables 20

7.1.1 playerId 20

7.1.2 firstName 20

7.1.3 lastName 20

7.1.4 x 20

7.1.5 y 20

7.1.3 expectedX 20

7.1.4 expectedY 20

7.2 Class Functions 21

7.2.1 getPlayerId 21

7.2.2 getFirstName 21

7.2.3 getLastName 21

7.2.4 getXValue 21

7.2.5 getYValue 22

7.2.6 getExpectedX 22

7.2.7 getExpectedY 22

7.2.8 getDifferenceX 23

7.2.9 getDifferenceY 23

7.2.10 setPlayerId 23

7.2.11 setFirstName 23

7.2.12 setLastName 24

7.2.13 setXValue 24

7.2.14 setYValue 24

7.2.15 setExpectedX 25

7.2.16 setExpectedY 25

# 1 Preface

This document provides an overview and explanation of the functions in the software package for Capstone Analytics’ Sports Analytics Software Package.

# 2 Introduction

2.1 Purpose

Purpose will be ignored in accordance with the Grading Rubric.

## 2.2 Scope of the Product

The product has extracted annual statistics and salaries from publicly provided databases. With both performance based statistics and current salaries, we have done statistical analysis to determine the fair market value of a player.

Once the statistical analysis is complete, players that are underpaid based on performance would be identified as optimal candidates for acquisition. Conversely overpaid players could be avoided. Ultimately the user could use these tools to maximize their profits by ensuring they avoid signing bad contracts.

## 2.3 Definitions, acronyms, and abbreviations

LAMP TO DO Define these

Regression

Mean

Standard Deviation

Sum

Intercept

Slope

## 2.4 References

http://www.seanlahman.com/baseball-archive/statistics

Used for statistics references.

# 3 General Description

## 3.1 Product Perspective

Forbes evaluates Major League Baseball at approximately 36 billion dollars and the average team being worth 1.2 billion dollars. Even with these large values, payroll for players make up a large part of the expenses for baseball clubs. Depending on revenue and revenue sharing amongst the teams, MLB has attempted to minimize the advantage larger clubs have over small clubs. However, the 2014 payroll shows a gap of 190 million dollars between the highest and lowest payroll. This product would help smaller clubs evaluate players on an objective basis in order to identify players that should be acquired at a minimal cost.

## 3.2 Product Functions

The software will allow users to select up to four statistical categories to evaluate. The user will be able to weigh the importance of each statistic being evaluated. This will allow the user to increase the importance of power (homeruns) or contact (hits) hitting styles as appropriate to find their ideal replacement player. After the search is complete, the program will return the players the most undervalued players in a list.

## 3.3 User Characteristics

The initial market for users of this program would be teams and agents involved in MLB. However, expansions should be made into the fantasy sports market, as there is a much larger audience that could offset the cost of production. Further releases could focus on development of a mobile specific web application or smart phone application using the same design to evaluate players.

## 3.4 General Constraints

The current version 1.0 will be a web based only application. The user will require internet access in order to use the application. Other limitations are that the user will require their own web browser in order to access the site.

## 3.5 Assumptions and Dependencies

The assumptions will be that the user will require a compatible web browser. Web browsers supported include Chrome, Internet Explorer, Safari and Firefox. Versions updated as of the 15th of April 2015 should be compatible. View of the website will also depend on firewall settings and not being blocked by other safe viewing mechanisms.

# 4 Specific Requirements Large Revisions needed here, Versioning

## 4.1 Database

A data base will need to be stored on the web application server in order to allow the front end to access statistics of players to be evaluated.

### 4.1.1 Master

Master will account for one member fields in the database and will contain information to identify the baseball player.

### 4.1.2 Salaries

Fielding will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

### 4.1.3 Teams

Teams will be a member field in the database to account for the team a player is currently under contract with.

### 4.1.4 Appearances

Plate appearance will be a member field in the database to account for how many times the player has appeared for home plate.

