Lavinia Meyer

1009497

MDU118

Tutorial Class 1 (Alan Murray)

Assessment 118.2

Technical Design Documentation

Schedule

Week 2	Draft Documentation - Create delivery schedule - Basic description of project - UML use case diagram and description
Week 3	Final Documentation Due - UML class diagram and descriptions - UI Mockups - Finish document (format, polish)
Week 4	Begin Alpha Build - Create brick grid - Create basic brick type
Week 5	- Allow placement of one brick at a time in grid - Create templates for background - Create special brick types
Week 6	- Allow user to select brick type and background template - Create option for pre-made templates (User doesn't have to place the bricks themselves) - Create brush sizes
Week 7	Finish Alpha Build - Enable saving and loading - Fix bugs - Polish and finish everything above
Week 8	Alpha Milestone Due - Continue fixing bugs & finishing remainder - Ensure all documentation is correct and up to date
Week 9	Begin Beta Build - Test, document bugs
Week 10	- Fix bugs - Polish code
Week 11	Finish Beta Build - Continue polishing code and fixing bugs
Week 12	Beta Milestone Due - Ensure all comments and documentation are correct and up to date

Description

The Brick Breaker level editor will allow the user to create the layout of their own level by selecting a template picture, or by placing their own bricks. If the player selects a template, the editor will create that image using brick colours in the grid.

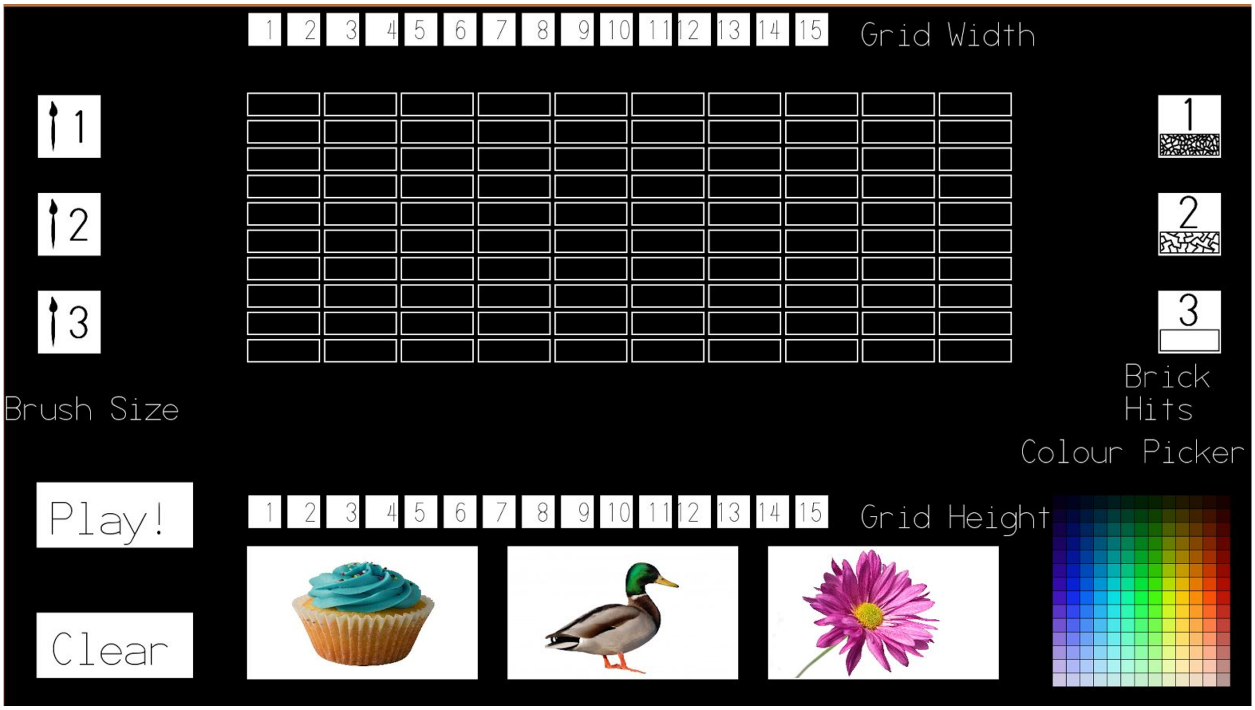
The game will have a pre-existing grid of positions in which bricks can be placed, to ensure bricks are always properly aligned and are the same size. There will also be a pre-existing set of colours to choose from, and a selectable number of hits needed to destroy a brick.

When using the drawing tools, the player will select their brick colour and number of hits from the UI, and can then draw their own pattern on the grid. If the player chooses a template, they can further edit the grid and existing bricks.

