**YatzyWeb**

Final Project

Java EE & XML

Tarleena Marttila

Markus Paappanen

Ohjelmistotekniikan koulutusohjelma

Tekniikan ja liikenteen ala

Contents

[1 Project subject 2](#_Toc406187864)

[1.1 Requirements 2](#_Toc406187865)

[1.2 Tecnologies 2](#_Toc406187866)

[2 Application structure 2](#_Toc406187867)

[3 Classes and tecnologies 3](#_Toc406187868)

[3.1 XMLController.java 3](#_Toc406187869)

[3.1.1 private String[ ][ ] scoreplayers 4](#_Toc406187870)

[3.1.2 public void checkIfHighScore(String name, String score) 4](#_Toc406187871)

[3.1.3 public void readHighScoresFile() 4](#_Toc406187872)

[3.1.4 public void createHighScoresFile() 4](#_Toc406187873)

[3.1.5 public void readBackupFile() 4](#_Toc406187874)

[3.1.6 public void createBackupFile() 4](#_Toc406187875)

[3.1.7 About used techniques 5](#_Toc406187876)

[3.2 LobbyPlayer.java 5](#_Toc406187877)

[3.3 LobbyPlayers.java 5](#_Toc406187878)

[3.4 UIController.java 5](#_Toc406187879)

[3.5 Dice.java 5](#_Toc406187880)

[3.6 Player.java 5](#_Toc406187881)

[3.7 Game.java 5](#_Toc406187882)

[3.8 GameList.java 6](#_Toc406187883)

[3.9 GameController.java 6](#_Toc406187884)

[3.10 JavaServerFaces Pages 6](#_Toc406187885)

[4 Yatzyweb06 6](#_Toc406187886)

[4.1 YatzyWeb06 vs. requirements 6](#_Toc406187887)

[4.2 Challenges 7](#_Toc406187888)

# Project subject

YatzyWeb is final project for two separated courses: JavaEE and XML courses. YatzyWeb is a game that uses XML as a data source.

## Requirements

There were some requirements set by us and set by assigment:

* WebUI
* XML data source
* Two people can play the game
* Player can continue game later
* Best scores will be saved
* Players see the best scores

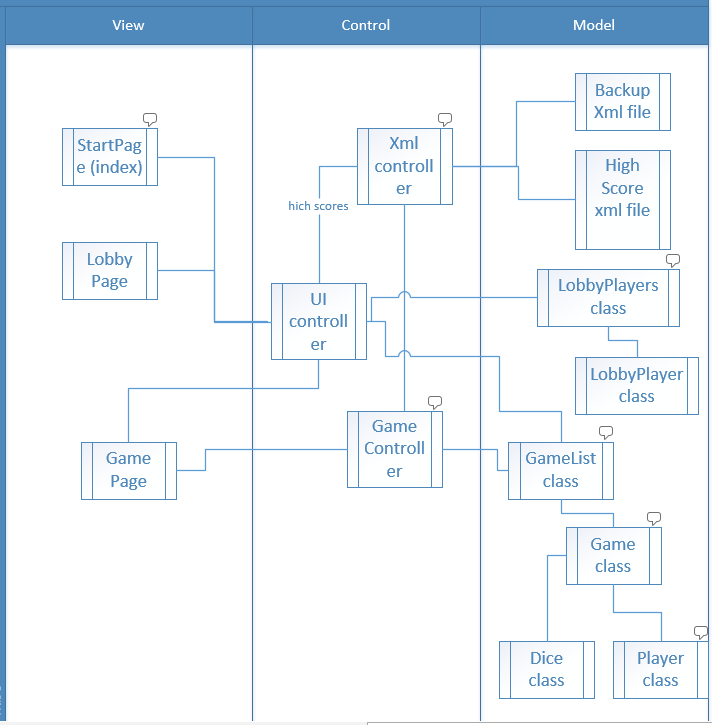
## Tecnologies

YatzyWeb uses MVC-model and JSF-framework. Data source connection was done using DOM specifications.

# Application structure

Picture 1. Site map.

Application has three xhtml pages (picture 1): index, Lobby and Yatzy. Index is the first page where player gives his/her name and then moves to Lobby. In lobby player sees all the available player and Highshores. There player can challenge another player and start a game and then player is directed to Yatzy page. Yatzy page handles showing the scores and dices. Project was design to use MVC model (picture 2).

Picture 2. Application structure presented via MVC model.

# Classes and tecnologies

The classes and the most important methods and properties are introduced in this character. There are also some explanations for chose technologies: comparing the other techniques possible to use.

## XMLController.java

XMLController.java is a controller class which is responsible for reading and writing XML files. It is *@SessionScoped* and *@ManagedBean* so every user have their own instance of it.