### 4.1.5 Batting

Batting will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

### 4.1.6 Fielding

Fielding will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

### 4.1.7 Pitching

Pitching will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

## 4.2 Security

The web application will require security to ensure that subscription services are being used. Username and Password will be required. Those will be obtained via e-mail through subscription services.

### 4.2.1 Username

Usernames will be verified as a unique id prior to being issued by subscription services. Once username is established a field will be provided for login and correct entry will be required to gain access to web application.

### 4.2.2 Password

Passwords will be issued by subscription services. Passwords can be changed but must be verified as 8 characters at least one upper and lowercase letter, one number, and a special character (!@#$%^&\*()\_-+/\’:,?{}[]~). Password entry field should be next to or below Username.

## 4.3 Application

The web application will provide analysis of baseball players using statistics with salary to determine what the player’s market value is.

## 4.3.1 Linear Regression

Linear regression is an approach for modeling the relationship between a scalar dependent variable y (salary) and one or more explanatory variables or independent variable (this will be a baseball statistic) denoted X. The application will utilize linear regression to determine fair market value.

## 4.3.2 Standard Deviation

The standard deviation is a measure that is used to quantify the amount of variation or dispersion of a set of data values. The application will apply this variation to determine the variation in salary. This will also be based upon the choice of statistic.

## 4.3.3 Minimum Plate Appearances

This will be used as population to control to provide more meaningful data sets. It will serve to remove players who did not play a statistically significant amount of time.

## 4.4 Application GUI

The web application GUI will provide choice and feed back to the user to manipulate the data for their use.

## 4.4.1 Statistic Drop Down Menu

This will provide a drop down menu to choose from batting statistics in order to perform analysis on the database and provide feedback to the user.

## 4.4.2 Minimum Plate Appearances Drop Down Menu

This will provide a drop down menu to choose minimum plate appearances in order to filter players who did not have a significant amount of plate appearances. This will remove players from consideration to provide tailored results.

## 4.4.3 Data Results Return

Results will return players sorted by the difference between salary and expected salary. Players with the largest negative difference will be listed first. Data table will provide player name, statistic being used, salary, and expected salary.

# 3 LinearRegression.php

LinearRegression.php holds the definition of the LinearRegression class.

## 3.1 Variables

Variables for this class are set to private and are accessible through setter and getter functions and are declared private.

### 3.1.1 meanX

The variable meanX is used to hold the value of the mean or mathematical average of the set of values of the set X.

### 3.1.2 stddevX

The variable stddevX is used to hold the value of the standard deviation of the values in the set X.

### 3.1.3 sumX

The variable sumX is holds the value of the sum of all the numbers in the set X.

### 3.1.4 sumXsqr

The variable sumXSqr holds the sum of the squared difference between the mean and the value X.

### 3.1.5 meanY

The variable meanY is used to hold the value of the mean or mathematical average of the set of values of the set Y.

### 3.1.6 stddevY

The variable stddevY is used to hold the value of the standard deviation of the values in the set Y.

### 3.1.7 sumY

The variable sumY is holds the value of the sum of all the numbers in the set Y.

### 3.1.8 sumYsqr

The variable sumYSqr holds the sum of the squared difference between the mean and the value Y.

3.1.9 psum

The variable psum holds the sum of the players in the dataset.

3.1.10 n

The variable n holds the count.

3.1.11 slope

The variable slope holds the slope of the linear regression line for the slope intercept equation

3.1.12 intercept

The variable intercept holds the Y intercept of the slope intercept equation for the linear regression function.

## 3.2 Class Functions

### 3.2.1 getMeanX()

#### Summary

A getter function for the variable meanX

#### Definition

public function getMeanX()

#### Parameters

None

#### Return Value

The return value is that of meanX, and is dependent upon it.

### 3.2.2 getStdDevX()

#### Summary

A getter function for the variable stddevX

#### Definition

public function getStdDevX()

#### Parameters

None

#### Return Value

The return value is that of stddevX, and is dependent upon it.

