Feasibility Study

The Group:

Jennifer Green, [Jennifer\_Green@student.uml.edu](mailto:Jennifer_Green@student.uml.edu)

Hansel De La Cruz, [Hansel\_DeLaCruz@student.uml.edu](mailto:Hansel_DeLaCruz@student.uml.edu)

Kyle Jolicoeur, [Kyle\_Jolicoeur@student.uml.edu](mailto:Kyle_Jolicoeur@student.uml.edu)

GitHub Repository: https://github.com/hanselrd/bubble-warrior-adventures

The Client:

RPG fans

The Task to be Undertaken:

The project is to develop a 2D, top-down RPG similar to older games such as Final Fantasy VI and Legend of Zelda. The player will control a single character who journeys back to their home kingdom in an attempt to correct a mistake they made ten years ago. Along the way, they will gain access to stronger weapons and armor, as well as special magical orbs known that bubbles that can confer great power on their wielders.

Designing the game will consist of four main parts: creating and implementing a variety of maps for the player to explore, integrating the story in to the world, and the creation of both a battle and inventory system.

Benefits:

Everyone likes a good 2D RPG, and by using open-source assets we leave us the possibility of selling the game at some point in the future. This program is just for entertainment of the user.

Preliminary Requirements Analysis:

1. Main display
   1. Background
      1. Title / option backgrounds
      2. Map background
         1. Load in maps from files
         2. Load collision (not walkable) areas from map file
         3. Player view
   2. Sprites
      1. Load sprites / textures from files
      2. Player
         1. Animate player movement
         2. Animate player attack
         3. Collision detection
      3. Friendly NPC’s
         1. Text / Lore only NPC’s
         2. Story progressing NPC’s
         3. Currency shop NPC’s
         4. Collision detection
      4. Hostile NPC’s
         1. Animate NPC’s
         2. Casual enemy NPC’s
         3. Boss enemy NPC’s
         4. Enemy projectiles?
         5. Collision detection
2. User Interface
   1. Display interface
      1. Title screen
         1. New game
         2. Load from save
         3. Options
      2. Options screen
         1. Audio options
         2. Resolution options
         3. Full screen/ windowed
      3. Inventory
         1. Equip-able
         2. Slots
         3. Trash items
      4. Main gameplay overlay
         1. Resources (health/mana?)
         2. Currency
         3. Player name
         4. Player level / experience
         5. Player class
         6. Inventory button
   2. Controls
      1. Player movement
      2. Player attack
      3. Menu shortcuts
      4. Saving and loading the game
3. Collaboration
   1. Github
      1. Code sharing
      2. Version control
      3. Code backups
   2. Gitter
      1. Idea sharing
      2. Code issue resolution
      3. Planning for deadlines

Technical Requirements – Feasibility:

1. Images – As of now, all images, tile-sets, and audio will be used from OpenGameArt (website) which can be used royalty-free.
2. Database – Python will store scripts, AI. Possibly will use JSON for saving and loading the game. Tmx file type will store map data including collision areas.
3. Libraries – SFML will handle events, and window display. TGUI will be used for overlaying the main gameplay window, displaying inventory, and for title screen / options windows. Python will store scripts, NPC AI,
   1. Data storing
      1. Python for scripts and NPC AI
      2. Tmx format for map / tile files
   2. multimedia
      1. OpenGameArt website for images, tile-sets, and audio
      2. Tiled map maker / map generator

Scope:

Our goal for this program is to make a fun story and action driven game. We will have currency and shops, battle, player stats, equip-able items and stat changing items, a level up system, upgrading items, enemy NPC’s.

Suggested Deliverables:

Management:

Technical:

Walk-Through:

Software Development Process:

Outline Plan:

Visibility Plan:

Business Considerations:

Risk Analysis:

Conclusion:

Load in map files from Tiled

Use Python for scripts

Inventory system drop, pick up, sort, etc vector

Shops

Collision detection, knows where character can and cannot go

Story and cutscenes

GUI

Compiles on Windows, Linux, and Mac

Subclass for scripts, NPCs, UI, enemies, etc