### private String[ ][ ] scoreplayers

XMLController class has a variable *scoreplayers* which is a two dimensional array holding a local copy of the high scores.

### public void checkIfHighScore(String name, String score)

This method checks if new high score has made and modify the local *scoreplayers* array if needed. At first the method updates the *scoreplayers* array with data of XML file by using the *readHighScoresFile()* method. Also calls the *createHighScoresFile()* method if new high score has made.

### public void readHighScoresFile()

The *readHighScoresFile()* is a method which reads an XML file and saves the data to the local variable *scoreplayers.* For handling the XML file the method uses javax.xml.parsers library’s classes DocumentBuilder and DocumentBuilderFactory. By these classes the method parses the XML file to Document object (DOM). The local variable *scoreplayers* is filled with the data of Document object. If there is not a XML file available the method fills the *scoreplayers* with hard coded default data which is introduced in the method.

### public void createHighScoresFile()

The method does everything like the *readHighScoresFile()* method – only opposite order: it builds a Document object from the data of variable *scoreplayers* and forms an XML file by it. For transforming the Document object to an XML file the method uses javax.xml.transform library’s classes Transformer, TransformerFactory and StreamResult and also java.io.File library’s class File.

### public void readBackupFile()

The *readBackupFile()* is for reading an XML file which contains the data of all games. The method uses the same libraries and classes as the *readHighScoresFile()* for reading an XML file and saves data to the managed bean of GameList class.

### public void createBackupFile()

The method does everything like the *readBackupFile()* method – only opposite order: it builds a Document object from the data of the managed bean of GameList class and forms an XML file by it. For transforming the Document object to an XML file the method uses the same libraries and classes as the *createHighScoresFile()* method.

### About used techniques

Some internet sources blame the DOM Parser being not the best option: slow and consuming a lot of memory when it loads an XML document which contains a lot of data.

There is a several libraries available for reading and writing XML files: for example using JAXB annotation to convert Java object to / from XML file. It’s more like binding technique. In this option the XML elements are declared in the class which is performing the data source, in this case Player class. I tried out this option but did not get it to work so I chose another way, the DOM. Probably the problem was my own logical mistake but there was not enough time to figure it out.

## LobbyPlayer.java

LobbyPlayer class handles user name and user game state.

## LobbyPlayers.java

LobbyPlayers is a list of LobbyPlayer classes. This class is applicationscoped so every user can join to same lobby and challenge each other. There are mothods to challenge users and check what stat user is in.

## UIController.java

UIController is sessionscoped managedbean that knows player name and who player challenged. UIController manages LobbyPlayers class and handles the site page flow. UIController also creates your GameController and creates the game to GameList.

## Dice.java

Dice is a simple class that stores roll(1-6) and if the dice is pressed. Dice class also implements Comparable so the score checking would be easier.

## Player.java

Player class handles all data needed to show scores for the player. Player class also knows what round is going.

## Game.java

Game has two lists, player list and dices list. Player list is a collection of all the players in the game and dices list is list of 5 different dices game needs.

## GameList.java

GameList is list of ongoing games. GameList is managedbean and ApplicationScoped and this allows users to share sessions and play against each other. There are methods to save points to specific player score and those methods use simple switch case method to calculate proper points using game dices. There are also methods to get your game object and your player object using your name(unique) stored in gameController and in your player class. GameList also contains some simple checking boolean methods like *isMyTurn* and *isAlsone*.

## GameController.java

GameController handles all the reading and writing to class GameList and classes that it contains. It is SessionScoped so every player have their own GameController. GameController has properties number of rolls and player name. This class also destroys game and player objects after the game finishes.

## JavaServerFaces Pages

The navigation from page to another is carried out by two different ways. Buttons on index.xhtml and Lobby.xhtml call the methods of managed bean class UIController which return next page as string. On the Lobby.xhtml and Yatzy.xhtml there are also a JavaScript function which refresh the page automatic after few seconds. This is not the best way to implement automatic refreshing, but it’s chose because the lack of time. The dilemma is that the user should be directed to another page when property value of managed bean is changed. Or information that is changing may be located somewhere else. There are many ways to implement page refreshing and several different technologies for that.

# Yatzyweb06

## YatzyWeb06 vs. requirements

The current version 06 corresponds to set requirements mainly: it’s a web application for two player that uses XML data source for saving high scores. Only requirement not fulfilled is the option to player to continue game later. There are methods for this in the XMLController, but requirement is not implemented in the yatzy page. You can test these methods in the testpage testisivu.xhtml.

## Challenges

Worst problem for this project was both of us getting sick in the last week of the project. So the game is not as polished as we hoped for.

JavaEE was new to us and it was long time since we coded any java. So we had to do quite a few proof of concepts. There are some test cases still at the project and unnessesarily comments and parts in code.