Software Design Document

for

Game of Checkers

Version 1.0 approved

Prepared by Natalie Ownby, Anna Cowsar, Cassandra Coyle, Niraj Regmee, and Alexander Rodriguez

Texas State University, CS 3398

September 10, 2018

Table of Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Cassandra Coyle	Initial documentation.	9/23/18
1.1	Alexander R.	Second documentation	10/3/18
1.2	Anna Cowsar, Cassie Coyle, Alexander R	Definitions, Sequential Diagram	10/3/18
1.3	Alexander R.	State Diagram, component functions, High Lvl De.	10/16/18
1.4	Alexander R.	Section 3.1 / 3.2	10/23/18
1.5	Alexander R.	Revisions to 2.1/2.2	10/29/18
1.6	Anna Cowsar	Document Revisions	10/29/18

1. Introduction

1.1 Purpose

The purpose of the System Design Document (SDD) is to document the results of the system design process. This document completely describes the system at the architecture level, including subsystems and their services. It includes hardware mapping, data management, access control, global software control structure, and any boundary conditions.

1.2 System Overview

This document contains a complete description of the Checkers application. For the application the basic architecture will consist of a web page that will pop up and allow the user to play a game of chess against a CPU. The page will include HTML and Java Script code. The Java Script is client-side Java Script, so no node express.

1.3 Definitions, Acronyms and Abbreviations

King: When a player reaches the “king” row (the last row on the opposite end of the board from the player), the player will be kinged. This means that they may now move in a forward position diagonal and backwards diagonal instead of the normal forwards diagonal for regular non-kinged pieces.

Crowning: When a player has finally reached the last row with a piece, they become “crowned” or “kinged.”

Jump: A player may jump a piece if there is a piece in the left or right diagonal and an empty spot in the next place in the diagonal. They must apply the rules to whether it is a king or regular piece. A regular piece may only jump in the forward left or right diagonal direction. A king may jump in a forward position left or right or backwards left or right diagonal.

Checkers Board: A 8x8 plane which includes 24 pieces in total with 12 on each side.

Player: The User and the AI are the ones that move the pieces on the board.

A.I: Is an algorithm that will be playing against the User which is the player.

Checkers piece: A round piece either black or white then switch to king if case is met.

1.4 Supporting Materials

Rules of Checkers

- http://www.indepthinfo.com/checkers/

1.5 Document Overview

First Section contains the overall outline of how the application will be structured and definitions.

Second Section contains the architecture and the basis of how the design will be implemented also describes the elements.

Third Section contains the high-level design of the application.

2. Architecture

2.1 Overview

The checkers game utilizes the Document Object Model (DOM) which is an application programming interface (API) for HTML documents. It defines the logical structure of the documents and the way a document is accessed and manipulated. It provides a JavaScript mapping of all the elements on the page and provides a set of methods for accessing the elements, their attributes, and their contents. JavaScript tools and libraries shall be used.

The checkers game shall be played through an Internet connection due to the application being pulled up through HTML and CSS. In order to play the game, a player must have a web browser that supports JS, CSS, and HTML, for example Edge or Chrome. The user also needs a mouse or touch-screen that allows them to select and move their chosen pieces on the game board and interact with other user interface components.

The checkers game is client-side JavaScript meaning that it is not a server-based application. Therefore, it only requires an interconnection between the web browser and the HTML and JavaScript files. Any changes made in the HTML and JavaScript files need to be saved and then the changes will be reflected in the web browser after simply refreshing the page. There is no communication security, encryption issues, data transfer rates, or synchronization mechanism to worry about.

2.2 Components

2.2.1 Function Draw

This function draws the checkers board on to the web page with the 12 checker pieces on each side of the board.

2.2.2 Function Welcome

This function display text welcoming the user to the game application.

2.2.3 Function Time

This function is started once the start time button is pressed. It keeps track of how much time has passed since the start of the game.

2.2.4 Function Name

The purpose of this function is for the user to enter the name they want to go by while playing the game of checkers.

