Project 2: Door Prize Contest

Comp 250

100 Points

At the beginning of Computing Seminars on Friday mornings, we sometimes run a program that randomly selects four or five contestants to compete from those that are in attendance. The competition usually involves a race of some sort with one winner being selected at the end who receives the door prize. You are to create a JavaScript implementation of your own Door Prize Contest using whatever theme you would like. Dr. McCown will demo his Happy Face Race in class to give you an idea of how it should work.

The following specifications must be met in order to receive full credit:

1. **Participants Screen**: The program should display a Participants Screen where the user can type in all the names of those that are in attendance at the seminar. This should be done with a large <textarea> because normally the user will be copy and pasting a large list of names into the web page. A Pick Contestants button should be present which switches to the Contestants Screen.

2. **Contestants Screen**: The program should display four to six competitors who are randomly selected from the list of participants. Clearly identify these competitors using their full names and associated images. There should be two buttons available: 1) Race which switches to the Race Screen, and 2) Cancel which switches back to the Participants Screen and shows the participants the user typed in earlier.

3. **Race Screen**: The program should display a competition or race of some type that should take on average 4 – 45 seconds to complete. After the competitors are shown in their initial configurations, there should be a brief pause before the competition begins. There should be no need for user input once the competition starts. The competition must involve animation of some sort. The animation could be as simple as boats that travel from the left side of the screen to the right, but a more elaborate animation will be more visually appealing. The winner should be randomly chosen from the competitors, and when the competition is complete, there should be a brief pause before the Winner Screen is displayed.

4. **Winner Screen**: The winner should be clearly identified by name and image. A New Race button should be present which allows the user to start a new competition by displaying the Participants Screen which lists the same participants that were entered earlier.

5. Your program should be extremely robust. That means, for example, checking the name list for at least 5 names before trying to obtain 5 different names from the list. You should also ignore blank lines that may have been entered.

6. Your program must play at least one sound file (WAV, MP3, etc.) during the competition or once it has completed.

7. Place all your HTML code into `index.html` and JavaScript code in `doorprize.js` so your `index.html` file has no JavaScript code in it. You may use an embedded style sheet or external style sheet if you wish, but there should be only one HTML file.

8. Bonus: You can earn 5 bonus points (15 total points) for each of these feature you implement:

    a. **Kill a competitor**: During the competition, spawn an enemy that "kills" one of the competitors at random. For example, a shark could appear that sinks one of the boats. The enemy should be animated.

b. **Save the participants:** If you reload your webpage, the participant's names that were previously entered will disappear. Add a Save button which when pressed will store all the names using the `localStorage` object. When the page is first loaded, load all the names that were previously saved in the `localStorage` object. This way the names persist even if the user browses away from the web page.

c. **Canvas animation:** Use a `<canvas>` to display the winner's image, and use the `context.rotate()` method to animate a rotation so the winner's picture rotates a complete 360 degrees.

Submit your completed `index.html` and `doorprize.js` files to Easel before it is due. You must make your Door Prize Contest available on Taz at a URL of your choosing. Choose a URL that is not obvious so others are not tempted to find your code on Taz. Example:

```
http://taz.harding.edu/~username/Dprize999/
```

Place the URL in comments at the top of the `doorprize.js` file you submit to Easel so I can copy and paste it into my browser to test your game. *Do not* modify any of your files after you have submitted it to Easel.

The following criteria will be used when grading your projects, I will be looking for:

- Correctness - program should work correctly and implement all the requirements
- Robustness – program should not display JavaScript errors and handle unexpected input properly
- Pleasing interface – easy to figure out what to do, colors and graphics are pleasing to the eye, etc.
- HTML coding - tags are used correctly, appropriate title, etc.
- Coding style - use of arrays where appropriate, use of good variable and functions names, functions should have a single purpose and be used to reduce redundant code, proper indentation, use of the W3C event registration model (no onclick attributes).
- Citations - If you use some code you find on the Web, you better cite (author and URL) where you got the code in a comment. Do your best to write original code since you will not learn much if you just copy-paste someone else's solution to a problem. *No credit* will be given if you reuse large chunks of code others have written, even if you cite it.

Implementation Notes

The contest consists of four "screens" (Participants, Contestants, Race, and Winner), but these will just be tags in your index.html file where one is visible while the rest are hidden. You should investigate the CSS property "display" to see how this can be used to hide/display a .

You will use timers to perform the animations. You will find it helpful to position each of the competitors with relative or absolute positioning and then update the "left" property of each competitor via a function that is called repeatedly until one of the competitors finishes. Each competitor should advance by some random value so they don't all finish at the same time. See the animation example in the notes for code that will be helpful in implementing the animation.

When picking random contestants to participate in the competition, you will find it helpful to keep track of who has already been selected so you don't re-select the same person multiple times. You could store their names in

an array or just store their position from the master list. Storing their position will allow for multiple contestants that have the same name to appear, but that's acceptable for this program. Make sure you account for blank lines which should *not* be selected as participants! You might find the array function `array.splice()` helpful in removing blank lines from your array of names.

See the notes on how to use `<audio>` with JavaScript in order to play sound effects in your race. Sound effects that are used in the right place will make your race much more enjoyable.

It can be tempting to spend lots of time on the graphics, sounds, or a snazzy user interface. Please avoid doing so until *after* you have the project working correctly. You will be judged on how well the project implements the requirements, not how pretty the graphics are!