Chase Hatch and Brennan Strom

Mr. Fast

Web Application Development (PM)

4 November 2019

Project Description

Overview

In this interactive story project, we will be working with a partner to create our own code-based interactive stories targeted towards an eight- to twelve-year-old audience based on the twelve steps of a hero's journey that will run in JavaScript. The story with have three unique beginnings, each of which will have the potential to result in any of at least three different endings, and must branch off into at least two different paths per storyline-split. Every part of the story must be stored as an object, and all of the possible choices for each path must be stored inside an array. We will have limited time to create our project, so we must plan our time accordingly.

Pre-Production Phase

Firstly, to begin our planning process, we must first demonstrate our knowledge of the task at hand by writing a description of the steps that we will be taking in order to create our final product. Once our project description is complete, we must then create a flowchart using lucidchart.com (as will be demonstrated by our teacher in class) that will outline the different branching paths that our story will be able to take. Following the creation of this flowchart, we then must create a separate flowchart that will outline how our code will function. Finally, the pre-production phase will be concluded with the creation of a list of the tasks that we will need to

complete in order to create our final project and a meeting with our teacher to ensure that we have made sufficient-enough progress to begin the actual creation of our stories.

Production Phase

Coding:

After the pre-production phase, the coding will begin to be carried out on daily on a team Git repository, which we will be shown how to use and access by our teacher during class time. When coding, we must be sure to format everything properly and include helpful comments that properly document the function and flow of our program. All bindings, functions, and parameter names must be camelCased and be thoughtfully and meaningfully named as to describe the job that they perform. Additionally, all operations must be performed within functions, bindings must all be at the global level, and the code should follow the D.R.Y. (Don't Repeat Yourself) principles (or in other words, all repetitive code should be turned into functions and called as needed).

Writing:

In the meantime, we must also begin to type out our actual story. The story must follow three main guidelines. First, it must follow the twelve main progressions of the hero's journey: In Act 1, the hero's start in the ordinary world, their call to adventure, their refusal of the call, and the meeting of their mentor; in Act 2, the hero's first crossing of the threshold into the special world, the hero's confrontations with various tests, allies, and enemies; the approach to their end goal, the act of achieving their goal, and the final achievement of their goal; and in Act 3, the hero's attempt to return to the ordinary world, their final trial, and finally, the hero's triumphant return to the ordinary world. Secondly, the story must offer at least two different branching

points per choice made, and each optional choice must also abide to the twelve steps of the hero's journey. Finally, the story feature at least three optional beginnings and three optional endings with as few branchless encounters along the way to reaching these endings, and these beginnings and endings must also abide by the steps of the hero's journey.

The Program:

Additionally, while putting the project together, there are a few guidelines that we must keep in mind for how our program should function. Firstly, while in use, the program should display every part of the story that has occurred up until the reader's current spot in the story and append to it with each choice made. All of the choices that the user makes should be kept track of in an array. Secondly, every story part should be stored in an object, and the collection of all of the potential story choices should be stored in an array. Lastly, the program must not use the prompt, alert, or confirm methods in order to gather or display information; rather, it must feature clickable buttons or sections in order for the user to make each story choice.

Daily Reflections:

Finally, one last crucial step to the production-phase is the daily creation of written reflections of our progress. Throughout the production-phase, we must be sure to keep a personal journal of all of our progress, set-backs, and reflections on our work for the day on a Google Document, as it will be essential for proper planning, priority maintenance, and the insurance of focused, hard work during each day of our project's development.

Deadlines

One last and final thing that we must keep in mind while creating our project is the deadlines. Every week, we will hold a meeting with our teacher and assess our project's progress and discuss how we can better move forward with our projects. The following dates and their corresponding groups of assignments indicate on what day each task will be due and when the meetings for them will be held:

Date:	**Assignments Due:**
October 16th, 2019	● Project Description ● Task List
October 18th 2019	● Story Flowchart ● Task List ● Journals
October 23rd, 2019	● Complete Story ● Program Flowchart ● Task List ● Journals
October 25th, 2019	● Code Progress (Git) ● Task List ● Journals
October 29th, 2019	● Presentation Rough-Draft ● Code Progress (Git) ● Task List ● Journals
October 30th, 2019	● Presentation Final-Draft ● Code Progress (Git) ● Task List ● Journals
October 31st, 2019	● Final Presentation ● Final Task List
November 1st, 2019	● Final Code ● Final Journals ● Final Git Commit
November 4th, 2019	● Written Project Reflection

Project Description List

Here we have provided a brief list describing what we will be doing in order to complete our project:

- We are being asked to write in our own words what to do.
- We make a chart that has branching paths.
- We make a chart based off program logic.
- A task list.
- It will be updated as the days go on
- Plan first
- Show story and combine it with the choices that correlate to it.
- Don’t use pop up methods to gather data.
- Have buttons
- Use object
- Use array
- Keep track of choices
- Good code
- Well written code
- camelCase and good names
- Don’t repeat yourself
- Make functions
- Only bindings on global
- Commit to Git

- Hero’s 12 steps to success
- 2-3 branches.
- Limit number of no branches.
- 3 starts at least 3 ends
- Keep a journal