**[3D-Platformer]**

**Technical Design Document**

Version [0.3]

**Version History**

|  |  |  |
| --- | --- | --- |
| **Version Number** | **Edited By** | **Date** |
| 0.1 | James Aiken | 4/28/2017 |
| 0.2 | James Aiken | 5/7/2017 |
| 0.3 | James Aiken | 5/25/2017 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[Game Overview 3](#_TOC_250023)

[Game Summary 3](#_TOC_250022)

[Platform 3](#_TOC_250021)

[Development Overview 3](#_TOC_250020)

[Development Team 3](#_TOC_250019)

[Development Environment 3](#_TOC_250018)

[Development Hardware 3](#_TOC_250017)

[Development Software 3](#_TOC_250016)

[External Code 3](#_TOC_250015)

[Game Mechanics 4](#_TOC_250014)

[Main Technical Requirements 4](#_TOC_250013)

[Architecture 4](#_TOC_250012)

[Game Flow 4](#_TOC_250011)

[Graphics 4](#_TOC_250010)

[Audio 4](#_TOC_250009)

[Artificial Intelligence 4](#_TOC_250008)

[Networking (If Applicable) 5](#_TOC_250007)

[Physics (If Applicable) 5](#_TOC_250006)

[Game Objects and Logic 5](#_TOC_250005)

[Data Management and Flow 5](#_TOC_250004)

[User Interface 5](#_TOC_250003)

[Game Shell 5](#_TOC_250002)

[Play Screen 6](#_TOC_250001)

[Technical Risk 6](#_TOC_250000)

# Game Overview

This section provides an overview of the game, suitable to someone new to the project.

## Game Summary

Provide a one or two paragraph overview of what the game is about. This material can come from your pitch, or can be something completely new for this.

The game will be a 3D-Platformer based on titles of past such as Banjo-Kazooie and Super Mario 64 and Sunshine, and of a more recent title Yakoo-Laylee. The game will have a overworld or “hub” for the player to cool down between levels and play with some mechanics and their known abilities to explore a bit, experiment, and blow off steam. The game will include a level designed to fit more than one or two things to do. This is to offer replay-ability and to allow the level to be multi-purpose.

## Platform

Describe the hardware/software platform that the game is going to be played on. This can come from the corresponding section of your pitch.

Windows 10

# Development Overview

This section discusses key aspects of the development of the game, as opposed to the game itself.

## Development Team

List the team members and their roles during the development process.

James Aiken – Project Manager, AI Programmer, Secondary Programmer  
Matthew Kelly – Animator, Sound Design, Secondary 3D Modeler  
Kelly Kreutz – Lead 3D Modeler and Artist  
Josue Lopes – Graphics/Shaders/Effects Programmer  
Alex Williams – Game Mechanic Programmer

## Development Environment

This describes the environment used to develop the game. In theory, someone could set up an environment like what is described here and everything needed to work on the game.  
  
Unreal Engine 4.15.1  
Windows 10  
3DS Max  
Photoshop

### Development Hardware

Provide a brief description of the hardware used during development. Provide hardware specifications as well as operating systems, and anything relevant that way.

### Development Software

Provide a description of any tools used during development, including IDEs, compilers, editors, debuggers, and so on. If you are using any version control software, backup software, and so on,

mention that here too.

### External Code

Provide a description of all of the code used but not development by the team. This includes engine code, libraries, 3D APIs and drivers, and so on. A discussion of the content pipelined

used to bring art and sound assets into the game would be useful as well.

# Game Mechanics

This is the main part of the document. This goes through the architecture and implementation notes on the main subsystems and control elements of the game.

## Main Technical Requirements

This provides a description of the key technical requirements imposed by the game’s design. This will include critical functionality as well as constraints on this functionality (like performance, and so on). Some of these issues can be discussed in the other subsections under Game Mechanics.

## Architecture

In this section, you provide a description of the game’s architecture, including all of its modules and the interactions between them. This should be done in text, as well as using one or more diagrams to depict the architecture.

## Game Flow

This section provides a discussion of how control flows through the game. This includes a

description of the main control loop of the game and how it interacts with the various parts of the game, as well as the various different states that the game can be in (title screen, menus, playing, paused, and so on, perhaps with sub-states). Again, diagrams should be provided.

## Graphics

This section discusses the 3D/2D graphics elements of the game. Focus on the core aspects as opposed to interface elements, as these will be discussed later. (This would include rendering, sprite management, and so on.) If there are any constraints on the type of content that can be used here, it should be discussed.

## Audio

This section discusses the sound elements of the game, including music playback, sound effect playback, and so on. If there are any constraints on the type of content that can be used here, it should be discussed.

## Artificial Intelligence

This section discusses the control of all non-player active entities in the game. Identify how this control is done (scripting, state machines, rules, and so on) and provide details for each entity.

## Networking (If Applicable)

If there are online features to the game, they should be discussed here. This includes a description of protocols used, and other technical details.

## Physics (If Applicable)

If physical simulation is used, for collision detection, object interactions, and so on, a discussion of what is done and how is provided here.

