The Adventures of Sam the Pirate

Design Document

A Swashbucklin Studios Production

<http://swashbucklinstudios.com>

# 

# 

*Contents*

[The Pitch](#h.o0dimatg0jan)

[Concept](#h.7m0kia6i2rzm)

[Game Overview](#h.yc2im9bk7pc0)

[Release Platforms](#h.qbx1mfu9dlre)

[Reasons for Joining the Project](#h.2s4zuq35bp0z)

[Team Management](#h.o6fnbiyxyezv)

[Team Creation](#h.4h4tg61x9h1)

[Team Members](#h.4ndxad1biu0q)

[Specialist Interest Area](#h.pxl16otqnttf)

[Research Capstone](#h.hezojybooodg)

[Development Capstone](#h.nav7wb65mvsx)

[Project Planning](#h.rzofupgcbgnx)

[Project Goals](#h.ctqwajk8zw5m)

[Milestones](#h.6qmltrwq399b)

[Stretch Milestones](#h.jkvbkofe5q6y)

[Marketing](#h.ordu7cvquzyc)

[Meetings](#h.qzp2ei3vm0m1)

[Research](#h.hd83m0mbm9vt)

[Design and Development Tools](#h.idxipq2mqgne)

[Project Management](#h.b511275wa9bv)

[Version Control](#h.hotl92dbdwdm)

[Programming Environment](#h.3ry8mx65j7vj)

[Graphics and Input](#h.veeh3c8fvk6)

[Physics](#h.ix44b02dzur9)

[Sound](#h.qdcg7xsarwsj)

[Data Storage](#h.jo4wvic81jj7)

[Market Demographics](#h.ccg6owfc8qs6)

[Legal](#h.bn52vshm5z5z)

[Profit Sharing](#h.vyiwvno9q457)

[Social](#h.rup6bkcnb06w)

[Ethical](#h.kf5wigc2xvqn)

[Game Design](#h.r1umxbl12820)

[Prototype](#h.2w5tu6o3mblg)

[Narrative](#h.qk2pwd290vb1)

[Gameplay](#h.h9zve01bdwtc)

[Score, Collectables, and Time](#h.4z2gl0vgu5pn)

[Level Timing](#h.ki5bbzh9a384)

[Art Style](#h.20cmk9tk0fv5)

[Art Layers](#h.og3lfttlszz8)

[Concept Art](#h.5ds0eq4nvdzt)

[Characters](#h.qqdnglqrhqi9)

[Level Backgrounds](#h.yrb5ob1jz14p)

[Environment Objects](#h.brgbdbr7r79y)

[UI Mockups](#h.rmh5n9yjkli3)

[Priority List](#h.1i09zy5jlyld)

[Interface](#h.3l7kvh96evnr)

[Characters](#h.frpi88rh4pc4)

[Level Backgrounds](#h.giv430sn77fe)

[Environment](#h.78y1kp3vxc3s)

[Game Flow](#h.k5wnashvp5vv)

[Menus](#h.tnp2291q62nn)

[Gameplay](#h.5mtfl3qeto88)

[Overworld Map](#h.b7vexmq7tlfc)

[Levels](#h.hpnxahgjjaox)

[Enemy Types](#h.g084re3qtx1a)

[Artificial Intelligence](#h.qhzlxztt9f9j)

[Level Design](#h.flgzws4vdefl)

[Boss Encounter Levels](#h.vc6e3gkp0a7r)

[Level Progression](#h.vgpf0t478a28)

[Controls](#h.svh406lx9whb)

[Menus/Overworld](#h.v45hon3oelky)

[Levels](#h.t3az16nodl2s)

[Level Editor](#h.1x13wt1zy4h3)

[Mock Up](#h.w7h2o2vcxug1)

[Audio](#h.fipvd1h5aaq4)

[Design](#h.1cv302rzj21k)

[Sourcing](#h.713hohzxf55)

[Music](#h.vfq5ceucsinx)

[Sound Effects](#h.9xfv9351ky3f)

[Design Assumptions](#h.mu17sl7t1d2r)

[Technical Design](#h.cguemp6fko74)

[Design Goals - The Papyrus Engine](#h.uh7203vya0qs)

[Overview](#h.bui31muyomk7)

[Unit Tests](#h.j1gx9zqm4tvu)

[Rendering](#h.73nr48ez6f0k)

[Input Events](#h.dr7angwojih2)

[Logging](#h.9gg1b5poqmcl)

[Text File Parsing](#h.ok6bxzmvpsxv)

[Sprites](#h.xrxo933ugw1)

[Parallax](#h.cd336yhnw40l)

[Physics](#h.ae4dh7xzsyox)

[Sound](#h.3rlqdxgwls6e)

[User Interface](#h.x0kasxshcg3y)

[Font](#h.bnyupxiuknco)

[Installers](#h.qle8a8xa68dx)

[References](#h.x2l68nfl6jcb)

# 

# The Pitch

## Concept

The Adventures of Sam the Pirate is a side scrolling 2D platformer game inspired by the likes of Super Mario Brothers, Super Meat Boy, and Rayman Legends. The game is targeted to be released on Steam.

The focus of the game is on platforming and avoiding environmental hazards, with some enemies throughout the game as an additional obstacle. There are coins to collect through the game, that contribute to the players score, as well as hidden collectables in each level.

## Game Overview

The player takes control of Sam the Pirate, as he wakes up one morning to find his ship wrecked, his crew mutinous zombies, and his precious treasure chest gone.

Sam is equipped with his pirate cutlass, to dispatch the enemies he encounters throughout his adventure. Coins are collected throughout each level that contribute to the players overall score, and Sam can find Barrels of Rum in each level, that act as a rare, hidden collectable and provide a substantial bonus to the score.

Levels are displayed on an overworld map that allows players to choose what order they want to progress through the game in. Completing a level acts as a checkpoint, and allows the player to come back to the game without losing their progress. If Sam dies during a level, that level must be restarted from scratch.

