The Purple Cobras

10/2/2015

CS 4560

Team Members:

Emily Moore

Jordan Leach

Charles Chen

Kevin Xiao

Ryan Miers

Kameron Feola

Executive Summary

This project will give us some new experiences with our development capabilities such as graphical visualizations. Our API is to visualize data structures such as elements of an array, frames per second, or even mouse coordinates. To begin with, we will start small with simple elements such as the ones listed previously. It will have visualizations for more data structures in the future as we continue to work on it. Ease of use is also a major factor in our project as the typical client of API users are developers themselves. As the development of our API moves further along, more complex data structures and flexibility of usage will be our focus.

In order to test our API, we will create a simple learning game. This game will be a game centered around boolean logic. The player will control a logical input. This input will either be “true” or toggled to “false”. The state of the player could be shown with our API whether they are true or in false. There will be a series of logic gates with given inputs coming from the top of the screen. These gates will have only one open input that the player must satisfy. For example if they player is in “true” and the gate the player wishes to collect is an OR gate with one input that is false, the player will obtain that gate. If the player does not satisfy the gate, for example, a player in “true” mode tries to collect a NAND gate with an input “true”, the player will lose a life. Lives will be kept within an array with a maximum of 3 lives. Future games could be created

for testing purposes.

|  |  |
| --- | --- |
| Use Case Name | Using API |
| Participating Actors | * Game Developer * API Object |
| Entry Condition | The game developer successfully downloads/includes the API in their project |
| Flow of Events | 1. The developer includes the API in their project 2. The developer uses the functionality of the API and attempts to access data from within their project 3. The API object obtains the data 4. The API object generates a visualization of the data based on the developers specification |
| Exit Condition | 1. The API successfully visualizes the given data 2. The developer decides to not use the API |
| Exceptions | * The API fails to visualize the data * The developer incorrectly uses the API   + Syntactically, semantically, etc. * The developer incorrectly includes the API * The download of the API fails |
| Special Requirements | * The developer is given extensive documentation (including examples) of how to properly use the API |

|  |  |
| --- | --- |
| Use Case Name | Browsing Website |
| Participating Actors | * User * Web browser |
| Entry Condition | The GUI of the website is successfully loaded onto the users web browser |
| Flow of Events | 1. The user enters the URL belonging to the Learning Games website 2. The web browser sends a request to the specified server 3. The webpage is returned and loaded into the browser 4. The user is met with multiple options at the Home page:    1. View the “About” page which explains the purpose of the website    2. View the selection of available games to play 5. Upon choosing, the web browser sends a request for the new page 6. The new page is loaded onto the web browser for the user to view |
| Exit Condition | 1. The user closes the web browser 2. The web browser fails to load one of the options selected by the user 3. The server crashes |
| Exceptions | * The user enters the wrong URL * The server does not send the correct information to the web browser |
| Special Requirements |  |

|  |  |
| --- | --- |
| Use Case Name | Playing Game |
| Participating Actors | * User * Game * API Object |
| Entry Condition | The user successfully navigates the webpage and selects a game to play |
| Flow of events | 1. The user selects the game of their choice 2. The data within the game is obtained by the API 3. A display window is created during gameplay 4. The API displays the inner workings of the game to the user 5. The user continues playing the game    1. API modifies the visualization based on the current state of the game |
| Exit Condition | * The user closes the browser * The user leaves the website |
| Exceptions | * The API fails to visualize the data |
| Special Requirements | * A brief description of how to play the game is presented to the user |

Minimum Viable Product

Our API will at least be able to visualize the values of a given piece of data as well as whole structures. It will be easy to use as well as well documented. More complexity of data structure visualization will be added in the future.

Technology Platforms

* + D3.js
    - <http://d3js.org/>
    - D3.js is a JavaScript library that assists in creating visualizations from sets of data. Our reason for including this is to make more detailed visualizations and to possibly minimize and simplify the amount of code we write. We believe this will be used more for the visualization API although some features of D3.js may also be used to implement our game.
  + D3plus
    - <http://d3plus.org/>
    - D3plus is a JavaScript library that extends the functionality of d3.js. The reason for inclusion is to possibly add more functionality to the D3 code we write. This library also claims to simplify the code written for d3.js visualizations.
  + Processing.js
    - Processingjs.org
    - Processing.js is a JavaScript port of the processing visual programming language. The reason for including this library is to offer an alternative for visualization alongside d3.js. We believe that certain aspects of our game may be easier to visualize with d3.js and some may be easier to visualize with processing, therefore we have included both.
  + Yui Library
    - <http://yuilibrary.com/>
    - YUI is an open source JavaScript library for crating interactive web pages. We believe that this library can assist us in making our game and making our visualizations for the API.