### 3.2.3 getSumX()

#### Summary

A getter function for the variable sumX

#### Definition

public function getSumX()

#### Parameters

*None*

#### Return Value

The return value is that of sumX, and is dependent upon it.

### 3.2.4 getSumXSquared()

#### Summary

A getter function for the variable sumXsq

#### Definition

public function getSumXSquared()

#### Parameters

None

#### Return Value

The return value is that of sumXsq, and is dependent upon it.

### 3.2.5 getSumXSquared()

#### Summary

A getter function for the variable sumXsq

#### Definition

public function getSumXSquared()

#### Parameters

None

#### Return Value

The return value is that of sumXsq, and is dependent upon it.

### 3.2.6 getMeanY()

#### Summary

A getter function for the variable meanY

#### Definition

public function getMeanY()

#### Parameters

None

#### Return Value

The return value is that of meanY, and is dependent upon it.

### 3.2.7 getStdDevY()

#### Summary

A getter function for the variable stddevY

#### Definition

public function getStdDevY()

#### Parameters

None

#### Return Value

The return value is that of stddevY, and is dependent upon it.

### 3.2.8 getSumX()

#### Summary

A getter function for the variable sumY

#### Definition

public function getSumY()

#### Parameters

*None*

#### Return Value

The return value is that of sumY, and is dependent upon it.

### 3.2.9 getSumXSquared()

#### Summary

A getter function for the variable sumYsq

#### Definition

public function getSumYSquared()

#### Parameters

*None*

#### Return Value

The return value is that of sumYsq, and is dependent upon it.

### 3.2.10 getSumYSquared()

#### Summary

A getter function for the variable sumYsq

#### Definition

public function getSumYSquared()

#### Parameters

*None*

#### Return Value

The return value is that of sumYsq, and is dependent upon it.

### 3.2.11 getPsum()

#### Summary

A getter function for the variable psum.

#### Definition

public function getPsum()

#### Parameters

*None*

#### Return Value

The return value is that of psum, and is dependent upon it.

### 3.2.12 getCount()

#### Summary

A getter function for the variable n.

#### Definition

public function getCount()

#### Parameters

*None*

#### Return Value

The return value is that of n, and is dependent upon it.

### 3.2.13 getMeanX()

#### Summary

A getter function for the variable meanX

#### Definition

public function getMeanX()

#### Parameters

*None*

#### Return Value

The return value is that of meanX, and is dependent upon it.

### 3.2.14 getStdDevX()

#### Summary

A getter function for the variable stddevX

#### Definition

public function getStdDevX()

#### Parameters

*None*

#### Return Value

The return value is that of stddevX, and is dependent upon it.

### 3.2.15 getSumX()

#### Summary

A getter function for the variable sumX

#### Definition

public function getSumX()

#### Parameters

*None*

#### Return Value

The return value is that of sumX, and is dependent upon it.

### 3.2.16 getSumXSquared()

#### Summary

A getter function for the variable sumXsq

#### Definition

public function getSumXSquared()

#### Parameters

*None*

#### Return Value

The return value is that of sumXsq, and is dependent upon it.

### 3.2.16 getMeanY()

#### Summary

A getter function for the variable meanY

#### Definition

public function getMeanY()

#### Parameters

*None*

#### Return Value

The return value is that of meanY, and is dependent upon it.

### 3.2.17 getStdDevY()

#### Summary

A getter function for the variable stddevY

#### Definition

public function getStdDevY()

#### Parameters

*None*

#### Return Value

The return value is that of stddevY, and is dependent upon it.

### 3.2.18 getSumY()

#### Summary

A getter function for the variable sumY

#### Definition

public function getSumY()

#### Parameters

*None*

#### Return Value

The return value is that of sumY, and is dependent upon it.

### 3.2.19 getSumYSquared()

#### Summary

A getter function for the variable sumYsq

#### Definition

public function getSumYSquared()

#### Parameters

*None*

#### Return Value

The return value is that of sumYsq, and is dependent upon it.

### 3.2.20 getPsum()

#### Summary

A getter function for the variable psum.