## Release Platforms

The targeted release platform is Steam Greenlight, and potentially via Green Man Gaming.

It could be given away free of charge, this is going to depend on what software/tools/libraries are used, and what their licenses are. This is completely fine, as at this stage of our careers, the marketing value and exposure from successfully putting a commercial quality game on a commercial platform while students is worth more than whatever money we could potentially make from it. It would also be good marketing for Media Design School, having students create a commercial quality product that is released on a commercial platform.

## Reasons for Joining the Project

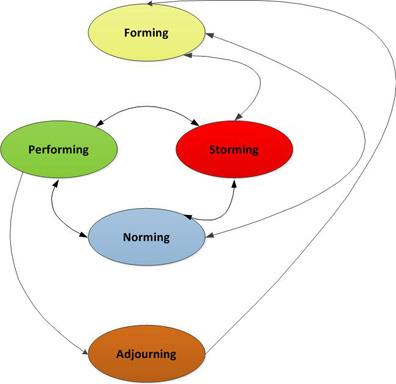
I have had the idea for this project in my head for over a year, really wanted to do it, even if that meant doing the programming side of the project myself. I want to make a polished, commercial quality product, and release it on a commercial platform.

# Team Management

## Team Creation

The process by which the creation of teams and projects were formed, began with a single directive; find a project that you want to work on and make it happen. This approach helped to generate teams and projects that people were committed/personally invested in developing. However, this took some time before the work flow became efficient and fully utilised, which usually manifested as time spent with conversions that were off topic half the time and the remainder was spent playing computer games (BattleField 3).

While this may have seemed like a waste of time I believe that it formed part of the Tuckman Model which involved the class getting familiar with each other and re-establishing familiar bonds within a relaxed environment.



**Figure 1**: Tuckman Model [1]

This eventually made the creative process more interactive with class members bouncing ideas off each other and providing suggestions enriching these ideas. Over time this began to increase the level of discussions and people began to gravitate towards the projects that they liked. The presentations at the end of the week helped to answer any remaining questions by other class members who were on the fence with which project they would like to develop. This also helped the lecturers see the state of each idea and the progress made thus far.

The class then had the weekend and the monday morning to finalise their decisions and form the teams. The development of the teams happened quickly and the groups began planning and improving the game from their original concepts.

## Team Members

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Roles** | **Justification** |
| **Ash** | Project Manager | * Currently has a high level oversight of the game requirements. * Has some experience managing other small projects. * Very organised. * Wants more experience in this area. |
|  | Lead Designer | * Is the original ideator and has the vision of what the game is going to be. * Has been thinking up the design for a whole year! * Is an open minded person and does not shutdown ideas. |
|  | Lead Programmer | * Has most experience developing 2D platformers. * Has some understanding of potential challenges and solutions. * Has already identified research areas. * Has already developed a suitable code framework for development. |
|  | Marketing Lead | * Has some experience developing a marketing campaign as part of curriculum. * Wants more experience in this field. |

## 

## 

## 

## Specialist Interest Area

My specialist interest area for this project is the overall game design, and business focused activities such as marketing and management. As I am looking to potentially start a game studio after graduation, or potentially during the later stages of the project, these areas are most interesting to me, rather than a programming specific area such as AI or networking.

### Research Capstone

I intend to research level design as part of my research focused capstone, as I believe it will be of the most benefit to this project. From a personal standpoint, I have no qualms with the development side of the project, I am fully confident that I am capable of getting the work done in the given timeframe. I am more concerned with how I am going to design the required number of levels in a way that they are are challenging without being impossible, and fun to play. Researching level design will assist with this when it comes to testing levels, responding to feedback.

### Development Capstone

Tools to assist with development shall be investigated and created for my development focused capstone. Currently, the main tool that will be developed is a tool that allows artists to see how their art and animations look in the game engine. This tool would also handle sprite sheets, and saving the relevant data ready to be loaded straight into the game.

# 

# 

# Project Planning

## Project Goals

* The team learns new development tools and methodologies.
* The game is fun to play.
* The game is published, or in the process of being published through the Steam Greenlight process by the end of May 2014.
* The game enables Ash to do a presentation on integrating Steamworks at next years New Zealand Game Developers Conference.

## Milestones

|  |  |
| --- | --- |
| **Date** | **Description** |
| End of Term 1 | All design documentation complete, including game design and an initial design for all 40 levels. |
| End of Week 4, Term 2 | Base rendering engine complete, including logging and input. |
| End of Term 2 | Platformer controls, physics, and UI framework complete. |
| End of Week 4, Term 3 | Level editor complete. |
| End of Term 3 | Initial art implemented, initial levels created. |
| End of Week 4, Term 4 | Steamworks implemented, all levels created and being iterated over through user testing, tools for artists finished or being created. |
| End of Term 4 | Game has no critical bugs, is polished and ready to be published. |

## Stretch Milestones

These milestones are on the project plan, but there are currently no plans to implement them. If production goes well, and it becomes possible to implement one or more of them, then they will be looked at in further detail. They are listed in order of priority/feasibility for a single programmer.

* Global leaderboards - may require external servers.
* Level editor with Steam Workshop integration - may require external servers.
* Four player cooperative play through Steam.

## 

## 

## 

## Marketing

As the game is planned to be released on Steam through the Greenlight process, there will need to be a fairly substantial amount of marketing work done in order to get awareness of the project out there, and hopefully encourage people to vote for the game on Steam. As part of the marketing campaign, the following should be actioned, or at the very least investigated:

* Get the Media Design School marketing team involved for most, if not all of the below steps.
* Write a weekly blog during production, possibly using a tool like WordPress. Post this blog to reddit, and other gaming related sites such as IndieDB.
* Be active on Twitter.
* Create and maintain Facebook fan page for the game.
* During the beta, talk to active streamers on Twitch.tv and YouTube to see if there is any interest in streaming the game.
* Invite Gameplanet and/or NZGamer.com in to do a feature on the development. This would not be restricted to just this team, other teams could get involved as well. Potentially look at other avenues such as the New Zealand Herald/Stuff.co.nz as well.

