|  |  |  |
| --- | --- | --- |
|  | **Activity Guide - The Clicker Game** |  |

## The Clicker Game

You will be **creating your own “clicker” game** similar to the Apple Grabber game you worked on in this lesson.

The general object of the game is to click on an element that jumps around every time you click it. You will pick your own theme and decide what the rules are and how to keep score.

## Template

A template for the app is provided in Code Studio. The template has **4 screens** and some **basic navigation functionality and event handlers** set up for you.

The game play screen uses the images from the Apple Grabber game, but you should replace these with images related to your chosen theme.

Your **main tasks** are to:

* pick a theme for your game and add appropriate images and styling
* add variables to track some data during game play
* add code to event handlers to update the variables and display appropriately

## Requirements

Your application must have the following components:

* Your **game must end** - there must be a way to “win” and a way to “lose.”
* You **must use at least one variable** (but you may use as many as you like) to keep track of some data during game play (such as a score, or a number of attempts remaining, number of times a certain element was clicked, etc.).
* Your app will have at least **4 screens:**
	1. **A welcome screen** that explains what your game is and how to play, and lets the user start
	2. **A screen for game play** that displays some data on the screen that updates during play (such as the running score, number of attempts remaining, etc.)
	3. **A “win” screen** to show a congratulatory message if the player “wins” the game
	4. **A “loss” screen** to show when the player “loses” the game
* From the the win/loss screens, it **must** be possible to start the game over without simply re-running the app from scratch; this means resetting all variables and displays back to initial values.
* Your program code should follow good style, particularly by making sure to **create global variables in the first few lines of code** and giving UI elements and variables **descriptive and meaningful IDs.**
* Your user interface should be **intuitive to use.**

## Process

* **Choose** the theme of your game: what is jumping around the screen that the user is trying to click? Many themes and metaphors are possible.
* **Program** your app: it’s suggested you start by adding some functionality before style. Add one variable into the program, and add code to update and display it properly.
* **Have a peer test it out** to see if there are any more improvements you should make.
* **Make any necessary final adjustments.**
* **Peer Review:** you will review at least one of your classmates’ apps using the Peer Review Rubric, and someone will review yours.

**Advanced option: use getTime() to factor time into your game**

In the toolbox you’ll see a function called **getTime()** which returns a number that’s called a *timestamp,* representing the moment in time (to the millisecond) when the function was called. If you store a timestamp in a variable when the player starts the game and then grab a timestamp when they win, you can calculate how long it took and you could factor this into your score. Read the documentation for getTime() to see how it works.

|  |  |  |
| --- | --- | --- |
|  | **Peer Review Rubric** |  |

Project being reviewed:

Reviewer:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Yes** | **Almost** | **No** | **Comments** |
| **Using the App / Playing the game** |
| Welcome screen explains the game, how to win or lose, and allows the player to start. |  |  |  |  |
| At least one value is shown on screen that changes during game play (for example: a running score). |  |  |  |  |
| It is possible to win, and the app switches to a “win” screen. |  |  |  |  |
| It is possible to lose, and the app switches to a “loss” screen. |  |  |  |  |
| It is possible to start the game over *without* having to restart the program (i.e., the win/loss screens allow you to navigate back to the welcome screen). |  |  |  |  |
| When you start over, variables or other data and displays are properly reset to initial values. |  |  |  |  |
| The app is visually appealing and the user interface is intuitive to use. |  |  |  |  |
| **The Code** |
| The code contains at least one global variable; it appears at or near the first lines of code. |  |  |  |  |
| You can find the line or lines of code in an event handler function that updates a global variable. |  |  |  |  |
| UI elements have meaningful/ descriptive IDs. |  |  |  |  |