## Game Objects and Logic

This is a discussion of all gameplay objects and logic used to support the game, as opposed to the various core subsystems discussed above. (One way of thinking about it is that these are all the game specific elements to your game … the other aspects described in the subsections above could theoretically be provided by an engine, and might be, depending on your game.)

This should include subsections for each main game object or set of objects.

## Data Management and Flow

Game data will need to be loaded and saved at various points in time. This section should describe the procedures used to do loading, saving, and any processing of this data (such as compression/decompression, encryption/decryption, parsing, and so on), as well as the various different types of files used to contain this information.

# User Interface

This section describes how information is presented to the player, and how input is received in return. This includes display layout, feedback, controls, and so on.

## Game Shell

This discusses all screens presented to the player outside of game play. If there will be sound,

music, or tactile feedback in any of these screens, it should be discussed. Controls should also be discussed to describe how screens are changed or how interface elements (buttons, etc.) are manipulated by the player.

The game will likely have menus given to the player to control the flow of out-of-game activities. This includes setting game options, controlling game flow (starting, stopping, pausing, and so on), and a variety of other tasks. The various menus must be identified, as well as the flow between them, and circumstances under which they appear and disappear.

Additionally, other screens may be presented to the player. This includes splash screens with logos and/or legal notices, credit screens, and so on. These should also be discussed.

Diagrams should be provided both as concept art and to illustrate screen flow.

## Play Screen

This is a discussion of what is presented to the player during game-play. This includes the depiction of all game elements, as well as other information presented in the game’s heads up display (HUD). Any audio and tactile feedback provided should also be discussed. Diagrams should be provided as concept art to illustrate how the game screen appears during play.

In addition, a discussion of all controls should be provided. Not only should this describe what each control is and how it impacts gameplay, but this should also include how control events are processed and disseminated to the various subsystems and game objects as described above.

# Technical Risk

In this section, you discuss the various elements of technical risk associated with the development of your game. This can include things such as unfamiliarity with the platform or development tools being used, complex aspects of the game, other constraints on your schedules or resources, and so on. Each risk should be accompanied by a strategy to minimize, reduce, or mitigate the risk, to ensure that you will be able to handle the challenge appropriately.

# Each risk (or set or related risks) can be discussed in their own subsection. Notes

All assets brought into Unreal Engine should be organized properly so the group can keep track and understand the location of everything.   
  
Ex. Assets > Exported Models > Collectibles > Coin  
Assets > Exported Models > Level Details > Pillar  
Assets > Blueprints > Collectibles > Coin\_Blueprint  
  
Models are good to keep separate because then in the Engine we do have a backup and possibly something to use stand-alone without a blueprint. Blueprints tend to become separate things in Unreal, so we can organize them into their own folder.   
  
All 3DS Max files should be kept together and labeled with an idea of what it contains.  
All Exported .FBX files should also be kept together and given a name of what they are with a version number.   
  
Ex. “Pillar\_vers\_01” This will help identify how old a model could be, and it means if we decide to go back to a previous version and use it again in Unreal, it’s possible.   
  
  
**Meeting Details**

**Details Covered in Meetings**

**Week of May 7th to 13th**  
Steampunk Platformer Mechanics

1. Basic Attack

2. Mechanical Wings/Glider

3. Thruster

4. Grappling Hook

5. Camera

World has multiple objectives to complete the whole world.

Based on scale of world and puzzles, player may leave level and return to airship, or may stay in the world and keep running around.

Only one world accessible at the start of the game. Player does not see true scale the game could be until after first world is completed and they collect a map piece.

Hub World - Airship

Secondary Character - Person who introduces new mechanics to you and can give advice

Sound Design

Voice Acting - Possibility but low priority

World Design

- Airship as a Hub World

- Acts as player's home, related characters home,

- Used for travel between worlds

- Stationed in open air or nearby first world

- Each Level has multiple objectives to complete whole level

- Each level's main collectible is a Map Piece.

- Each level has other collectible's

Artwork Design

- Airship

- Main Characters

**Week of May 21st to 27th**  
  
First Level Theme – Jungle/Temple/Tomb  
  
Protagonist - Wears a Mask or facial cover. This hides the gender and allows for mechanics to possibly be associated with the mask.  
  
Environmental Mechanics Ideas  
 - Spike Traps  
 - Swinging Traps  
 - Swinging Platforms  
 - Grappling Points for the Grappling Hooks  
 - Tree-Top Trekking  
 - Rolling Boulders  
 - Possibly basic enemy AI  
  
Story/Motivation  
- A single continent that featured a civilization spread out in very different biomes.   
- This civilization developed a dangerous weapon and it led to the continent breaking and splitting into different pieces.   
- Each level houses a piece of this super weapon.  
- There’s a villain trying to gather all the pieces of this weapon to use it for malicious purposes  
- The player was originally just looking to explore and uncover secrets of the past  
- At the end of the first world/level, they run into the villain and this introduces a more overarching plot.   
- Player must race against the villain to collect pieces of the super weapon

Next Step

* Decide on details related to concept art and make changes where necessary
* Build mechanics in an Unreal Test Environment to prototype and experiment with what works and does not work.
* Agree on additional details/objects/items that fit the theme of the level and what could possibly work for puzzles
* After mechanics have been prototyped, consider puzzles, level layout, what mechanics should stay or go