#### Definition

public function getPsum()

#### Parameters

*None*

#### Return Value

The return value is that of psum, and is dependent upon it.

### 3.2.21 getCount()

#### Summary

A getter function for the variable n.

#### Definition

public function getCount()

#### Parameters

*None*

#### Return Value

The return value is that of n, and is dependent upon it.

### 3.2.22 setMeanX()

#### Summary

A setter function for the variable meanX

#### Definition

public function setMeanX()

#### Parameters

None

#### Return Value

None

### 3.2.23 setStdDevX()

#### Summary

A setter function for the variable stddevX

#### Definition

public function setStdDevX()

#### Parameters

*None*

#### Return Value

None

### 3.2.24 setSumX()

#### Summary

A getter function for the variable sumX

#### Definition

public function setSumX()

#### Parameters

None

#### Return Value

None

### 3.2.25 setSumXSquared()

#### Summary

A setter function for the variable sumXsq

#### Definition

public function setSumXSquared()

#### Parameters

None

#### Return Value

None

### 3.2.26 setMeanY()

#### Summary

A setter function for the variable meanY

#### Definition

public function setMeanY()

#### Parameters

None

#### Return Value

None

### 3.2.27 setStdDevY()

#### Summary

A setter function for the variable stddevY

#### Definition

public function setStdDevY()

#### Parameters

None

#### Return Value

None

### 3.2.29 setSumY()

#### Summary

A setter function for the variable sumY

#### Definition

public function setSumY()

#### Parameters

None

#### Return Value

None

### 3.2.30 setSumYSquared()

#### Summary

A setter function for the variable sumYsq

#### Definition

public function setSumYSquared()

#### Parameters

None

#### Return Value

None

### 3.2.31 setPsum()

#### Summary

A setter function for the variable psum.

#### Definition

public function setPsum()

#### Parameters

None

#### Return Value

None

### 3.2.32 setCount()

#### Summary

A setter function for the variable n.

#### Definition

public function setCount()

#### Parameters

None

#### Return Value

None

### 3.2.33 getCorrelationCoefficient()

#### Summary

A getter function for the variable r. The correlation coefficient is calculated within this function.

#### Definition

public function getCorrelationCoefficient()

#### Parameters

*None*

#### Return Value

Returns the value of the correlation coefficient.

### 3.2.34 getSlope()

#### Summary

A getter function for the variable slope. The value is calculated within this function and represents the slope of the linear regression line.

#### Definition

public function getSlope()

#### Parameters

*None*

#### Return Value

Returns the value of the slope for the linear regression function.

### 3.2.35 getIntercept()

#### Summary

A getter function for the variable r. The correlation coefficient is calculated within this function.

#### Definition

public function getIntercept()

#### Parameters

*None*

#### Return Value

Returns the value of the intercept for the Linear regression function

### 3.2.36 getY()

#### Summary

A function to calculate the Y value in the regression function for any value of X input based on the slope and intercept.

#### Definition

public function getY()

#### Parameters

*x*

#### Return Value

None

### 3.2.37 toString()

#### Summary

This function provides a method to output an element

#### Definition

public function toString()

#### Parameters

None

#### Return Value

A String representing the data in an element

# 4 LinearRegressionConstants.php

LinearRegressionConstants.php holds the definition of the LinearRegressionConstants, and PlayerDataPointConstants classes.

## 4.1 class LinearRegressionConstants

The LinearRegressionConstants.php holds the definition of the LinearRegression class.

#### Summary

This class will hold the calculated regression variables for analysis and selection of players

#### Definition

public class LinearRegressionConstants()

#### Parameters

*None*

#### Return Value

None

## 4.2 class PlayerDataPointConstants

The LinearRegressionConstants.php holds the definition of the PlayerDataPointConstants class.

#### Summary

This class will hold information related to the player

#### Definition

public class LinearRegressionConstants()

#### Parameters

*None*

#### Return Value

None

# 5 LinearRegressionCriteria.php

LinearRegressionCriteria.php is a utility class for building SQL statements for harvesting LinearRegression records/objects.

