Doodle Dash (Preliminary Title)

Software Project Proposal

By

Candice Academia

Johnny Chi

Nicholas Ruiz

Robert Steiminger

Stephen Sing

San Jose State University

Spring 2017

Dr. Ramin Moazeni

2 March 2017

Description of the Product

The final deliverable will be a simple mobile phone game. The idea of the game will be simple. All the user has to do is tap the screen in order to jump. Horizontal motion will be set, as well as the levels. The game will be more in the vein of something like Geometry Dash, compared to Flappy Birds. The difficulty of the game will come from timing the jumps in order to avoid obstacles. The rhythm of jumps will ideally be synced to music or create a rhythm.

Geometry Dash is a rhythm based running game that was developed in 2013. Geometry Dash reached international fame and topped the iTunes and Android store. Known for its simplistic controls that were easy to pick up but hard to master, it would go on to receive the most popular paid iTunes app in 2014.

The main difference between our game and Geometry Dash will be that our game level design will have a music element to it that Geometry Dash does not have. In Geometry Dash, the level design is separate from the background music that accompanies the level. With our game, we will strive to match obstacles with specific beats within the song and craft the platforms of our level based on the tempo and rhythm of the song that accompanies each level. With our game, we will also incorporate a variety of different genres of music that will help to create a new and different feel for every level which will also appeal to a wider range of audience with varied musical tastes.

The game itself will be themed off of like a pen and paper type of sketch style of art. The main player avatar will be a circle, with a user uploaded image inside of it. This ball will just roll around and jump as well as the other mechanics we decide to add.

Depending on the scope, there also are plans to implement a social leaderboard where users can share the highest level they have completed. For the release, we are planning to have 5 fully fleshed out levels, each adding a new mechanic.

Need for the Product

There is an ever constant need for new games for entertainment in the market. With stress making daily life challenging, some people deal with that daily stress by taking some down-time to play games whenever they can.

Currently in the market, there is not much competition to Geometry Dash when it comes to difficult auto-running platformers. Hopefully this application will be able to compete with it.

Since our game is going to be on a mobile platform, it will be easy to play wherever and whenever the player wants to (given that it will not be life-threatening).

Potential Audience

The main audience will definitely be the millennials. They are going to be the group that is always looking for something new and exciting to try out. The millennials will also be the most techy, they are the ones that are looking on the app store, when they are bored, and trying different games out. This will be a large group to target. This audience group will have the most access to smart phones and be pretty good about using technology.

Younger kids are sort of a subsection of that group and they will also be a potential target audience. Our plan is to have the art scheme be very childish and young, yet clean. Kids might really enjoy this type of art style. The gameplay is also simple enough for young children to pick up. The younger kids technically are considered millennials, but they probably will not be as proficient with technology compared to the millennials that are in their teens.

Another potential audience group will be the working class, middle-aged user who will typically be holding an office job. This group, depending on what field they work in, may or may not be as technology proficient compared to the younger generation. We do believe that our game can cater to them just through the ease of the controls and easy-on-the-eyes art style.

The gaming audience will be another large potential group of users. We would want to appease a multitude of different gamers, ranging from the super hardcore gamers to the more casual gamers. This audience will be the most technology proficient out of our target audience.

Since the objective of the game is fairly simple, quick and easy to pick up, we have the potential for a large scale, global reaching audience. Almost everyone has a smartphone device and some downtime which allows for them to quickly pick up and play this game. If we integrate a social leaderboard the potential for reaching a broader audience is increased.

Discussion of Competing Products

There are many Unity games that currently exist in the market and it will be very hard to compete against them if we do not have a game that stands out. Our game will be similar, in that it will be a 2D sidescrolling game, but we will make it different by being simple to play, allowing customizable icons, sharing high scores to Google Play Games, a unique artstyle, catchy music selection, and adding different dynamics that do not exist in the other games.

High-level Technical Design

To make our game, we are going to use Unity. The main reason for this is mainly because of our desire to learn how to work with a tool that is being used in the industry. We are going into it pretty blind, with none of our group members having any experience working with

Unity. Because this is a project that all of us have a shared interest in, we have a huge motivation to learn Unity as well as work on the completion of the project.