## Meetings

Meetings, when they need to occur, will be held on Friday afternoons at 2pm in the Level 19 Kitchen area. Agendas for these meetings will be emailed out to all involved on the Wednesday before.

Those involved in these meetings include, but are not limited to:

* Ash
* Paul
* Ivan
* Artists

# 

# 

# Research

## Design and Development Tools

### Project Management

The agile project management software known as Hansoft will be used for this project. While the small team size means that there probably will not be much project management to do, as Hansoft is used in professional game development studios, it makes sense from a learning perspective to learn it now, so to be better prepared for the work environment.

It will be run on a remote micro Amazon Web Services server, as this is free for 12 months, and will allow team members to access Hansoft from wherever they are working.

The Hansoft server is located at <http://hansoft.swashbucklinstudios.com>

### Version Control

GitHub will be used for version control, as it is free and widely used. This project is also an opportunity to learn a new form of version control, as the team has previously used TortoiseSVN.

The GitHub location for this project is <https://github.com/ash-c/pirategame>

### Programming Environment

Due to the school only having an academic license for the versions of Visual Studio installed, Visual Studio 2012 Express will be used instead, as it is completely free, and this would allow the final product to be sold commercially.

### Graphics and Input

SDL will be used to handle input and rendering as it is distributed under the zlib license, which allows commercial distribution. These have been chosen over using an existing DirectX framework, which the team is already familiar with, as it is cross platform, and allows the team to learn something new. It has also been chosen over using an existing game engine such as Unity, as Ash, the sole programmer for the team, has a personal preference for working at a lower level with C++ directly.

The artists will use Photoshop, and whatever other tools they may require, to produce the art. However this does bring up licensing concerns, as the school likely only has an academic license to Photoshop. It may be necessary to export the art assets in free image editing software such as Gimp.

### Physics

Initially the plan was to use an existing library such as Box2D to handle all the physics, however further research into the topic of physics for platformers has shown this to be not such a great idea. Pretty much all of the comments and articles on the subject suggested that using an existing library would work to start with, however in order to have expected platformer movement you would likely have to spend some time fighting the physics engine in order for things to work correctly. <http://www.learn-cocos2d.com/2013/08/physics-engine-platformer-terrible-idea/> This time would be better spent writing custom physics that can be debugged properly.

Instead, custom physics will be written, using Axis Aligned Bounding Boxes(AABB) as hitboxes. For platforms, only objects in tiles near the players location will be tested against, while for enemies and other objects, those visible on the screen will be tested against. Various tutorials and other resources on the internet will be used to assist with writing the collision detection and reaction code.

### Sound

FMOD is a premier sound engine, used in commercial projects around the world, however in order to release the end product commercially, even for free, requires a license. A casual license for a title with a budget of under $US100,000 costs $US500 for the first platform, and another $US500 for each subsequent platform.

FL Studio Fruity Edition can be used for creating music, it does not allow audio recording, but only costs $US99.

### Data Storage

Levels are represented as one single canvas, with the player viewing an area centered around Sam. Information such as each object's position and the object’s type need to be stored. All of this information will be stored as key/value pairs in a text file.

This information will be stored in the JSON format using one of the various open source JSON C++ interpreters. JSON has been chosen as it is a language independent data format, and the lead programmer and game designer has prior experience with JSON.

XML and INI files have been considered, however the team has a lack of experience with the XML format, and INI files have felt too simplistic for level data. INIs may be used in some other places.

## 

## 

## Market Demographics

The target demographic for The Adventures of Sam the Pirate is platformer fans of all ages. It is intended that anyone with an interest in platformer games may enjoy playing The Adventures of Sam the Pirate.

There are many popular platformers, both new and old, including Super Meat Boy, Sonic the Hedgehog, Commander Keen, Braid, Rayman Legends, Mark of the Ninja, to name a few. There are also some that have been successfully funded on Kickstarter.

* Mighty No. 9 <http://www.kickstarter.com/projects/mightyno9/mighty-no-9?ref=category>
* Legend of Iya <http://www.kickstarter.com/projects/523651724/legend-of-iya?ref=card>
* Tesla Breaks the World <http://www.kickstarter.com/projects/1660691250/tesla-breaks-the-world>
* Brave Bit <http://www.kickstarter.com/projects/1806980110/brave-bit-0?ref=card>

These successful Kickstarters and Super Meat Boy selling over 1 million units show that there is a consumer demand for platformers.

## Legal

Most of the following legal issues will be finalised later on in the project.

Academic licenses for some software may mean that the end product will not be released commercially through Media Design School. Assuming IP ownership is held or transferred to the team creating the game, a Kickstarter campaign could be run after the project in order to fund the licenses required for a commercial release.

The other significant legal issue is the ownership of the IP. This is originally held by the school, which would prevent any commercialisation of the project after the team graduates. If the school is able to transfer ownership to Swashbucklin Studios at, or before the team graduates then the project could be commercialised.

Any external contractors that are bought in to produce sound effects and/or music will need a contract, to ensure they are either paid for their work when the project is commercialised. If the project is not commercialised, then these external contractors would need to volunteer their time, as there would be no money in the project.

There is no requirement for a classification rating to release on Steam Greenlight, however it is something that should be considered. There should not be any problems with the lack of a rating as Sam the Pirate would likely have been given a G or equivalent rating.