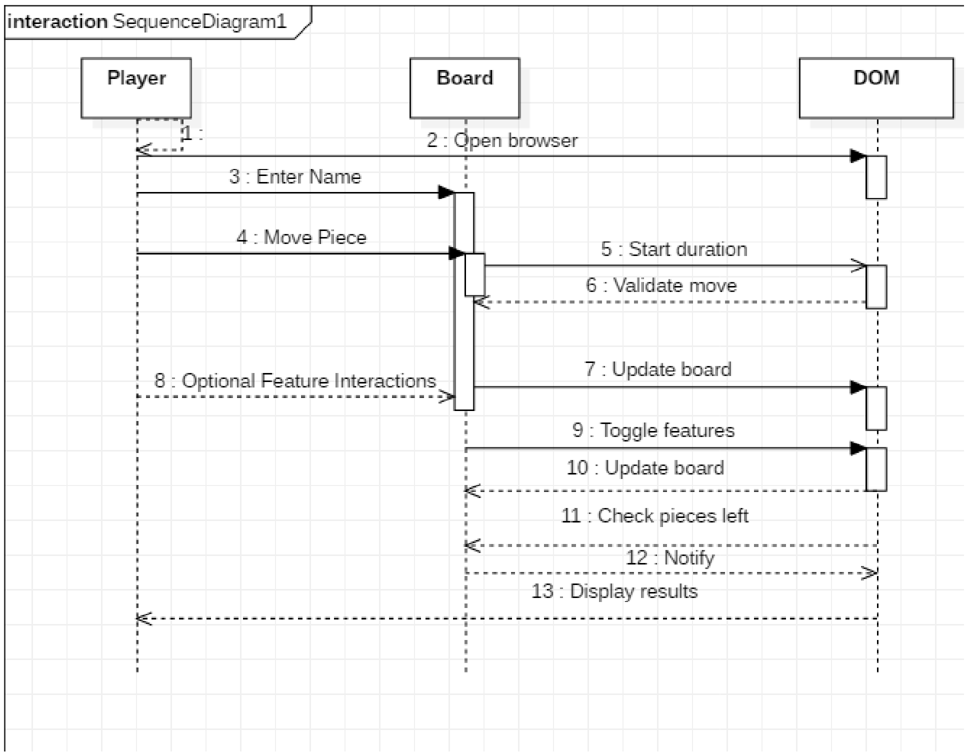
2.2.5 Function Validate

The purpose of this function is for the application to validate if a move is valid or not. This uses the rules of checkers as its base for code.

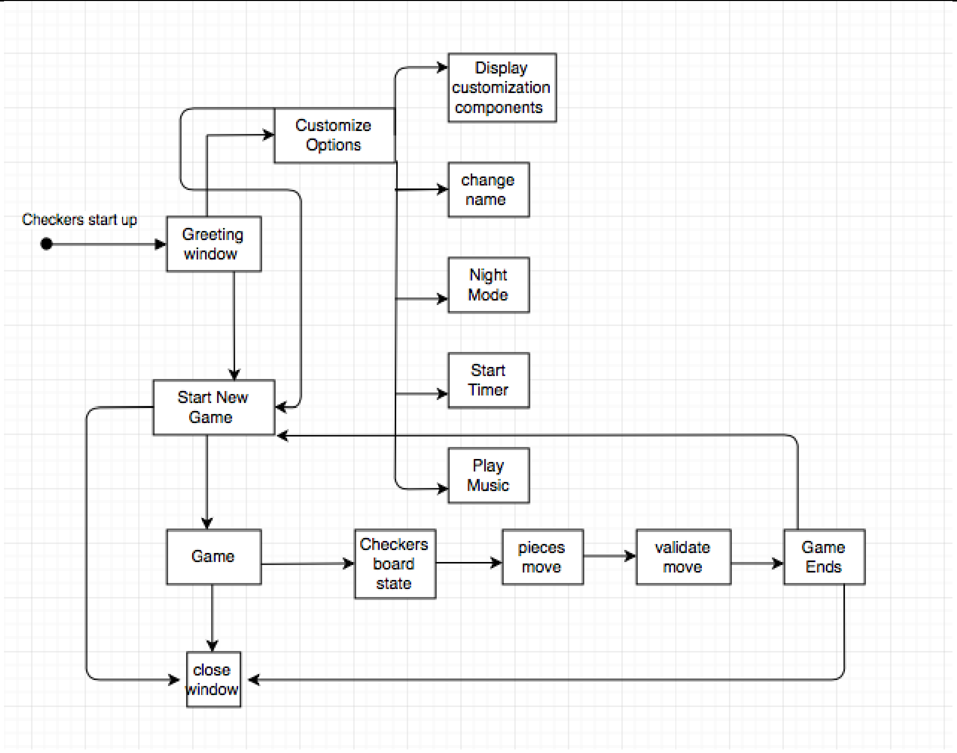
2.2.6 Function Change Game Mode

The purpose of this function is to change between day and night mode for the board.

