Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

GAME NAME  
Cross Platform Development

Author: Thomas Dufresne

# Change Log

Updates made to the document should be described below.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Date of change | Description |
| 0.0.0 | AIE | 31/08/2020 | Initial Template created |
| 0.1.0 | Thomas D | 8/09/2020 | Begin writing design document.  Add game overview, controls, game flow, items, and progress report for Friday 4th. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Change Log 1](#_Toc49774281)

[1.0 Development Environment 3](#_Toc49774282)

[1.1 Game Engine 3](#_Toc49774283)

[1.2 Source Control 3](#_Toc49774284)

[1.3 Third-Party Libraries / assets 3](#_Toc49774285)

[2.0 Game Overview 3](#_Toc49774286)

[2.1 Genre 3](#_Toc49774287)

[2.2 Camera Perspective and Movement 3](#_Toc49774288)

[2.3 Platform 3](#_Toc49774289)

[2.4 Technical Goals 4](#_Toc49774290)

[2.5 Game Objects and Logic 4](#_Toc49774291)

[3.0 Controls 4](#_Toc49774292)

[3.1 Windows / Web 4](#_Toc49774293)

[3.2 Console / Xbox 4](#_Toc49774294)

[3.1 Android / Touch 4](#_Toc49774295)

[4.0 Mechanics 4](#_Toc49774296)

[5.0 Graphics 5](#_Toc49774297)

[6.0 Audio 5](#_Toc49774298)

[7.0 Artificial Intelligence 5](#_Toc49774299)

[8.0 Game Flow 5](#_Toc49774300)

[8.1 ‘Mission’ / ‘Level’ structure 5](#_Toc49774301)

[8.2 Objectives/Goal 5](#_Toc49774302)

[11.0 Interface 5](#_Toc49774303)

[11.1 Menu 5](#_Toc49774304)

[11.2 High scores 5](#_Toc49774305)

[11.3 UI/HUD 5](#_Toc49774306)

[12.0 Progress report and feedback Meeting Minutes 6](#_Toc49774307)

[Friday 4th September 6](#_Toc49774308)

[Wednesday 9th September 6](#_Toc49774309)

[Thursday 10th September 6](#_Toc49774310)

[Friday 11th September 7](#_Toc49774311)

# Development Environment

## Game Engine

Unity version 2019.3.6. This is used as it is the version currently installed on campus computers.

## Source Control

Link to GitHub repo: https://github.com/TheTastyGravy/Six-DoF-Game

## Third-Party Libraries / assets

*<State and explain the reason of use for any third-party libraries, assets from engine specific market places (Approval will be required) or packages.****All chosen third party libraries must be reviewed before adding to project by supervising teacher and licenses must be checked****>*

|  |  |  |
| --- | --- | --- |
| Asset Name  License | Url | Reason for use |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Game Overview

## Genre

6DoF FPS Action game.

## Camera Perspective and Movement

Controlling a drone in first person, with 6 degrees of freedom, i.e. able to move vertically and roll the camera.

## Platform

PC, iOS, android, Xbox, web. Unity makes deploying to multiple platforms very easy, with everything just working with very few exceptions. Through development, any such components that are not cross platform will be avoided.

## Technical Goals

* Options menu using PlayerPrefs
* Saving and loading
* *Goal*
* …

## Game Objects and Logic

*A list of logical elements in the game, i.e. door, button, pistol, ammo, light, bullet, wall, character etc. and description of their behavior and purpose*

* The **player** has 2 ‘currencies’: health and ammo. Ammo is used to fire their weapon, and health is reduced upon taking damage, and when it reaches 0, the player dies, reaching a game over state. Both of these currencies are increased by collecting pickups, described later. The player takes user input to be able to move, rotate, and fire their weapon. The weapon consists of a single script with a Shoot function, called by the player. The script creates a bullet and applies a force to it in the forward relative to the player. Upon entering a trigger with the tag ‘Pickup’, it will call a function from its script, then destroy the game object.
* **Pickups** are simple objects that only have a trigger and a pickup script. The player will call a function in the script upon entering its trigger. This function will give the player ammo and health equal to its preset values.
* **Bullets** are another simple object. They have a collider and a bullet script, which will call a deal damage function on any object it collides with, before destroying itself.
* **Shields** contain no logic, but are instead used by bullets. They only have a trigger, but when a bullet enters a shields trigger, it will destroy itself. This results in an object that players and enemies can pass through, but not bullets.