### 

### Profit Sharing

With Media Design School as publisher, after Steam takes their cut:

|  |  |
| --- | --- |
| **Party** | **Share** |
| Media Design School | 20% |
| Swashbucklin Studios | 40% |
| Split between other. | 40% |

After graduation, with Swashbucklin Studios as publisher, after Steam takes their cut:

|  |  |
| --- | --- |
| **Party** | **Share** |
| Swashbucklin Studios | 50% |
| Split between other. | 50% |

## Social

One of the social issues that has been noticed during development is that all characters in the game are male. Given how male dominated gaming in general is, with the lack of playable female characters, and instances such as Anita Sarkeesian’s successful Kickstarter campaign (<http://www.kickstarter.com/projects/566429325/tropes-vs-women-in-video-games>) to do a series of videos on women in video games, this seems like an opportunity to provide a female playable character. Potentially, some of the enemy types could also be female.

The downside to having both male and female characters is that it is an increase in workload for the artists. If the gender of each enemy type is decided after concept art is done, then the extra workload should not be that high, as the only extra art would be for the second version of the main character. Optionally, the main character could simply be female.

Doing so does require the name of the game to be changed, as Tim is a male oriented name. Options for name changes include Sam, Alex, and potentially any other gender neutral names that are thought of in the future. The name of the game could also be changed completely, to remove all references to a character name, and allow the male and female character to be have their own name and identity.

Another potential issue is how playable the game will be for people who suffer from colour blindness. This can be remedied somewhat by designing the colours appropriately for the different types of colour blindness. For example, avoiding mixing red and green colours together.

## Ethical

As the target audience for the game consists of people of all ages, the depiction of violence in the game and whether it is appropriate for younger gamers should be considered. As the intended art style is lighthearted and cartoony, death and combat animations should follow suit and not portray a realistic version of violence at all.

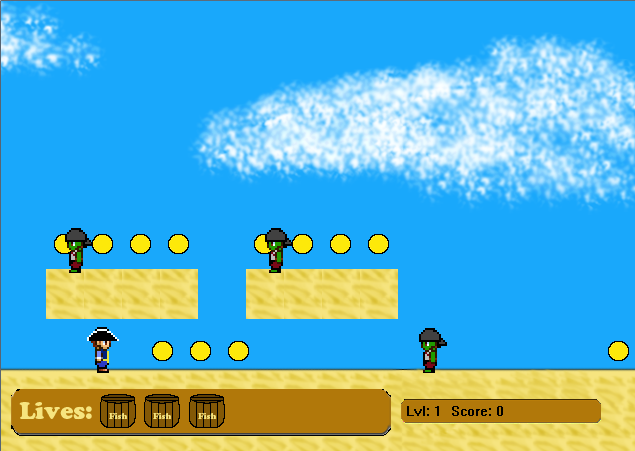
# 

# 

# Game Design

## Prototype

The prototype for The Adventures of Sam the Pirate was created in 2011, as part of a sprite game project using the Windows GDI framework. It is the inspiration for this project.



**Figure 2:** Screenshot of a beach level from the prototype.



**Figure 3:** Screenshot of a jungle level from the prototype.

## Narrative

Sam the Pirate wakes up on his ship, the Black Flag one morning, to find that his precious treasure chest which was stored under his hammock has been stolen! On his way to the deck, he discovers that his crew has turned into zombies! Upon reaching the deck, he sees his treasure chest in a row boat, being taken away by his former first mate, who is now leading the band of zombie pirates. These zombie pirates appear to be heading to a nearby island, so Sam heads over there on an adventure to get his precious treasure chest back.

## Gameplay

The focus of the game play is more on traditional platforming elements like running, jumping, and avoiding hazards, rather than combat. There will still be enemies, however combat with them is not the focus.

As such, Sam is equipped with a pirate cutlass for basic melee attacks in whichever direction he is facing. Any contact with an enemy or environmental hazard will kill him and force the player to restart that level from the beginning.

Rayman Legends is a good fit for what the game is intended to play like.



<http://www.technobuffalo.com/wp-content/uploads/2013/04/Rayman-Legends.jpeg>

## Score, Collectables, and Time

As they play through the game, the player earns points through various actions.

|  |  |  |
| --- | --- | --- |
| **Action** | **Description** | **Increase to score** |
| Pirate Coins | Scattered throughout each level. | 100 |
| Killing Enemies | Enemies scattered throughout each level. | 500 |
| Barrels of Rum | Rare collectables, a fixed number per level, some hidden, some more easily found. | 1000, Steam Achievement for finding the very first one, all of the ones in a level, and all of the ones in the game. |
| Finishing a level. | Time how long the level took. | Add more score for a shorter time, maximum of 10,000. |

Levels are timed in order to encourage speed, saving scores could also lead to doing global leaderboards based on score per level.

### 

### 

### 

### Level Timing

Level timing is displayed to users in the format *minutes:seconds*, with the pure seconds value being used behind the scenes to determine how much score is added. Examples:

1. A level takes 10 minutes(600 seconds) to complete. 10000 - (600 \* 10) = 4000, so 4000 is added to the players score.
2. A level takes 5 minutes(300 seconds) to complete. 10000 - (300 \* 10) = 7000, so 7000 is added to the players score.
3. A level takes 20 minutes(1200 seconds) to complete. 10000 - (1200 \* 10) = -2000, so nothing is added to the players score as the value is negative.