## 5.1 Variables

Variables for this class are set to private

### 5.1.1 tableX

The variable tableX is used to store the name of the desired table X, used for joining

### 5.1.2 tableY

The variable tableY is used to store the name of the desired table Y

### 5.1.3 rowX

The variable rowX is used to store the rowX value

### 5.1.4 rowY

The variable rowY is used to store the rowY value

### 5.1.5 jTables

The array jTables holds tables that need to be added to the query from joins

### 5.1.6 joinTablesSql

The variable holds sql to add the tables to the final query

### 5.1.7 joinTablesSql

The variable holds sql for joining tables on the columns given

### 5.1.7 playerJoinTables

The array holds tables that need to be added to the query from joins for selecting player data

### 5.1.8 playerJoinTablesSql

The variable holds sql to add the tables to the final query for player data

### 5.1.8 comparisonSql

The variable holds sql to add to both the linear regression and player data based on search criteria

## 5.2 Class Functions

### 5.2.1 \_construct()

#### Summary

The \_construct function is a non-default constructor for the class which has 4parameters and serves to set up the variable definitions.

#### Definition

public function getMeanX($tableX, $rowX, $tableY, $rowY)

#### Parameters

$tableX, $rowX, $tableY, $rowY

#### Return Value

None

### 5.2.1 addJoin ()

#### Summary

The addJoin function is used to create an inner join statement to build the sql statement for obtaining the proper data from the database. The statement updates the join tables array and add join table sql.

#### Definition

public function adjoin($tableLeft, $columnLeft, $tableRight, $columnRight)

#### Parameters

$tableLeft, $columnLeft, $tableRight, $columnRight

#### Return Value

None

5.2.2 addComparison()

#### Summary

The addComparison function is used to add search criteria to the sql based on a comparison.

#### Definition

public function addComparison ($table, $column, $value, $comparison)

#### Parameters

$table, $column, $value, $comparison

#### Return Value

None

### 5.2.3 getLinearRegressionSql()

#### Summary

The getLinearRegressionSql function is used to create the sql statement

#### Definition

public function getLinearRegressionSql()

#### Parameters

None

#### Return Value

A string representing the SQL statement for the linear regression function

### 5.2.4 getPlayerDataPointSql()

#### Summary

The getPlayerDataPointSql function is a utility function to build an SQL statement to gather records for PlayerDataPoints

#### Definition

public function getPlayerDataPointSql()

#### Parameters

None

#### Return Value

A string representing the SQL statement for player data points.

### 5.2.4 getTableAlias()

#### Summary

The getTableAlias function checks to see if we’re using the given table name as our x or y table and returns the corresponding alias or the table name.

#### Definition

Private function getTableAlias($tableName)

#### Parameters

$tableName

#### Return Value

A string representing the alias or table name

### 5.2.5 updateJoinTables ()

#### Summary

The updateJoinTables function adds to the joinTables arrays if there is no table or corresponding alias

#### Definition

Private function updateJoinTables ($tableName)

#### Parameters

$tableName

#### Return Value

None

# 6 LinearRegressionUtils.php

LinearRegressionUtils.php has several utility functions related to linear regression.

## 6.1 rowToLinearRegression

#### Summary

This function will take the result row that utilizes the constant names and builds a LinearRegression object from it.

#### Definition

public class LinearRegressionConstants($row)

#### Parameters

*row*

#### Return Value

This function returns a LinearRegression object.

## 6.2 rowsToPlayerDataPoints

#### Summary

This function takes rows that utilize the constant names and build an array of PlayerDataPoint Objects from them.

#### Definition

public class rowsToPlayerDataPoints($result, $linearRegression)

#### Parameters

*result, linearRegression*

#### Return Value

This function returns an array of PlayerDataPoint objects

# 7 PlayerDataPoint.php

PlayerDataPoint.php contains the definition for the PlayerDataPoint class

## 7.1 Variables

Variables for this class are set to private

### 7.1.1 playerId

The variable playerId holds the playerId that will be obtained from the database.

