Software Engineering Group 18 Project

Design Specification

|  |  |
| --- | --- |
| Authors: | Jasper Crabb [jac127] |
| Config Ref: | DSGroup18 |
| Date: | 18th March 2023 |
| Version: | 0.1 |
| Status: | Draft |

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Copyright © Aberystwyth University 2021

CONTENTS

CONTENTS 2

1. Introduction 3

1.1 Purpose of this Document 3

1.2 Scope 3

1.3 Objectives 3

2. Decomposition description 3

2.1 Programs in System 3

2.2 Significant Classes 3

2.3 Table Mapping Requirements onto Classes 3

3. Dependency description 3

3.1 Component Diagrams 3

4. Interface Description 3

5. DEtailed Design 4

5.1 Sequence Diagrams 4

5.2 Significant Algorithms 4

5.3 UML Class Diagrams 4

5.4 Significant Data Structures 4

REFERENCES 4

DOCUMENT HISTORY 5

# Introduction

## Purpose of this Document

The purpose of this document is to describe how we have designed our chess tutor. This document will do this by showing relationships between the requirements, classes, methods and algorithms and how they combine into our final product

## Scope

This document specifies each aspect of our program and how they work in our final product.

This follows the standards laid down by the Design Specification Standards [1]

This document should be read by all project members. Before reading this document the reader should familiarise themselves with the UI Specification [2]

## Objectives

The objective of this document is to show how each component of our program interacts to create the final product.

# Decomposition description

## Programs in System

Our system will only have one program in it. This program is structured into classes which handle graphics (using JavaFx) and the logic of the chess game. The JavaFx classes use event handlers to manage user input and we use JavaFx files for the layout.

## Significant Classes

Our program has many classes but there are some significant classes that are detailed below:

* Game:
  + This class manages the graphical aspects of the game and player input. It initialises the chess board and the buttons on the screen. It also communicates with the board class to update the chess pieces in the backend when they are moved by the player.
* Board:
  + This class contains a 2d array of tiles and acts as a platform that ties the logic aspects of the game together. It also initialises the pieces on the chess board, whether that be a saved layout or the standard starting position
* Tiles:
  + The tiles class is used to store the chess pieces alongside their positions. It allows for us to flip the coordinates of the board without modifying the piece array. This is needed so that we can apply the piece move algorithms to both sides of the board.
* Piece abstract class:
  + This class acts as a template for all of the other piece classes, which are for specific pieces. It has methods for valid move generation and a way to store each valid move in an arrayList

## Table Mapping Requirements onto Classes

# Dependency description

## Component Diagrams

# Interface Description

# DEtailed Design

## Sequence Diagrams

## Significant Algorithms

## UML Class Diagrams

## Significant Data Structures

REFERENCES

[1] Software Engineering Group Projects – Design Specification Standards / 2.3 (Release)

[2] Use Case Document/1.0(Release)

DOCUMENT HISTORY

| *Version* | *Issue No.* | *Date* | *Changes made to document* | *Changed by* |
| --- | --- | --- | --- | --- |
| 0.1 | N/A | 08/03/23 | N/A - original version | Jac127 |