Tic-Tac-Toe

Kasper Jacobsen

Project Duration (28-05-2020) – ()

Table of Contents

[Project Requirements & specifications 1](#_Toc42485162)

[Must have(s) 2](#_Toc42485163)

[Server Site 2](#_Toc42485164)

[WebSocketServer 2](#_Toc42485165)

[Server\_Models 2](#_Toc42485166)

[Server\_Repository 2](#_Toc42485167)

[Client Site 3](#_Toc42485168)

[Tic-Tac-Toe 3](#_Toc42485169)

[TTT\_Models 3](#_Toc42485170)

[TTT\_Repository 3](#_Toc42485171)

[Time spent on the project 3](#_Toc42485172)

[References and tools used 5](#_Toc42485173)

[Tools 5](#_Toc42485174)

[References 5](#_Toc42485175)

[Inspirational YouTube Videos 5](#_Toc42485176)

[Inspirational Websites 5](#_Toc42485177)

[Inspirational Code 5](#_Toc42485178)

[Others code I have used I my project 6](#_Toc42485179)

[Others design (UI) that I have used 6](#_Toc42485180)

[Brainstorming my ideas with 7](#_Toc42485181)

[Project Help 7](#_Toc42485182)

[What was challenging in this project? 7](#_Toc42485183)

[Rules of the game / How to play 7](#_Toc42485184)

[Installation Guide 7](#_Toc42485185)

[How to install and set up the Tic-Tac-Toe Application. 7](#_Toc42485186)

[Disclaimer: 8](#_Toc42485187)

[Installation 8](#_Toc42485188)

# Project Requirements & specifications

## Must have(s)

* Single and multiplayer option
  + Single player should be controlled from one pc.
    - Play with yourself.
  + Multiplayer should be played from two different pc’s via a connection to a server
    - Players playing in multiplayer mode will have to wait for another player to join.
* **A player loses the game if** the following happens:
  + If a player leaves the website/application he/she will lose the game.
  + The player did not get three in a row.
  + The game ended in a tie.
* You cannot do the following.
  + Change display name while in a game.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Server Site

## WebSocketServer

The WebSocketServer is made to connect two players in a game of Tic-Tac-Toe

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  WPF

## Server\_Models

Contain ALL the models for the WebSocketServer application.

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  .Net Core Class Library

## Server\_Repository

Contain ALL the logic for the WebSocketServer application.

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  .Net Core Class Library

# Client Site

## Tic-Tac-Toe

‘Tic-Tac-Toe’ is a Game with a UI build with WPF.

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  WPF

## TTT\_Models

Contain ALL the models for the ‘Tic-Tac-Toe’ game

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  .Net Core Class Library

## TTT\_Repository

Contain ALL the logic for the application ‘Tic-Tac-Toe’.

**Languages:** C#

**Framework:** .Net Core 3.1

**ORM** *(object-relational mapper)***:**  .Net Core Class Library

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Time spent on the project

* **28/05/2020 - 31/05/2020** - approximately (9 hours and 30 minutes)total
  + Planning,
    - Setting up my documentation structure
    - Planning how I will build the program and what to build it with.
  + Building diagrams
    - Mock-up image
    - Flowchart
    - Class Diagram