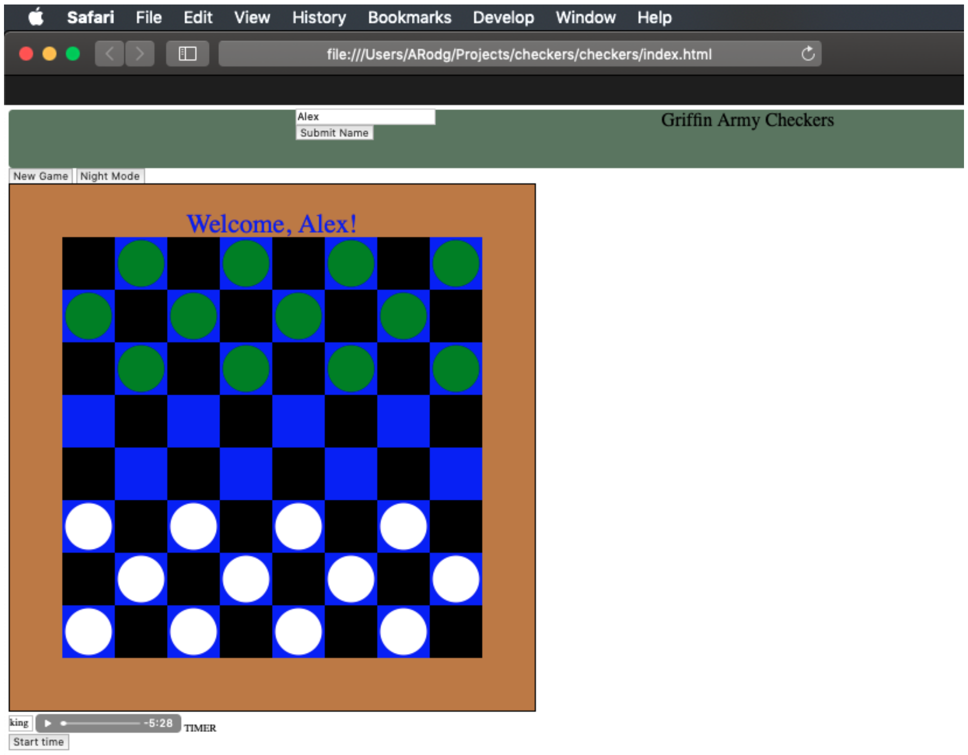
3. High-Level Design



User will enter display name. After the game will start and the board will generate. A piece will be validated then if it meets it will move to the desired square. It will continue until there is a winner or a draw. The user will run the program and will be sent to a web page that has the start for the checkers game. The user has the choice to customize the name displayed for them. The user can toggle night mode on or off. The user can also play music by toggling the button.

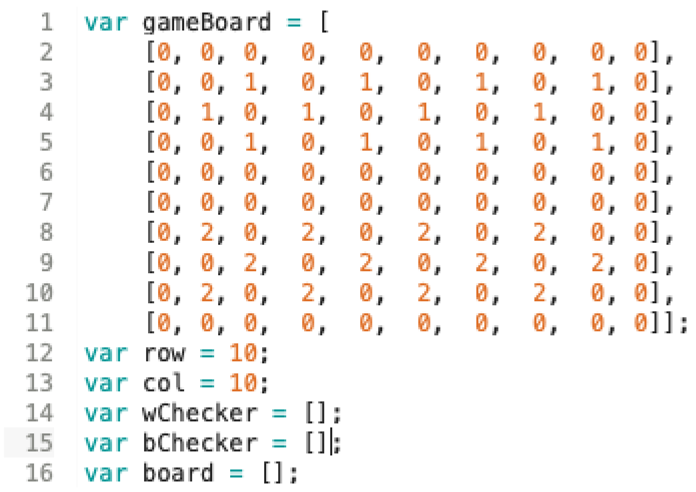


3.1 View / Model Component


Safari File Edit View History Bookmarks Develop Window Help
file:///Users/ARodg/Projects/checkers/checkers/index.html
Alex
Submit Name
Griffin Army Checkers
New Game
Night Mode
Welcome, Alex!
king
-5:28
TIMER
Start time

The web browser is used as the substitute for the model view component. The web browser is the focus for the gaming application because it is set up through a web page. It opens inside the browser and uses it to interact between the user and the application. The model is given by the JavaScript functions and html/css since they work together to create the view. The basic board model set up is shown below in the gameBoard array.

3.2 Code Design Overview



```
var gameBoard = [
    [0, 0, 0,  0,  0,  0,  0,  0,  0, 0],
    [0, 0, 1,  0,  1,  0,  1,  0,  1, 0],
    [0, 1, 0,  1,  0,  1,  0,  1,  0, 0],
    [0, 0, 1,  0,  1,  0,  1,  0,  1, 0],
    [0, 0, 0,  0,  0,  0,  0,  0,  0, 0],
    [0, 0, 0,  0,  0,  0,  0,  0,  0, 0],
    [0, 2, 0,  2,  0,  2,  0,  2,  0, 0],
    [0, 0, 2,  0,  2,  0,  2,  0,  2, 0],
    [0, 2, 0,  2,  0,  2,  0,  2,  0, 0],
    [0, 0, 0,  0,  0,  0,  0,  0,  0, 0]];
var row = 10;
var col = 10;
var wChecker = [];
var bChecker = [];
var board = [];
```

The code design revolved around a 10x10 array instead of a traditional 8x8 due to the fact that it was used to set bounds for the board and not let the user able to move a piece outside the board. Once a piece is clicked the x and y coordinates are taken. Once they are captured, they are divided by 90 to determine what piece will be moved. This process helps with figuring out what checker is selected and validate where to move. The application consists of one JavaScript file, CSS file, and html file. Each one has a purpose to combine once the web browser is launched to view the checkers game.