UniBOOM

**Technical Design Document**

(v0.2)

Table of Contents

[1. Overview 3](#_Toc260387709)

[2. Requirements 3](#_Toc260387710)

[2.1. Brief 3](#_Toc260387711)

[2.2. Reference 3](#_Toc260387712)

[3. Dependencies 3](#_Toc260387713)

[4. Existing Technology 4](#_Toc260387714)

[4.1. Features 4](#_Toc260387715)

[4.2. Reference 4](#_Toc260387716)

[5. Implementation details 4](#_Toc260387717)

[6. Proof 5](#_Toc260387718)

[7. Issues 5](#_Toc260387719)

[8. Risks 6](#_Toc260387720)

[9. Estimates 6](#_Toc260387721)

# Overview

This system, UniBOOM, is a bomberman game. It shall let players, as Unity-chan in the game, experience a classic bomberman gameplay, with some innovative features.

The system shall include all typical bomberman gameplay, such as moving in the stages, setting up bombs, break walls, killing enemies/opponents, and getting bonus items.

Besides, the system shall provide players some action game features and role playing game features, such as playing this game in third-person perspective, moving at different speed, minimaps, detecting enemies/components and changing equipment/skill points.

The system shall also provide a stage editor, which can make custom stages to play. Stage data shall be saved as files in hard disks.

# Requirements

## Brief

* Pre-game Features
  + Setting up games, including graphics, audio, etc.
  + Choosing Single Player/Multiplayer mode
  + Choosing local/network play, if in Multiplayer mode
  + Choosing game mode
  + Choosing levels which have been played before
  + Choosing characters
  + Saving/loading games
  + Loading stages from files
* Game Features
  + Moving in the stages
  + Setting up bombs
  + Bombs explode and generate damage zones
  + Breaking blocks (using bombs)
  + Damage enemies/component/self
  + Items drop from blocks
  + Picking up items and being granted abilities
  + Recording scores for players
  + Enemy AI
  + HP bars for players and enemies
  + Third-person perspective (as another game mode)
  + Minimap (in third-person mode)
  + Moving at different speed (upon holding different buttons)
  + detecting enemies/components and showing on minimap
  + Change stages upon winning
  + Pausing, exiting or restarting game
* Network features
  + Multiplayer game in LAN
* In-game stage editor
  + Saving/loading stage data
  + Putting/removing blocks (breakable)/walls (unbreakable)
  + Putting/removing enemies
  + Setting spawn point
  + Setting background and BGM

# Dependencies

* Unity documentation (<http://unity3d.com/learn/tutorials/modules>)

# Implementation details

Pre-game Features

This part describes an overview of the implementation of pre-game features.

Most features here are menu matters. Unity UI System will be used to implement those features.

‘Press any button’ is shown very first at the main scene. Upon pressing a button, the menu will be shown.

Main Menu:

* New Game
* Load Game
* Quit Game

Upon entering new game or loading previous game:

* Game Start
* VS Mode
  + Local Game
  + LAN Game
* Options
* Exit

Game Features

This part describes an overview of the implementation of all game features.

**Memory Implications:**

<How will this effect Memory, it may not>

**Performance Implications:**

<How will this effect Performance, it may not>

**Networking Implications:**

<How will this effect Networking, it may not>

# Proof

To be discussed.

<Example1 – math optimization>

< Unit Test – write unit test for sqrt function… >

<Example2 – Shadows>

< Test Case - We can see smooth shadow on ground that casted by cars and dynamic objects... >

<Example3 – Bullet optimization>

< Verify - After bullet optimized, the function World::stepSimulation() calling time will be in 14ms. >

# Issues

To be discussed.

* <Bullet points on any issues for this task.>

# Risks

To be discussed.

* <Bullet points on risk, mitigations and contingency plans> <Chance of Risk> <General Time Estimate if risk happens>

# Estimates

To be discussed.

<Tasks should be broken into as small pieces as makes sense. Normally no task can be completed in under half a day (considering check in times and building/syncing ect). If a task is more than a day though we should try to break it down into what will be tackled on day vs the next. This being said sometimes a task is 3 days long with no way to break it down more than that. Although this shouldn’t be the norm it does happen and is acceptable. If something is 5 days long make sure that more research isn’t necessary to break it down further. It’s often a sign that not enough research has gone into a task if it can’t be broken into smaller pieces than a week, though it can happen.>

|  |  |
| --- | --- |
| Tasks | Estimate in days/person |
|  |  |
|  |  |
|  |  |
| Total |  |