All diagrams were made using draw.io: <https://www.draw.io/>

* **31/05/2020** - approximately (33 minutes) total
  + Creating the project types that was planed in the previous step
  + Setting up a GitHub Repository so the project uses version control.
  + Setting up the Project environments
* **31/05/2020 – 01/06/2020** - approximately (8 hour 30 minutes) total
  + Reading Up on some of the WPF functions
  + Making a basic UI
  + Wrote further documentation as a went.
  + Updating my Class Diagram
  + Working on the Single player mode
* **02/06/2020** – approximately (14 hours 30 minutes) total
  + Re-structuring the game application
  + Separating single player and multi-player into separate windows.
  + Building an application launcher. For the single-player and multi-player windows.
  + Making a dynamic ‘gameLogic’ Class that can be used in both   
    a single player and multi-player game.
* **03/06/2020** - approximately (7 Hours 24 Minutes) total.
  + Updating Div. parts of documentation for the project.
  + Finished the Single-player Mode with a piece-by-piece reusable structure.
  + Reading up on what ‘web sockets’ are and how to use them.
* **04/06/2020** - approximately (7 Hours) total.
  + Reading up on WebSocket server
  + Testing WebSocket concepts and looking into what project type I should use.
  + Updating my documentation
  + Setting up a home windows server to host my application
* **05/06/2020** - approximately (13 Hours 25 minutes) total.
  + Reading up on send and return socket functionality.
  + Restructuring the WebSocket Server.
* **06/06/2020** - approximately (13 Hours 8 Minutes) total.
  + Working on Socket communication between the server and multiplayer application.
  + Making modular methods the client can call on the server.
  + Brainstorming with Jens
* **07/06/2020** - approximately (12 Hours) total.
  + Reading some of the last documentation.
  + Updating the class diagrams
  + Updating the flowchart for single-player mode
  + Brainstorming with Jens
  + Debugging the connection to and from the server.
* **08/06/2020** - approximately () total.
  + Working on displaying the client list from the server on the client.

# References and tools used

## Tools

* Draw.io – Used for making all my diagrams.   
  <https://www.draw.io/>
* Visual Studio 2019 – Used for making the whole project   
  <https://visualstudio.microsoft.com/vs/>
* Google Translate – Used for correcting most of my spelling errors  
  <https://translate.google.dk/?hl=da&tab=TT>

## References

Inspirational YouTube Videos:

* <https://www.youtube.com/watch?v=_OUs2kuI_Yo>
* <https://www.youtube.com/watch?v=yq0dSkA1vpM>
* <https://www.youtube.com/watch?v=7CkSJyZb6H0>
* <https://www.youtube.com/watch?v=mnTyiUAHuVk>
* <https://www.youtube.com/watch?v=sYKrMPhl59A>
* <https://www.youtube.com/watch?v=STuWW6pksXs>
* <https://www.youtube.com/watch?v=MiafbSe0Z5Q>
* <https://www.youtube.com/watch?v=KxdOOk6d_I0>
* <https://www.youtube.com/watch?v=2Nt-ZrNP22A>
* <https://www.youtube.com/watch?v=i5OVcTdt_OU>
* <https://www.youtube.com/watch?v=FYLMxrN5c6g>
* <https://www.youtube.com/watch?v=ycVgXe6v1VQ>

Inspirational Websites:

* <https://developer.okta.com/blog/2019/11/21/csharp-websockets-tutorial>
* <https://stackoverflow.com/questions/16280747/sending-message-to-a-specific-connected-users-using-websocket>

Inspirational Code:

* <https://www.codeproject.com/Tips/1235350/Switch-Type-WPF-ToggleButton-RadioButton-On-Off-Co>
* <https://www.youtube.com/watch?v=mnTyiUAHuVk>
* <https://stackoverflow.com/questions/11133947/how-do-i-open-a-second-window-from-the-first-window-in-wpf>
* <https://www.codeproject.com/Questions/80280/Show-Hide-the-Main-window>
* <https://stackoverflow.com/questions/1195554/how-can-i-remove-the-border-of-a-wpf-window-when-using-luna-or-classic>
* <https://stackoverflow.com/questions/7417739/make-wpf-window-draggable-no-matter-what-element-is-clicked>
* <https://www.youtube.com/watch?v=FYLMxrN5c6g>
* <https://stackoverflow.com/questions/13841880/convert-string-array-to-enum-on-the-fly>
* https://stackoverflow.com/a/60206398

Others code I have used I my project:

* **Creator:** AngelSix.   
  **Note:** I have used his basic game structure and a few of his lines of code in the WPF Backend.  
  **Link:** <https://www.youtube.com/watch?v=mnTyiUAHuVk&t=1053s>
* **Creater:** RezKey   
  **Note:** I have used this code to get a general understanding of what ‘web sockets’ are.  
   Some of his code can be found in my server/client setup.   
  **Link:** <https://www.youtube.com/watch?v=KxdOOk6d_I0>
* **Creator:** TokyoMike and UuDdLrLrSs  
  **Note:** I used this code concept to navigate between windows in my WPF aplication  
  **Link:** <https://stackoverflow.com/a/11134367>
* **Creator:** Vineeth P Joseph  
  **Note:** I used this code concept to close the main window in my WPF application   
  **Link:** <https://www.codeproject.com/Questions/80280/Show-Hide-the-Main-window>
* **Creator:** Rachel  
  **Note:** I have used this code to make my WPF application windows draggable   
   from anywhere in a window  
  **Link:** <https://stackoverflow.com/a/7418629>
* **Creator:** Benny Jørgensen  
  **Note:** I have used parts of his code to make my WebSocket connections.   
   In my test environment.  
  **Link:** <https://www.youtube.com/watch?v=FYLMxrN5c6g>
* **Creator:** Liam & dtb  
  **Note:** I have used their solution on how to generate a random alphanumeric string inside my WebSocketServer/ServerLogic  
  **Link:** <https://stackoverflow.com/a/1344242>
* **Creator:** [Joshua](https://stackoverflow.com/users/1104995/joshua)  
  **Note:** I have used this code in my server application to convert a string array to a Enum Array  
  **Link:** <https://stackoverflow.com/a/13842244>
* **Creator:** nrod  
  **Note:** Used for updating my UI from a separate Thread  
  **Link:** <https://stackoverflow.com/a/60206398>

Others design (UI) that I have used:

* <https://www.youtube.com/watch?v=MiafbSe0Z5Q&list=PLG2wob7K3fpcGzyJZPqyH2W9zDFBHuadI&index=12&t=6s> (No code was taken from this project, just the UI design idea)

Brainstorming my ideas with: (No code was given by these people.)

* Emil Raj Schmidt - [emil4746@elevcampus.dk](mailto:emil4746@elevcampus.dk)
* Jens Nissen - [jens047d@elevcampus.dk](mailto:jens047d@elevcampus.dk)

Project Help:

* Tommaso Briguglio - [tomm5517@elevcampus.dk](mailto:tomm5517@elevcampus.dk)   
  Daniel Krog Debel - [dani894r@elevcampus.dk](mailto:dani894r@elevcampus.dk)   
    
  Both helped out with quickly setting up a windows server in my home.   
  All the help included:   
  1. Transforming and transporting an older pc to my home.  
  2. Setting the pc up as a windows server (With a UI)  
  3. NAT & PORT forwarding on my home network.  
  4. Remote management setup.

### What was challenging in this project?

* **Design and backend functionality compatibility with the server.**   
  *How the UI should work with both single and multiplayer mode.*
* **Communication between players and server.**   
  *Using Web Sockets*
* **The deadline**   
  *Too little time to work with many previously unknown concepts.   
  Like WebSockets, multiplayer and games in general.*

# Rules of the game / How to play

The object of Tic Tac Toe is to get three in a row. You play on a three by three game board. The first player is known as X and the second is O. Players alternate placing X’s and O’s on the game board until either opponent has three in a row or all nine squares are filled. If all nine squares are filled, the game is a tie.

# Installation Guide

## How to install and set up the Tic-Tac-Toe Application.

### Disclaimer:

**This application is made in WPF .Net Core *and works on Windows systems only.***

The minimum system requirements for this application is:

* Windows 10   
  *or any windows system that sports and have .NET Core 3.1 and .NET Framework 4.7*
* A stable internet connection.

### Installation

1. Unzip the file called Student\_Asignment\_Intro\_2020.zip.
2. Open the folder Student\_Asignment\_Intro\_2020
3. Locate and DoubleClick on the Tic-Tac-Toe.exe To run the game.
4. Have Fun.

**This next part is for Multiplayer mode only.**

For this project I have provided a server where the server application if running on.

To connect to the game server, you will have to do the following.

1. Enter 93.160.9.167 into the Ip address text field.
2. Enter 45698 into the Port number text field.
3. Press the “Connect” button

Alternatively, you could enter this to run both the server and the client on your local machine.

1. Enter 127.0.0.1 into the Ip address text field.
2. Enter 8888 into the Port number text field.
3. Press the “Connect” button

This way you could play against a friend on your local machine, with both of you getting your own window to play in.

And because you can type in the address and port of the game server. You also have the opportunity to change server as you please.