# 

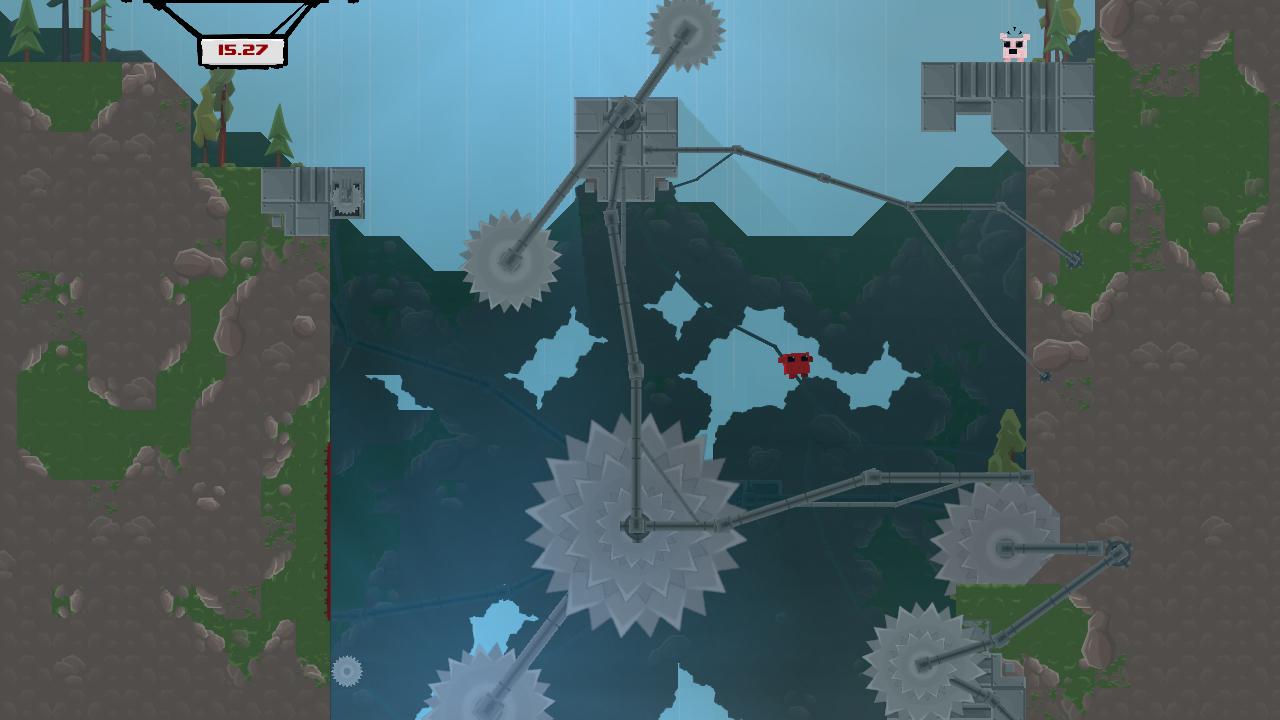
# 

# Art Style

The art style for the game takes inspiration from games like the newer Super Mario Brothers and Super Meat Boy. The intent is for it to be cartoony, using bright colours and a relatively simple colour palette. The opposite of a game like Braid, where the art is quite detailed. The Secret of Monkey Island Special Edition could also be used as inspiration, as it is a light hearted pirate themed game.

Different parts of the game should be coloured differently:

|  |  |
| --- | --- |
| **Object** | **Colours** |
| Player/platforms/helpful objects. | Warm colours. |
| Enemies/harmful objects. | Cool colours. |
| Backgrounds. | Dull, saturated colours. |



<http://supermeatboy.com/_media/meatboy/image/1269147440.jpg>



<http://gamersxtreme.org/wp-content/uploads/2012/12/New-Super-Mario-Bros-U-Gameplay-2.jpg>



<http://images2.wikia.nocookie.net/__cb20100507135312/tig/images/5/52/Braid-screen01.jpg>

## Art Layers

Each individual level will contain multiple layers of art that can move independently of each other. The closer layers will scroll more than than the further away layers.

|  |  |
| --- | --- |
| **Layer** | **Description** |
| Far background. | Far off in the distance. Potentially just a static colour, clouds, hill, mountains etc. |
| Middle background. | Medium distance objects, trees/hills, clouds, etc. |
| Close background. | Close objects, trees, bushes, buildings, etc. |
| Level objects. | Sam , enemies, level platforms, environmental hazards etc. |
| UI. | The user interface, menus, score, etc. |

## Concept Art

### Characters

player character - male and female

captain bwains - boss zombie

zombie pirates - male and female

zombie parrots - not a priority

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### Level Backgrounds



*Really rough mockup of a possible overworld map. Probably zoomed in more in the final product.*

inside/outside of a pirate ship

shipwrecks, ocean leading to beach

beach/sand

jungle



pirate fort

caves

### Environment Objects

carnivorous plants

swinging vines

rolling boulders

ladders?

spikes

swinging blades

torches + oil

spider webs

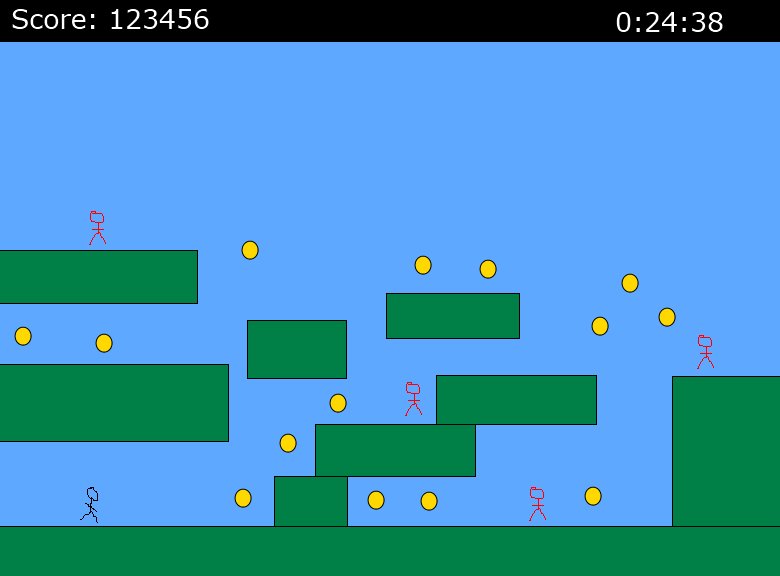
cannons

environment doodads?