Multiple members in our group have had experience working with GameMaker Studio. We wanted to work with something a bit more robust with more support and libraries. Unity is one of the staples in industry and that led us in deciding to using it. One of the main functions that make it great for developing games is the ability to work on scenes. Unity also has great support for adding scripts to objects, which will be key in this game, mainly for making sure we can get collision pixel perfect. Both Unity and GameMaker look like they have strong functionality for level development, which for us, will be a huge part of the polish.

The functionality and backend design of the code will be relatively simple, this is to allow us to spend more time on polishing the game. For a mobile game application, the refinement will be what really sets it apart from other games. Unity offers a very powerful tool for handling animations, which will play a huge factor in how all of our sprites will come out.

For leaderboard and cloud use however, we will need to look into cloud storing options. We can look into AWS and other cloud storage options when it comes to the leaderboard and social network aspect. We could have all of our code we need for social networking be held on the cloud and just query the server.

As a standard, we will have animation/frontend, and backend divided among team members as their main roles but it would not be limited to those. We will also have a dedicated level design person and a person dedicated to handling the cloud.

Core Features

- Intuitive Level Design
    - Levels will be premade and level design will match with accompanying music per each level.
- Responsive Controls
    - No lag for precision controls which are essential for level mastery.
- Customizable Settings
    - Allows users to upload images to serve as avatars in game and also allow users to control audio options as well as allow sharing of game scores.
- Interactive Menus
    - Allows users to customize their avatar, view their progress and different levels, check leaderboards, and adjust settings.
- Local leaderboard
    - Create a leaderboard which allows users to check their fastest time on completed levels and other various statistics.

Backlog Features

- Online leaderboard
    - Allows users to connect with an online leaderboard database to compete for best level times.
- Social Media Connectivity
    - Allows users to share their scores with friends through social media including Facebook and Twitter.
- Ghosting Friend Challenges
    - Allows users to race along a translucent avatar of their friend's fastest times in order to create a multiplayer experience.
- Level Creator
    - Allow users to create their own tracks with a predetermined set list of music and share it online for others to play.

Resource Requirements

Time and extra effort for team members are needed since we are building this game from the ground up. Assuming that no one has the expertise in using Unity per se, it will require effort in researching how we will be able to utilize what resources it could provide as well as integrating other libraries that would be the most optimal way to develop our game application. Knowing that each team member has their own other projects and classes to attend to, synchronizing schedules (i.e. division of work and work dependencies) will be an obstacle as well.

Other areas wherein extra time and effort is demanded will be to port the game from Unity to Android, as well as in compressing audio and visual assets in order to fit the hardware of the devices we will be developing.

A safe rough estimate would be to expect the project to be operational within 200-250 man hours combined. To have it ready for release, maybe another 100 man hours spent on refinement.

Potential Approaches

Divide the team into starting building the framework of our product and the research team to determine if what we want to achieve is even feasible in the given time frame of this course. After that, reevaluate our approach and see what we could do as an alternative product with the same concept in mind.

After creating the base framework of the project, each of the pages can be separated for each team member to work with. The problem with this however, is that the pages could feel disconnected because it will be a different person working on different parts of the application.

Another approach is to have different individuals work on the backend, the profile, the database, and the frontend. This will allow each part to have continuity since each part will be worked on by the same individual.

A problem that may be encountered will be accountability with team members. With this approach, it is mandatory that development must meet a strict schedule with definite deadlines in order for production to continually work as some team members may rely on another team member's parts of the project to complete their work.

Assessment of Risk

Since everyone has not worked with Unity before, learning it would require huge overhead to learn the software and implement our game. Thus, extra hours for solely understanding how everything works would be needed.

Another obstacle may be with porting the project to Android. Although Unity provides integration with Android Studios it is somewhat complex and under development which is also another area that our group does not have expertise on. In order to successfully complete the integration, we will need to either outsource the work to a knowledgeable third party or spend additional time familiarizing ourselves with the frameworks required.

On top of that, since no one has expertise in this software, we would not know as we are building if we are guaranteed a working final product or if we will even be able to complete our implementation given the time constraint we have for this class.

To reduce this risk, we need to do the above mentioned approach. Also, we should have a backup product plan.

Next Steps

Meet up and decide on the roles each team member wants to be in charge of as well as deciding on the overall flow of our application development. Begin to research additional software or libraries that can help us develop the mapping framework and provide cursory data that we can expand on in the development cycle. Be more proactive in communicating through Slack - especially since everyone has their own schedules and not all team members are in attendance during meetings.