### 7.1.2 firstName

The variable firstName holds the first name of the player matching the playerId obtained from the database.

### 7.1.3 lastName

The variable lastName holds the last name of the player matching the playerId obtained from the database.

### 7.1.4 x

The variable x holds the x value for the player

### 7.1.5 y

The variable y holds the y value for the player

### 7.1.3 expectedX

The variable expectedX holds the expected X value for the player in the database

### 7.1.4 expectedY

The variable expectedY holds the expected Y value for the player in the database

## 7.2 Class Functions

### 7.2.1 getPlayerId

#### Summary

This is a getter function to return the playerID in the object.

#### Definition

public class getPlayerId()

#### Parameters

*None*

#### Return Value

This function returns the playerId from the object.

### 7.2.2 getFirstName

#### Summary

This is a getter function to return the firstName variable in the object.

#### Definition

public class getFirstName()

#### Parameters

*None*

#### Return Value

This function returns the firstName variable from the object.

### 7.2.3 getLastName

#### Summary

This is a getter function to return the lastName variable in the object.

#### Definition

public class getLastName()

#### Parameters

*None*

#### Return Value

This function returns the lastName variable from the object.

### 7.2.4 getXValue

#### Summary

This is a getter function to return the x variable from the object.

#### Definition

public class getXValue()

#### Parameters

*None*

#### Return Value

This function returns the x variable from the object.

### 7.2.5 getYValue

#### Summary

This is a getter function to return the y variable from the object.

#### Definition

public class getYValue()

#### Parameters

*None*

#### Return Value

This function returns the y variable from the object.

### 7.2.6 getExpectedX

#### Summary

This is a getter function to return the x variable from the object.

#### Definition

public class getXExpectedX()

#### Parameters

*None*

#### Return Value

This function returns the expectedX variable from the object.

### 7.2.7 getExpectedY

#### Summary

This is a getter function to return the y variable from the object.

#### Definition

public class getExpectedY()

#### Parameters

*None*

#### Return Value

This function returns the expectedY variable from the object.

### 7.2.8 getDifferenceX

#### Summary

This is a getter function to return the difference between the x variable and the expected x variable from the object.

#### Definition

public class getDifferenceX()

#### Parameters

*None*

#### Return Value

This function returns the difference between the x variable and the expectedX variable.

### 7.2.9 getDifferenceY

#### Summary

This is a getter function to return the difference between the y variable and the expectedY variable from the object.

#### Definition

public class getDifferenceY()

#### Parameters

*None*

#### Return Value

This function returns the difference between the y variable and the expectedY variable.

### 7.2.10 setPlayerId

#### Summary

This is a setter function for the playerId variable

#### Definition

public class setPlayerId($id)

#### Parameters

*id*

#### Return Value

None

### 7.2.11 setFirstName

#### Summary

This is a setter function for the firstName variable in the object.

#### Definition

public class setFirstName($firstName)

#### Parameters

*firstName*

#### Return Value

None

### 7.2.12 setLastName

#### Summary

This is a setter function for the lastName variable in the object.

#### Definition

public class setLastName($lastName)

#### Parameters

*lastName*

#### Return Value

None

### 7.2.13 setXValue

#### Summary

This is a setter function for the x variable in the object.

#### Definition

public class setXValue($x)

#### Parameters

*x*

#### Return Value

None

### 7.2.14 setYValue

#### Summary

This is a setter function for the y variable in the object.

#### Definition

public class getYValue($y)

#### Parameters

*y*

#### Return Value

None

### 7.2.15 setExpectedX

#### Summary

This is a setter function for the expectedX variable in the object.

#### Definition

public class setXExpectedX($expectedX)

#### Parameters

*expectedX*

#### Return Value

None

### 7.2.16 setExpectedY

#### Summary

This is a setter function for the expectedY variable in the object.

#### Definition

public class getExpectedY($expectedY)

#### Parameters

None

#### Return Value

None