Anthony Hsu

Project 2

Design Documentation

**Summary statement:**

To create a gift for my players/ immortalize a moment from my first time DM-ing/ to share this hilarious story with my peers/classmates

I want to create the story of the “Bear Trap Bear-trap” for the viewing pleasure of my peers, DnD players, and myself. It is the story of My Party coming upon a cave with a valuable looking object hanging in the ceiling stuck in the center of a set bear-trap hanging upside down on a chain from the ceiling. On the back wall of the cave is a lever in the upright position and beside it are 2 stone statues. The party persuades the least favorite of the party to go in and pull the lever. Then…. A bear pops out of a dimension door that opens up directly under the hanging upside down bear trap and surprise attacks the poor party member that pulled the lever. The lever resets to the upright position. The whole party engages in combat with the bear. The wizard, who that session added enlarge person to their learned spells, enlarged our 7ft tall barbarian who then proceeded to fail a reflex save and grow into the bear-trap which promptly chomped on her head. The 1 druid abstained from combat and watched the hilarity ensue as a mad bear is in a cave beating the crap out of a ranged only character and a barbarian who is tethered to the cave ceiling and struggling to undo it………… plot stuff……. After the bear is defeated the party considers the fact that the lever reset and played with the idea of pulling it again to see if they could turn the encounter into an experience farm. They also considered building a tamed bear army since they had a druid and the main plot was not time sensitive.

**Technical Deployment Document:**

I will be using Ren.py which will take care of the majority of the back-of-house programing. My project will only be usable on computers that are Ren.py compatible. The use of a touch screen or point-and-click on a traditional monitor are the only major ways of playing. The ways in which user would be using the computer or tablet- laptop would be sitting, laying down, or standing.

**User Personas:**

My design process is focused on content quality and less visual effects since Ren.py standardized the format of the visual layout. In particular I will be making the content appropriate for my peers in a classroom setting as required for the project. I can include much more geeky content if I see fit because I know they will understand it. I will also be designing my content with my friends in mind so that. Inside jokes may be tossed in here and there and I will be making it in a way that they can play through some alternate plot endings to the encounter. Lastly I will have to consider my family and non-geek friends. I will need to have enough “normal” intuitive plot for someone to understand without any DnD experience to enjoy and also will be encouraging them to look into DnD through the project.

**UI/UX Specifications**

Ren.py has a preset color defaults and a layout for text, images, and selection buttons. The text box is black and text is default white but can be changed. The contrast is built-in to Ren.py for the interactive elements. The layout is repetitive so that a gamer will eventually intuitively know where to find certain information on the screen. Or that certain color text has a specific meaning. Ex: red text on a selection will lead to a combat or negative path and green will lead to a peaceful path. The text is aligned top left in the bottom 40% of the screen within the text box of Ren.py. The alignment of images may change to denote characters moving through the screen. The proximity of text is standardized with about a 1.15 line spacing and is semi-compact to accommodate fitting about 6 lines of text. Ren.py by design is made for large amounts of text and I can just place more on the next frame.

**Screen Wire Frame**

Background Image

Selection Pane

Avatar Ctr

Avatar Left

Avatar Right

Name

Text

**Screen Flows:**

Navigation is simple point-and-click. There are Ren.py navigation options preloaded into the interface such as skip and auto click.

**Typography:**

Ren.py supports TrueType/OpenType fonts and collections, and Image-Based fonts. Ren’py uses DejaVuSans font.

Sourced: <https://www.fontsquirrel.com/fonts/dejavu-sans>



**Color Schemes:**

I will be using 24 bit .gif images with transparency and will be taking advantage of the full range computer and laptop monitor displays. File size be damned (although the images will not be complex and will contain solid fills of colors.