Steven Shaw

Interactive Digital Media

May 15th, 2013

Project 2: Flash Game of AWESOME

1) Changes in Game Design

1.1) Story

My original idea for the game was to have a thought out story with a beginning and end. However, as I began to develop the game, that idea went away. There is still an introduction to the story to set the theme for the game, however, it is only there to tie the game together. I believe the game flows quite well and therefore I decided to not build up the story and risk messing up the flow of the game.

1.2) Platform

When I originally came up with my idea for this project, I had intended it to be developed for mobile devices. I was going to use the accelerometer capabilities of tablets for the player's movement and have two buttons to allow the player to fire bullets.

I decided that I should focus on the game play for the desktop version first, before starting the port to mobile. This lead to the mobile port not getting implanted due to time constraints and game play problems. I still intend on porting this project to tablets after the course has ended.

2) Implementation

2.1) Screens

The game currently has the following screens:

- Menu: First screen that the player encounters. This screen has buttons that take the player to the instructions, high score,, credits, and game screens.

- Instructions: This screen has useful information on how to play the game. It also contains the short back story that the game universe it set in. In addition, it contains buttons to go back to the menu screen, or to accept the mission which will start the game.

- High Score: The high score screen contains the top 10 high scores that any player has achieved. The scores are listed in descending order. This screen also contains a button that takes the player back to the Menu screen.

- Credits: The credits screen sites the various third party sources that I used when developing the game as well as a special thanks to two of my friends who tested the game during development and gave me feedback about what to change and improve.

- Game: The game screen is the main screen for the game. It is where all of the game elements are located. These elements are dynamically created at

runtime. This screen shows a heads-up-display (HUD) that displays the player's score, lives, current level, and bomb count.

- Score: The score screen is shown to the player after the player has run out of lives. It shows the player his/her score, and gives them the option to add their score to the high score list by typing in their name and clicking the "Add Score" button. In addition, it shows the top ten scores that were recorded in descending order. This screen also contains a button to take the player back to the main menu.

2.2) Game Play

2.2.1) Overview

- The game is pretty simple to play. The player, controls Tux, the Linux penguin, in order to defeat hordes of Apple and Microsoft themed enemies while trying to collect as many yellow point tokens as possible.
- The player moves with WASD, shoots bullets with the left and right arrow keys, and deploys bombs with the space bar.

2.2.2) Animations/Artwork

- Since my original intent was to develop a game for mobile devices, my game has limited animations.
- Each token, oxygen and points, have a simple shape tweens to give the illusion that they are spinning.

2.2.3) Mechanics

- There is a bullet/bomb delay so the playing may not spam the fire buttons
- Tux can fly off of the top of the screen, into outer space, and die.
- Tux's oxygen supply decreases every second, thus requiring the player to pick up the blue oxygen tokens to prevent Tux from suffocating.
- Each level consists of a certain number of enemies, each with a certain difficulty.
- Apple enemies are the main type of enemy, with Microsoft enemies appearing every 4th level.
- Microsoft enemies are noticeably more difficult than the Apple enemies. They fire faster and spawn two child enemies when they die. Killing a Microsoft enemy awards more points as well.

- They enemies move faster with each level, capping out at the 5th level.
- Both Tux and the enemies cannot kill themselves with their own bullets.
- The player is awarded a bomb power-up every certain number of points. The starting point amount, passed in through XML, is 100. This number doubles every time a bomb is obtained.

3) Class Hierarchy of the game

- XML is loaded by Document and passed to GameScreen
- Document uses ScoreManager, GameScreen, and SpaceBackground
- GameScreen uses PlayerLayer, HUDLayer, MicrosoftEnemy, AppleEnemy, Token, Bullet, and BombSound
- PlayerLayer uses Player, GameScreen, and KeyBoarder
- AppleEnemy, MicrosoftEnemy extend Enemy and use GameScreen
- Bullet uses GunShot
- The PlayerLayerMobile and HUDMobileLayer are used by GameScreen but are never called (Mobile not implemented).

4) Technical Issues

4.1) Realistic Moon Gravity

As my game takes place on the Moon, I needed to implement realistic Moon gravity and jet pack acceleration. This was a challenge for me as physics is not my strong suit. To solve this, I enlisted the help of one of my roommates who excels in physics. I had him explain many differ rent equations to me. After the explanation, I knew what I had to do in order to implement realistic Moon gravity and jet pack acceleration.

4.2) Lacking Artwork and Animation

My artwork and animations are severely lacking. This is due to the fact that I was spending most of my time developing the game play in such a manner that it could easily be ported to mobile devices. I was very hesitant with certain game play features. This hesitation lead to the artwork and animations getting put off till the later portions of the project.

4.3) Player Bullet and Bomb Delay

When I was implementing the player-keyboard interactions, I kept running into a problem when the player was either deploying a bomb, or shooting a bullet. The problem was that when the player hit either the space bar or the left/right arrow keys to shoot or deploy a bomb, it they would shoot two bullets or deploy two bombs with one push of a key. This was due to the fact that my implementation did not take into account key duration and was performing the same action multiple times in less than a second. To fix this, I implemented two timers, and bullet

and bomb counters. The counter simulated a 1 second delay in deploying bombs or shooting bullets.

5) Future Work

- Port to mobile devices
- Improve artwork and timeline animations
- Add new enemies
- Add new power-ups and player abilities
- Add new enemy abilities
- Add informative messages to increase game feedback to player.

6) External Sources

- High score manager code
  http://zanuzawa.blogspot.com/2012/10/making-highscore-using-sharedobject-in.html

- Asteroids code from class
  http://igm.rit.edu/~acjvks/courses/20123/330/outline.html

- Sound Effects
  http://soundbible.com/
  http://ningmp3.com/

- Microsoft/Apple/Linux Logos
  http://www.apple.com/
  http://www.microsoft.com/en-us/default.aspx
  http://www.linux.org/