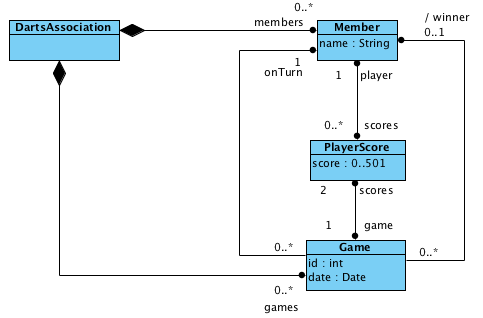


A Dart Association keeps data of members and of their games.

games are using 501 dart rules (<https://www.mastersofgames.com/rules/darts-rules.htm>). An example of the scores during a game is shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Player 1 | | Player 2 | |
|  | Turn score | Player score | Turn score | Player score |
|  |  | 501 |  | 501 |
| 1 | 140 | 361 | 98 | 403 |
| 2 | 180 | 181 | 180 | 223 |
| 3 | 100 | 81 | 180 | 43 |
| 4 | 71 | 10 | 40 | 3 |
| 5 | 6 | 4 | 21 | 3 |
| 6 | 4 | (winner) 0 |  |  |

The following design has been made (note: UML dot notation is used for associations /ownership in UML 2.1). The dot can be interpreted as an arrow (known from UML 1)



General requirements:

Member is responsible of keeping track of the name and number of members. It also manages all Playerscores of the member.

PlayerScore is responsible of keeping the current player score of a game. When a game is finished, it therefore contains the final score of a player.

Game is responsible of keeping track of who’s turn it is to play and the player scores during a game. When a game is finished, the winning member is also added.

DartsAssociation is responsible of the administration of members and games (CRUD). During a game, the turn score of a player is entered here.

Specific requirements:

A player score can be added to a member

a member can return a set of player scores

a member can return his name.

a game is played on a given, unchangeable date.

A date for a game is set when the game is created

A game keeps track of which member has the turn to play.

The turn changes automatically after a score has been entered.

A winner will be set automatically when the game is finished. In this case, the turn is not changed to the next player.

The turn score is only added when the game is not finished.

the current player can be returned.

The winner can be returned.

The game can return if a game is finished or not.

The game can return the game score, consisting of the player score of player 1 and player 2, separated by a ‘- ‘

The player score is [0..501]

the member of a player score can be returned.

The game of a player score can be returned

The Dartsassociation:

can add a member by name.

A member has a unique name

* it is possible to add a new member with a unique name
  + compare Member object to add with all current members.
  + if it's unique (by name): it should be added
* throw xxxx exception when adding member with non-unique name:
  + compare Member object to add with all current members.
  + create member object
  + if it's not unique (by name): it should NOT be added + exception xxx is thrown. (=assumption)
* throw illegalArgumentException when adding a null object

can change a member name to a new name.

can delete a member by name.

can create a game between two members

a game always has a unique id

can add the turn score of a game

can find a game by id.

Can return the current score of a game

NonUniqueNameException is thrown when using non unique names

IncorrectIdException is thrown when using incorrect id.

Define the tests for the constructors of the classes, based on the design. Start with classes which are furthest away from the encompassing DartsAssociation.

Define tests for the behaviour of every class (again start with the simple classes first)