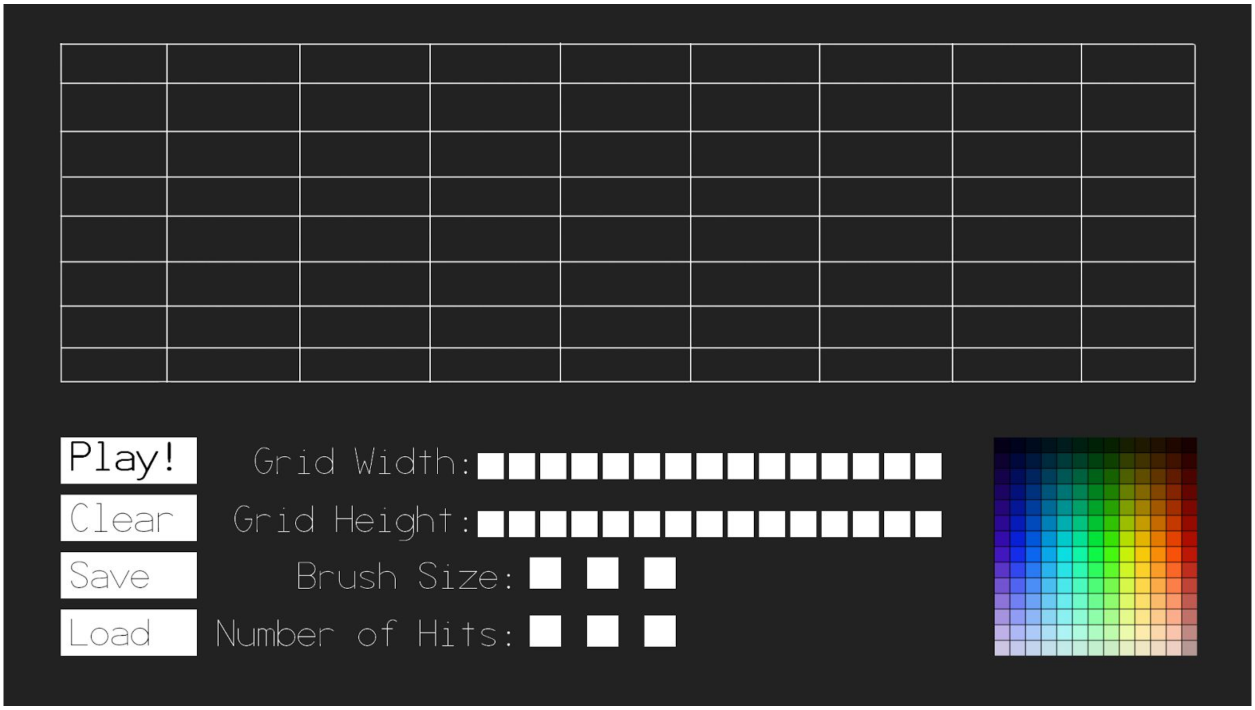
When the player is happy with the level, they can press a “Play!” button to play their level. The player can also change the size of the grid, clear the grid of bricks, and change the brush size. The brush size is the number of bricks placed when the player clicks on one brick.

The code has now been organised into the Crimson Zombie Framework.

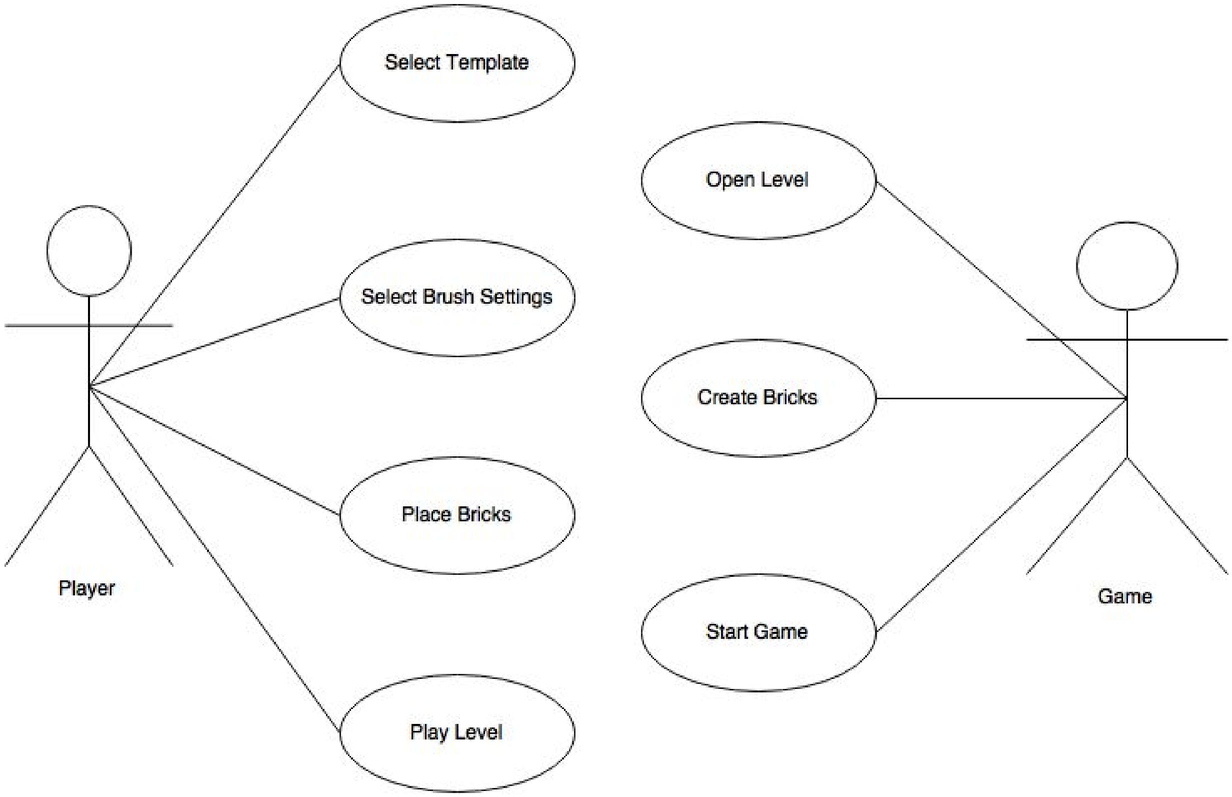
UI:



UI Mockup Beta:


Play!
Clear
Save
Load
Grid Width:
Grid Height:
Brush Size:
Number of Hits:

UML Use Case Diagram



Use Case Descriptions

Use Case Name:	Select Template

Actors:	Player
Description:	The player selects a template to use with the editor. The bricks change colours to mimic the chosen template.
Trigger:	The player clicks a template button on the UI.
Preconditions:	None.
Notes and Issues:	None.

Use Case Name:	Select Brush Settings

Actors:	Player
Description:	The player selects the settings for the bricks they place. They can choose how many hits it will take to destroy the brick, what colour the brick is, and the brush size to use when placing bricks. They can also choose the size of the grid.
Trigger:	The player clicks the colour, pattern, grid or brush size buttons on the UI.
Preconditions:	None.
Notes and Issues:	None.

Use Case Name:	Place Bricks

Actors:	Player
Description:	The player places bricks on the grid.
Trigger:	The player clicks spaces on the brick grid.
Preconditions:	None.
Notes and Issues:	None.

Use Case Name:	Play Level

Actors:	Player
Description:	The player plays their level in Arkanoid.
Trigger:	The player clicks the finish button after creating their level.
Preconditions:	The player must have placed bricks on the level.
Notes and Issues:	None.

Use Case Name:	Open Level

Actors:	Game
Description:	The game Arkanoid is loaded.
Trigger:	The player finishes their level and selects play.
Preconditions:	The player must have created a level in the level editor.
Notes and Issues:	The game will not be included with the editor - it is mentioned here because in a full program the editor would be used to also play the game, rather than to only design a level.

