**PlayerData**

**com.ipl.training.induction.draughts.controller.Draughts**

**Class Specification**

**Project Reference 999/170**

**Document Reference PlayerData\_cs**

**2 December 2011**

**Issue 2**

**Copyright Notice**

This document has been prepared by IPL Information Processing Limited (“IPL”) and, subject to any existing rights of other parties, IPL is the owner of the copyright of this document. No part of this document may be copied, reproduced, stored in a retrieval system, disclosed to a third party or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of IPL.

**© IPL Information Processing Limited 2011**

**Document History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Issue | Author | Comments |
| 10/06/2010 | 1 | R. Miskin |  |
| 02/12/2011 | 2 | Phil Lewis, C Harrison | Updated to reflect new Java standard, including diagrams. |

Contents

[1 Class Identity 1](#_Toc304891301)

[2 Description 1](#_Toc304891302)

[3 Interface 1](#_Toc304891303)

[4 Structure 1](#_Toc304891304)

[5 Element Descriptions and Interfaces 1](#_Toc304891305)

[5.1 Public Methods 1](#_Toc304891306)

[5.1.1 PlayerData 1](#_Toc304891307)

[5.1.2 Equals 2](#_Toc304891308)

[5.1.3 getName 3](#_Toc304891309)

[5.1.4 getType 3](#_Toc304891310)

[5.1.5 hashCode 4](#_Toc304891311)

[5.1.6 toString 4](#_Toc304891312)

[5.2 Package Access Methods 5](#_Toc304891313)

[5.3 Private Methods 5](#_Toc304891314)

[5.4 Public Fields 5](#_Toc304891315)

[5.5 Package Access Fields 5](#_Toc304891316)

[5.6 Private Fields 5](#_Toc304891317)

[5.7 Public Nested Classes 5](#_Toc304891318)

[5.7.1 Enumeration PlayerData.PlayerType 5](#_Toc304891319)

[5.8 Package Access Nested Classes 7](#_Toc304891320)

[5.9 Private Nested Classes 7](#_Toc304891321)

[6 Resource Requirements 7](#_Toc304891322)

[7 Test Plan 7](#_Toc304891323)

[8 Scenarios 7](#_Toc304891324)

[9 Supporting Documentation 7](#_Toc304891325)

# Class Identity

Class Name - PlayerData  
Class Identity - com.ipl.training.induction.draughts.controller.PlayerData

# Description

The PlayerData class is used to store the name and the type of a player.

# Interface

|  |  |
| --- | --- |
| Attribute | Value |
| Visibility | Public |
| Modifiers | Final |
| Extends | - |
| Implements | - |

# Structure



# Element Descriptions and Interfaces

## Public Methods

### PlayerData

Constructor to create a PlayerData object.

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | - |
| Throws | IllegalArgumentException |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| name | String | The name of this player |
| type | PlayerType | The type of this player |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| - |  |

**Processing Logic**

* If name or type are null throw an IllegalArgumentException.
* Otherwise store name in this.name and type in this.type.

### Equals

This methods overrides Object.equals(). It is essential that it obeys the general contract for java.lang.Object.equals() as described in the javadoc.

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | Object.equals() |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| obj | Object | The Object to use in the comparison |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| boolean | True if obj is a PlayerData with the same name and type as this PlayerData. Otherwise false. |

This method defines the requirements that need to be met in order for two PlayerData objects to be considered equal.

**Processing Logic**

* If obj is null return false
* If obj is not the same class as this return false
* If obj.type does not match this.type return false
* If obj.name does not match this.name return false
* Return true.

### getName

Return the name of this player.

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | - |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| - | - | - |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| String | The name of this player |

**Processing Logic**

* Return name

### getType

Return the type of this player (HUMAN/COMPUTER/REMOTE).

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | - |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| - | - | - |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| PlayerType | The type of this player. |

**Processing logic**

* Return type

### hashCode

This methods overrides Object.hashCode(). It is essential that it obeys the general contract for java.lang.Object.hashCode() as described in the javadoc.

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | Object.hashCode() |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| - | - | - |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| int | A hash value for this PlayerData. |

This method returns a hash code value for the PlayerData. It is essential that whenever the method is invoked more than once during the execution of the Java application on the same object, then the same hash code value must be returned each time. Also, two PlayerData objects that are considered equal according to the Equals method must return the same hash code value.

**Processing Logic**

* Return name.hashCode() + (31\*type.hashCode()).

### toString

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | Object.toString() |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| - | - | - |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| String | A representation of this PlayerData. |

**Processing Logic**

* Return a string comprising of name and type e.g. “Jack Computer” or “Jill Remote”.

## Package Access Methods

None

## Private Methods

None

## Public Fields

None

## Package Access Fields

None

## Private Fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Attributes | Value | Description |
| name | String | Final | - | The name of this player. |
| type | PlayerType | final | - | The type of this player e.g. Human or Computer. |

## Public Nested Classes

### Enumeration PlayerData.PlayerType

PlayerType is a public enumeration type used within PlayerData to specify the type of player e.g. HUMAN, COMPUTER, REMOTE.

#### Public Methods

#### toString

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | final |
| Overrides | Object.toString() |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| - | - | - |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| String | A representation of this PlayerType. |

**Processing Logic**

Return displayName.

#### Package Access Methods

None

#### Private Methods

#### PlayerType

Constructor to create a PlayerType object.

|  |  |
| --- | --- |
| Attribute | Value |
| Modifiers | - |
| Overrides | - |
| Throws | - |

**Inputs**

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| displayName | String | The string representation of this enumeration. |

**Outputs**

|  |  |
| --- | --- |
| Type | Description |
| - |  |

**Processing Logic**

Set this.displayName to displayName.

#### Public Fields

|  |  |
| --- | --- |
| Name | Description |
| COMPUTER | Indicates the computer will play this side. String representation will be “Computer”. |
| HUMAN | Indicates a human will play this side. String representation will be “Human”. |
| REMOTE | Indicates a remote player will play this side. String representation will be “Remote”. |

#### Package Access Fields

None

#### Private Fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Attributes | Value | Description |
| displayName | String | Final | - | The string representation of this PlayerType. |

## Package Access Nested Classes

None

## Private Nested Classes

None

# Resource Requirements

None

# Test Plan

* 1. Construct several PlayerData objects and check that the getName and getType methods return the expected values.
  2. Check that IllegalArgumentExceptions are thrown when a PlayerData is constructed with either parameter as null.
  3. Construct several PlayerData objects and check that the toString method returns the expected values.
  4. Construct several PlayerData objects and check that the equals and hashCode methods behave correctly.
  5. Use TestHelper.testGeneratedEnumMethods() to get code coverage on the methods on PlayerData.PlayerType.

# Scenarios

None

# Supporting Documentation

None

**- End of Document -**