### UI Mockups



*Mockup of the main menu.*



*Mockup of a level.*

## 

## 

## 

## 

## 

## 

## 

## 

## 

## Priority List

### Interface

* Must have
  + Logos
  + Title screen
  + Buttons
  + Fonts - Custom or pre-existing.
  + Dialogs

### Characters

* Must have
  + Player Character - Male
  + Basic Zombie
  + Musket Zombie
  + Captain Bwains
* Should have
  + Player Character - Female
  + Shield Zombie
  + Cannon Zombie
* Nice to have
  + Zombie Parrots
  + Variety within each enemy type, colours and gender

### Level Backgrounds

* Must have
  + Overworld map - needs to have a path.
  + Outside of pirate ship plus platform tiles
  + Beach/sand backgrounds - 1 plus platform tiles
  + Jungle backgrounds - 1 plus platform tiles
  + Pirate Fort backgrounds - 1 plus platform tiles
  + Caves backgrounds - 1 plus platform tiles
* Should have
  + Inside of pirate ship
  + Shipwrecks, ocean leading to beach
* Nice to have
  + A variety of backgrounds for each area
  + A variety of platform textures

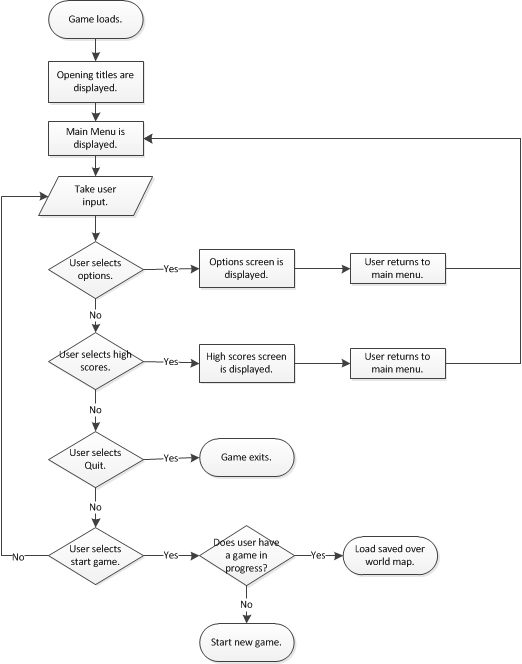
### 

### Environment

* Must have
  + Ladders
  + Swinging vines
  + Water
  + Spikes
  + Rolling boulders
  + Carnivorous plants
  + Quicksand
* Should have
  + Unstable platforms
  + Swinging blades
  + Spider Webs
* Nice to have
  + Torches and oil
  + Environment doodads

# Game Flow

## Menus



### 

### 

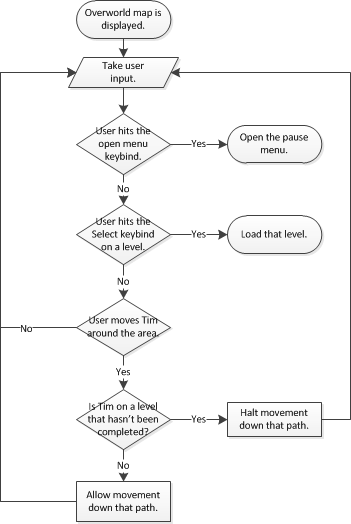
## Gameplay



### 

### 

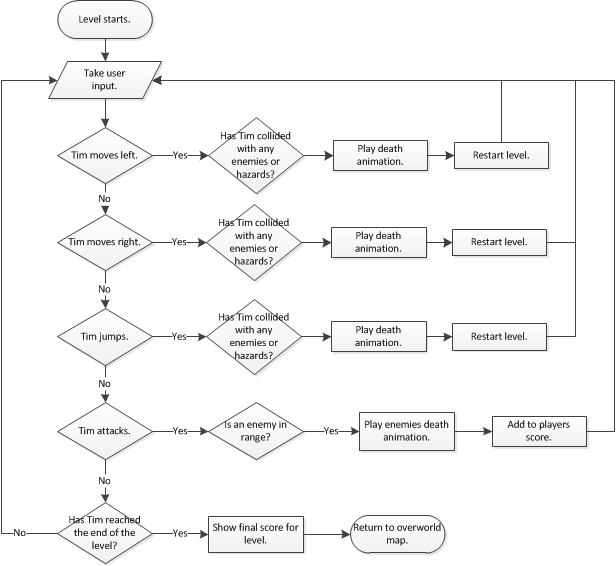
## Overworld Map



### 

### 

## Levels



## 

## 

# Enemy Types

Most enemies can only be killed with Sams cutlass attack. They all die in a single hit, in the same way that Sam dies in a single hit if he makes contact with any of them.

|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Movement Speed** |
| Basic Zombie Pirate. | Melee attack with cutlass. | Medium. |
| Musket Zombie Pirate. | Armed with a musket for ranged attacks, and with a cutlass for melee attacks. | Medium, stationary when firing. |
| Cannon Zombie Pirate. | Mans a cannon. The cannonballs have an area of effect. Fires in an arc. | Stationary. |
| Shield Zombie Pirate. | Armed with a shield. Cannot be killed as the shield blocks all attacks. Does not have an attack of his own. | Slow. |
| Zombie Parrots. | Melee attack. | Flyer, high speed. |

## Artificial Intelligence

The Artificial Intelligence for Sam the Pirate is very basic and follows some simple rules:

|  |  |
| --- | --- |
| **Event** | **Action** |
| Spawn. | Stand still. |
| Player camera moves close. | Start moving left. |
| Hit a wall or a ledge. | Turn around. |

## 

# Level Design

Levels will have be 10,000 pixels wide by 2,000 pixels high, with the players screen essentially acting like a camera onto the level. Thus a smaller screen or different aspect ratio will display a different amount of the level.

The overworld map will be built in the same kind of fashion as the levels, but with a top down view. Movement will also be far more restrictive, with the player only being able to move along set paths.



<http://forum.unity3d.com/attachment.php?attachmentid=63834&d=1376526627>

The overworld map provides a non-linear path through the game, allowing players to choose the levels they wish to beat to progress through the game.