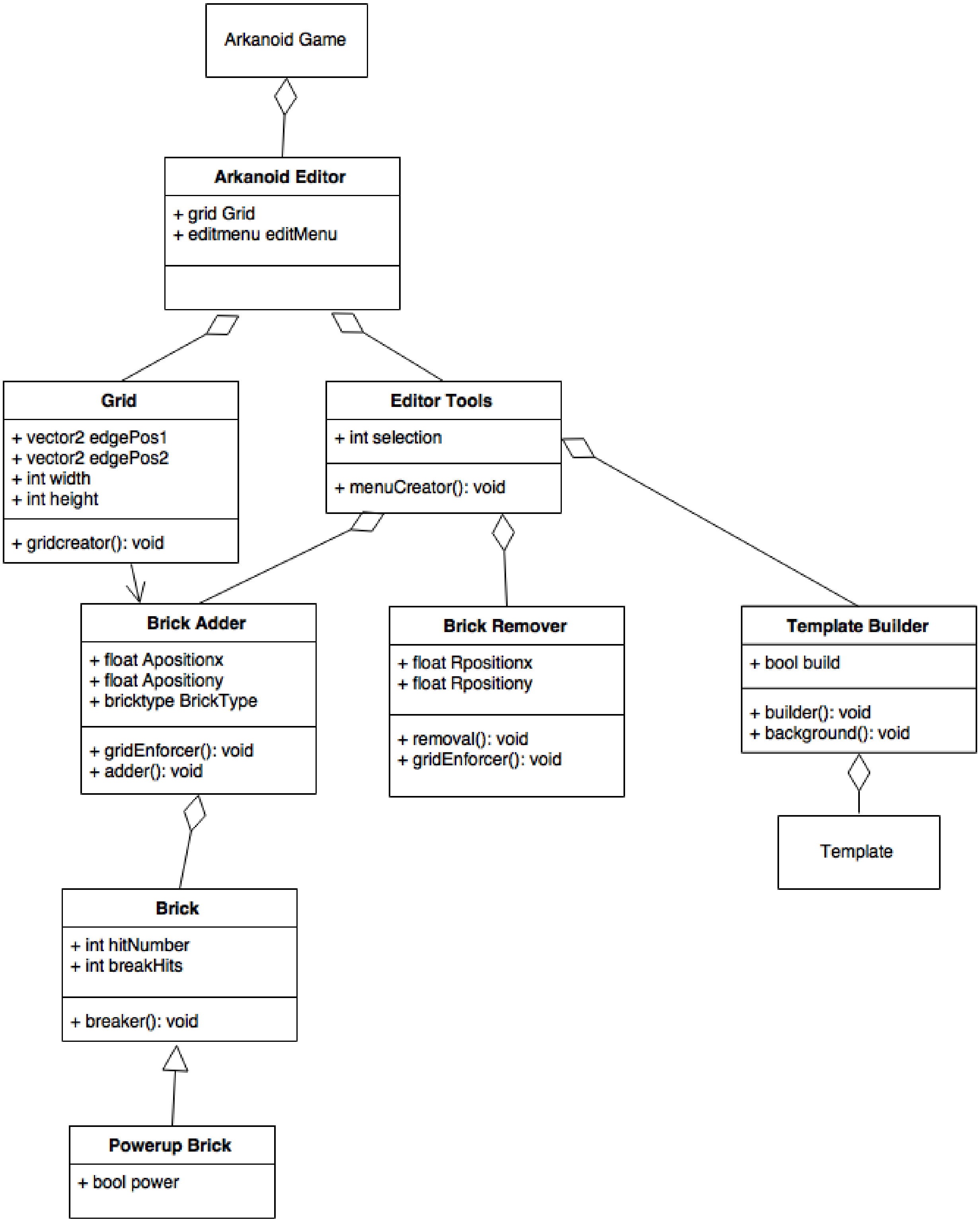
Use Case Name:	Create Bricks

Actors:	Game
Description:	The game uses the placement of bricks the player has chosen in the editor to create the level.
Trigger:	The game loads after the player chooses to play their level.
Preconditions:	The player must have created a level in the level editor.
Notes and Issues:	The game will not be included with the editor - it is mentioned here because in a full program the editor would be used to also play the game, rather than to only design a level.

Use Case Name:	Start Game

Actors:	Game
Description:	The game begins.
Trigger:	The game uses the level that the player created to allow the player to play the game.
Preconditions:	The player must have created a level in the level editor.
Notes and Issues:	The game will not be included with the editor - it is mentioned here because in a full program the editor would be used to also play the game, rather than to only design a level.

UML Class Diagram



Class descriptions

Powerup Brick

Powerup bricks are bricks which contain a boolean value indicating that they are a powerup. In game, this variable will be used to create a powerup when the brick is destroyed. The powerup brick is a child of the regular brick.

Brick

All bricks can be placed on the grid at certain positions. Each brick stores the variable hitNumber to track how many times the brick has been hit, and breakHits to store how many hits are needed to destory the brick. The breaker method is used to break the brick once it has been hit enough times.

Brick Adder

The brick adder takes the screen coordinates of where the mouse hovers, and calculates where the nearest position is on the grid where a brick can be placed using the gridEnforcer method. The bricktype variable stores which type of brick is being placed, and the adder method creates the brick in the position given by the grid enforcer.

Brick Remover

The brick enforcer works in a similar way to the brick adder, however instead of storing any specific type of brick it will remove any bricks underneath the grid position given by the grid enforcer.

Template Builder

The template builder creates a template level in the editor, either as a background for the player to use as a guide using the background method, or as a built level with pre-placed bricks using the build method. This is a choice for the player, stored in the build variable.

Template

The template object is the stored templates available for the player to choose from. These templates are used by the template builder when the player uses a template.

Editor Tools

The editor tools are a menu created in the UI to allow the player to select which brick options and template they want to use. The various buttons trigger the brick adder, brick remover, and template builder.

Grid

The grid is the set of screen coordinates and settings which determine where the player may place bricks. The vector2 positions of the top left and bottom right corners are stored to position the grid, and the resulting width and length of the grid are divided by the width and height variables to create a grid for bricks to be placed on.

Arkanoid Editor

The Arkanoid editor is the editor itself. It is used by the player to design a level, which is then used by the Arkanoid game for the player to play. The Arkanoid editor is made up of the grid and the editor menu.

Arkanoid Game

The Arkanoid game is not included with the editor in this program, however in a full program it would use levels built by the editor to allow the player to play the game.