# CSE2WDX Assessment 2: Objectives Report

Student ID: 20223133

Full name: Kwee Cheong Michael Brandon Lee Sui Yuen.

|  |  |  |  |
| --- | --- | --- | --- |
| Objective | Example | Location | Description (1-2 sentences) |
| A.1: Create the document structure by using HTML | 1 | yahtzee.html: line 12-16 <header> line 17-33 <main> line 191-193<footer> | Structure the UI by using semantic markup, including markup for search engines and screen readers, such as Section, Article, Nav, Header, Footer, and Aside: The <header>, <main> and <footer> tags were used for some basic semantic structure. |
|  | 2 | yahtzee.html: Line 33-171 | Create a layout container in HTML: The <div>   |  | | --- | | tag is a layout container for the two <table> tags within. | |
| A.3: Apply styling to HTML elements programmatically | 1 | yahtzee.js: line 11-12 | Apply a transform: The dice “jump” a little when rolled. This styling is applied programmatically via the “animate()” function in jQuery |
|  | 2 | yahtzee.js: line 13 | Show and hide elements: The following message is showed only after the dice are initially rolled as a tip to explain how to put some dice aside to prevent rerolling: “Click on a die to prevent rerolling. Click again to undo.” |
| A.5: Establish the scope of objects and variables | 1 | yahtzee.js: | Keep objects out of the global namespace: The JavaScript file uses an immediately invoked function to keep objects out of the global namespace |
|  | 2 | yahtzee.js: line 8, 19, 24, 42, ,56 ,75 ,98 ,116, 140, 183, 197, 211, 224, 238, 244, 245, 248, 258, 260, 261, 323. | Use the “this” keyword to reference an object that fired an event: The “this” keyword is used many times to reference the button that fired the “click” event. |
| A.6: Create and implement objects and methods | 1 | yahtzee.js: line 316. | Implement native objects: The native “Math” object has been implemented to support random number generation for choosing the outcomes of dice rolls. |
|  | 2 | yahtzee.js: line 253 - 335. | Implement native methods and create custom methods: Custom methods have been created many times to modularise the JavaScript code. |
| B.1: Implement program flow | 1 | yahtzee.js: line 18 - 29. | Iterate across collections and array items: The jQuery “each()” method is used to iterate across a collection of images that are used to represent the faces of the dice. |
|  | 2 | yahtzee.js: line 34 – 242. | Used getCount() method helper and if statement to and .includes() to get score from rolled dices. |
| B.2: Raise and handle an event | 1 | yahtzee.js: line 6, 33, 47, 61, 80, 103, 121, 135, 145, 161, 175, 189, 203, 216, 229, 243. | Handle common events exposed by DOM (OnBlur, OnFocus, OnClick): The “click” event is handled in the code for each of the buttons and also each dice picture (to note which dice should not be re-rolled |
|  | 2 | yahtzee.js: line 243- 251. | Handle an event by using an anonymous function: An anonymous function has been used to respond to the click event for reserving dice that do not need to be rerolled. Names functions are mostly used instead to maintain code readability. |
| D.1: Style HTML text properties | 1 |  | Apply styles to text font, including WOFF, @font-face, size, and understudy fonts: The font family and font size has been changed for the whole page. |
|  | 2 | yahtzee.css: line 136. | Apply styles to text alignment, spacing, and indentation: The alignment of the “points” data has been set to left alignment. |
| D.2: Style HTML box properties | 1 | yahtzee.css: line 69 -80. line 30 -42. | Apply styles to alter appearance attributes, including size, borders, outline, padding, and margin: The size and margins of the dice pictures have been adjusted. |
|  | 2 | yahtzee.css: line 78. | Apply styles to establish and change an element’s position: The dice have the “relative” position set so that they can be moved around via an animation when they are rolled. |
| D.3: Create a flexible content layout | 1 | yahtzee.css: line 97. | Implement a layout using a flexible box model: The flexbox model has been enabled for the container hosting the “upper score” and “lower score” tables allowing them to sit side by side. |
|  | 2 | yahtzee.css: line 12. | Implement a layout using justify-content:space-between. With the columns for the tilted dice set to have the minimum width and the column for the heading “Yahtzee” set to “auto”, the tilted dice were pushed to the sides. |
| D.4: Create an animated and adaptive UI | 1 | yahtzee.css: line 44 - 49. | Animate objects by applying CSS transitions: A transition to the hover style has been implemented for the two dice in the top corners. |
|  | 2 | yahtzee.css: line 40 & line 60. line 59 & line 39. | Apply 3-D and 2-D transformations: A transformation to rotate the two dice in the top corners by 90 degrees more is applied for the transition described above. Shadow of dice implemented. |
| D.5: Find elements by using CSS selectors and jQuery | 1 | yahtzee.js: line 295. | Choose the correct selector to reference an element: Using the jQuery global function “$("#pointsTotal")” correctly references the HTML tag with the ID of “pointsTotal” |
|  | 2 | yahtzee.html: line 28 yahtzee.js: line 6. | Each button has an ID. |
| D.6: Structure a CSS file by using CSS selectors | 1 | yahtzee.css: line 134. | Reference elements correctly: By saying “td, th”, we are referencing all and tags at the same time. |
|  | 2 |  | Style an element based on pseudo-elements and pseudo-classes: The :hover pseudo-class is used as a non-essential feature on the page for a fun effect |