## Boss Encounter Levels

Boss encounters happen every 10 levels, at level 10, 20, 30 and 40. They are designed similarly to regular levels, however the camera will permanently be scrolling towards the right, in order to force the player to move. Essentially these levels will act like a race, the player has to either run from something or attempt to catch something, without being killed by any enemies or environmental hazards present in the level.

## 

## Level Progression

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Levels** | **Theme** | **Enemies Introduced** | **Hazards Introduced** | **Other Elements** |
| 1 | Ship. | Basic Zombie Pirates. |  |  |
| 2 | Ship to Beach, shipwrecks. |  | Water (Sam can not swim). | Moving platforms. |
| 3-10 | Beach. |  | Quicksand (Slows movement, kills if you don’t get out of it). |  |
| 11-20 | Jungle. | Musket Zombie Pirates. | Carnivorous plants. | Swinging vines. |
| 21-30 | Pirate Fort. | Cannon Zombie Pirates. | Spikes, swinging blades, unstable platforms. | Ladders. |
| 31-40 | Caves. | Shield Zombie Pirates. | Falling boulders, lava, spider webs, torches and oil. |  |

# 

# 

# Controls

Input is open to change based on feedback from testing. All inputs are also customisable.

## Menus/Overworld

|  |  |  |
| --- | --- | --- |
| **Action** | **Keyboard/Mouse** | **Controller** |
| Change selection/Move Sam. | Mouse movement, Up or Down arrows | Up or Down on the D-Pad or Left Analog stick. |
| Select item. | Enter. | A button on XBOX controller, Cross button on PSP. |
| Cancel. | Escape. | B button on XBOX controller, Circle on PSP. |

*Note: If the player uses a controller or the keyboard to navigate menus, hide the mouse cursor.*

## Levels

|  |  |  |
| --- | --- | --- |
| **Action** | **Keyboard** | **Controller** |
| Move Left | Left Arrow | Left on D-Pad or Left Analog Stick |
| Move Right | Right Arrow | Right on D-Pad or Left Analog Stick |
| Jump | Up Arrow | A button on XBOX controller, Cross button on PSP. |
| Attack | Spacebar | B button on XBOX controller, Circle button on PSP. |
| Pause/Open Menu | Escape | Start on both controllers. |

*Note: Always hide the mouse cursor in levels, only display it if the menu is open and the mouse is active.*

# 

# Level Editor

The level editor will be built using the game engine. Its layout will be user friendly and designed with the consumer in mind, rather than for developer use only, as one of the stretch milestones is to release the editor with the game and Steam Workshop integration.

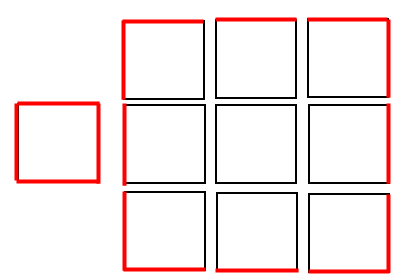
A tile based system will be used, with each level being broken up into squares. Various tools will be used to create platforms and place objects.

* Platform/wall creator.
* Environmental hazards.
* Enemies.
* Other doodads.

The only real differences between each theme will be the visuals, all objects and enemy types are available for each theme. This is being done for two reasons:

* Iterating over the level design later on may find that enemy types can or should be introduced earlier/later than currently planned.
* Future proofing, in case the editor does get publicly released with steam workshop integration.

Platforms and walls can be created by choosing the appropriate tool and “painting” over tiles by clicking and dragging the mouse cursor around. The editor will then automatically determine what type of tile piece, from the below options, is required.



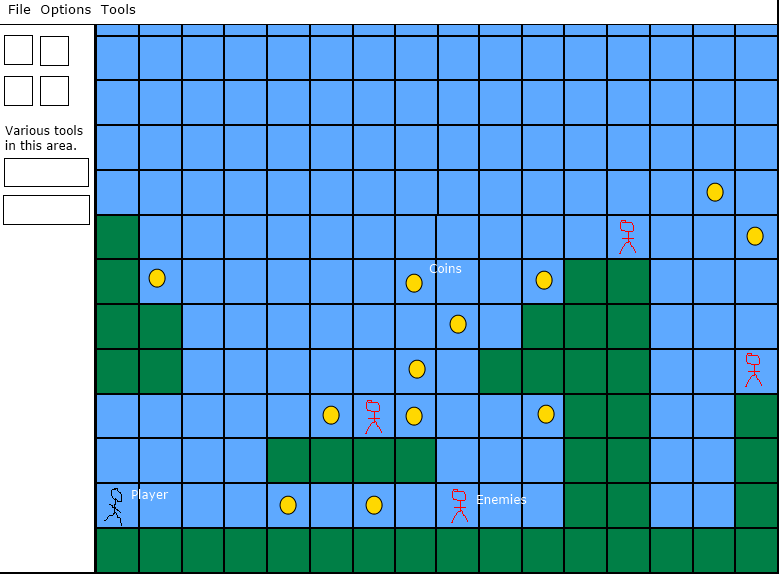
*Tile pieces, with red lines indicating an edge.*

Edges will line up with each other in such a way that different pieces can be placed next to each other in different configurations and visibly line up without any weird visual artifacts. Edges may be separate, graphically, from the backgrounds, in order to handle certain all corners better.

The following information will be stored for all the objects in a level:

* Its position in the level.
* The type of object it is.
* The theme for the object.

## Mock Up



*Visual mockup of the level editor.*

# 

# Audio

## Design

The audio for Sam the Pirate will follow a similar design as the art style. Light hearted pirate theme, possibly tending towards darker in the later levels of the game, to reflect the progression down into the caves the pirate fort is built on. The opening music from the prototype is a perfect example. The Secret of Monkey Island also contains music that could be considered as inspiration for the music in Sam the Pirate.

## Sourcing

### Music

Background music tracks for the game will be sourced from any of the following sources:

* Kevin MacLeod, at <http://incompetech.com>. Kevin has many pieces of music on his site, that are free to use under a Creative Commons license.
* Attend the monthly Auckland game developer meetups and put word out that we are looking for people to provide music and/or sound effects.