# Controls

A 6DoF game naturally requires 6 degrees of input. This is trivial for mouse and keyboard, and doable on a controller. Touch controls for mobile, however, is more difficult. Input is limited to the screen space, so the player will only be using their thumbs to play the game, reducing the possible degrees of input to 4. Because of this, the mobile version of the game will not feature vertical movement or rolling the camera.

## 3.1 Windows / Web

Mouse and keyboard using WASD and space/shift for movement, mouse and Q/E for camera control. Esc to pause, left click to fire, F to interact.

## 3.2 Console / Xbox

Controller using left stick and bumpers (vertical) for movement, right stick and D-pad (roll) for camera control. Right trigger to fire, start to pause, A to interact.

## 3.1 Android / Touch

Custom touch joysticks and buttons are necessary for mobile input. Due to the lack of controls available, the mobile version will lack vertical movement and the ability to roll the camera, but the game will remain playable.

Left joystick for movement, right joystick for camera control. Left button to fire, upper-right UI button to pause, right button to interact.

# Mechanics

*A list of intended core game mechanics. I.e., what the player can do and how they achieve this, and what this will trigger in the game. For example, shooting enemies is a core mechanic in an FPS.*

* ***Shooting****how does it work, direction of bulets, damage, speed, etc.*
* ***Jumping***

*how does it work, realistic, curved, double jump, how high?*

* ***Moving platforms****how or when do they move*
* ***Pressure plate****How does it work / active / inactive etc.*
* *Etc…*

## Hazards

*These are items or areas in the game that are considered dangerous, eg: jump gaps, spikes, lava.  
Depending on your game, this may or may not be relevant.*

## Obstacles

*These are thigs in the game that move and can provide additional challenges to reaching your goals, might include “Goomba” from Mario.*

## Items / Collectables

*Things in the game that can be collected. Coins, health packs, weapons, powerups etc… eg, for space invaders, enemies might drop a powerup for the player to collect. How are the items collected? Player collision, are they shot at? Etc.*

# Graphics

*Describe graphics features here. I.e., is your game top-down 2D? What post processing are you using? Include perspective, art style, graphic features. Justify graphics selection.*

# Audio

*Describe audio requirements. Sounds Effects, Ambient music etc.*

# Artificial Intelligence

*Describe how your AI will works, i.e. state machine, fuzzy logic, GOAP. Describe the various behaviors*

# Game Flow

## ‘Mission’ / ‘Level’ structure

Different levels will be in separate scenes, with the appropriate scene loaded upon entering the level. Data will be saved using the [binary formatter](https://answers.unity.com/questions/1320236/what-is-binaryformatter-and-how-to-use-it-and-how.html) to load a [serialized](https://docs.unity3d.com/Manual/script-Serialization.html) class from file, containing the player’s health, ammo, checkpoint and level. The level’s scene will be loaded, and the player will be initialized with the health and ammo at the designated checkpoint. Upon reaching the end, the checkpoint will be reset and the level index increased, with the player’s health and ammo updated.

## Objectives/Goal

The player simply moves through the level, taking out enemies, until they reach the exit.

1. Levels

*If any of the Levels require specific behaviors, describe those here. UML chats provided if applicable.*

*Level tiling tool use identified if relevant, use by designer discussed, how was it built*

1. Items

The only items will be pickups for health and ammo, which will be placed throughout the level and possibly dropped by enemies.

# Interface

*Make sure to address the differences needed per platform.*

## Menu

*What are the menu options, how is it presented to the player? Provide wireframe.  
How does this work for each input device chosen (keyboard/mouse, controller, touch)*

## UI/HUD

*What is involved in the UI/HUD, what information is being provided to the player. Mock up of intended UI/HUD design*

# Progress report and feedback Meeting Minutes

## Friday 4th September

Describe state of project

* Prototype with most of its core features working.

Feedback from teacher and peers:

* A good base without much substance.

Action Items:

* Work out the details of what the game will be.

## Wednesday 9th September

Describe state of project

* Thing
* Thing

Feedback from teacher and peers:

* Describe
* Describe
* Describe

Action Items:

* Describe
* Describe
* Describe

## Thursday 10th September

Describe state of project

* Thing
* Thing

Feedback from teacher and peers:

* Describe
* Describe
* Describe

Action Items:

* Describe
* Describe
* Describe

## Friday 11th September

Describe what has been done since last time

* Thing
* Thing

Feedback from teacher and peers:

* Describe
* Describe
* Describe

Action Items:

* Describe
* Describe
* Describe