### Sound Effects

Sound effects for the game will be sourced from any of the following sources:

* <http://freesound.org>
* The BBC Sound Library, if purchased by the school.
* Using free software such as Audacity to create sound effects from scratch.

# 

# 

# Design Assumptions

* There is a market for 2D side scrolling platformers.
* Mild cartoon violence is acceptable for young kids.
* Artists will provide commercial quality art.
* Ash can complete the game design and programming portions of the project by himself.
* Media Design School will cover the cost of any licenses.
* Media Design School will transfer ownership of the associated IP to Swashbucklin Studios so the project can be commercialised after graduation if the school is unable to cover license costs.
* Assumed physics library will work for a platformer.

# 

# 

# Technical Design

## Design Goals - The Papyrus Engine

* Component based.
  + Objects will only have what they need. If an object in the game needs a sprite, it has access to a sprite component, if it needs physics, it has access to a physics component, etc.
* Reusable modules.
  + Every module in the engine should be capable of being reused outside the engine. Each module is also behind a namespace.
* Cross platform.
  + By using SDL, any game created using the Papyrus Engine will be capable of being released on platforms other than Windows.
* Everything is interfaced.
  + Implementations for everything will be behind an interface. This should allow for the implementations to be swapped out without effecting anything else.
* As few dependencies between modules as possible.
  + This leads to modules that can be reused more easily, which is a design goal.
* As little class inheritance as possible.
  + To reduce complexity and make debugging a bit easier.
* Engine code independent from game code.
  + Should be able to easily reuse the engine for other games.

## 

## 

## Overview

The Papyrus Engine is based on an older DirectX 10 framework that was used for previous projects, with several changes. The biggest change is that SDL is being used as implementation for most modules. Other changes include some manager classes, no longer existing as singleton classes. Instead, they simply exist as a collection of static functions and data structures behind an appropriately named namespace.

## Unit Tests

Unit testing will be built into the Papyrus Engine from the start of production, using the Google Test framework. <https://code.google.com/p/googletest/>

## Rendering

The video component of SDL will be used to handle rendering. Any other module that wants to display to screen will depend on this rendering module.

## Input Events

The input component of SDL will be used to handle input events. This will allow the Papyrus Engine to handle mouse, keyboard, controllers, and joysticks. Any objects that want to know about input events will need to register with an event handler, so they can then be notified when the event happens. Essentially an observer pattern will be in use.

## Logging

Three types of logging will be available through the logging module:

* To screen.
  + Logging to the screen will require a pointer to a renderer interface with a work implementation, if none is present then this functionality will not work.
* To file.
  + May require a dependency on the Text File Parsing, or could be written directly into the logging module.
* To a debug console.
  + Debug console can display some other logging information, perhaps less important can be written to a debug console instead of being displayed to the screen. It can also be used to enter commands, such as turning debug display on/off, skipping to certain levels, etc. Also has a dependency on the renderer module.

The debug console may not be present in a release build of the game.

## 

## 

## Text File Parsing

This module will read in text files that contain key/value pairs. Current supported formats include JSON and INI. It is an upgraded and fixed version of an INI parser that was part of an old DirectX 10 Framework. This module will be developed during the Game Engine paper.

## Sprites

The sprite module will be responsible for loading and deleting sprites. It will need to handle both static and animated sprites. Animated sprites will use frame based animations, with all the animations for a character being stored in a single spritesheet. Current plans are for animated sprites to consist of six frames, with each frame being 100 pixels by 50-75 pixels.

Spritesheets will be cached, so that if multiple objects are using the same sprite, the spritesheet is only loaded into memory once. Objects requesting a sprite from this module will receive a pointer to a sprite interface, allowing them to update positions etc.

## Parallax

The parallax module will be responsible for handling each of the background layers. This includes creating each layer, determining when/what objects to draw (potentially set through scripting). The number of layers, their order, and the contents of each layer will all be defined externally by INI files.

## Physics

Custom platformer physics will be written for Sam the Pirate, based around Axis Aligned Bounding Boxes, or AABBs. Every object on the game play level will have an AABB, these boxes, if necessary, could be smaller than the visible sprite. This would potentially help reduce instances of the player dying when they don’t appear to be near anything that could kill them, and so provide a more pleasing experience visually.

In order to avoid doing collision detection against every object in a level, the players AABB will only be checked against objects that are being displayed on the screen. For detection against platforms, as levels are tile based, only tiles within a certain area around the player will be checked against. Perhaps two tiles in all directions. Nothing that is not visible on the screen will be tested against.

## Sound

FMOD will be used to handle audio in Sam the Pirate, with the sound module acting as an interface to it. This will allow for the implementation of the sound module to be swapped out, if for example we cannot secure FMOD licensing it could be swapped out for the audio component of SDL.

Similarly to the sprite module, sound files will be cached in such a way that if multiple objects are using the same sound effect, it will only be loaded once.

## User Interface

This module will handle all elements of loading, displaying, and interacting with the end user interface. This includes creating, loading, and displaying both static and animated images, buttons, and any other UI elements. Font spritesheets will be used for displaying text.

## Font

The font module will act as an interface to the SDL\_TTF library, which will be used for creating and maintaining fonts for use in displaying debug text to the screen.

## Installers

Installer tools will be required for creating publicly releasable packages. This will require further research later on in the project, but so far there are a couple of options:

* IzPack - open source, cross platform tool for creating installers. Requires Java however. <http://izpack.org/>
* Install Simple - free tool, but creates installers for Windows only. <http://installsimple.com/>

# 

# 

# References

[1]: **Title**: Group Dynamics and Project Teams, **Author**: Kathy L. Maschke, **Date Accessed**: 26/08/2013, **URL**: <http://www.kennesaw.edu/businessservices/Sept2011News/PMO_001.html>