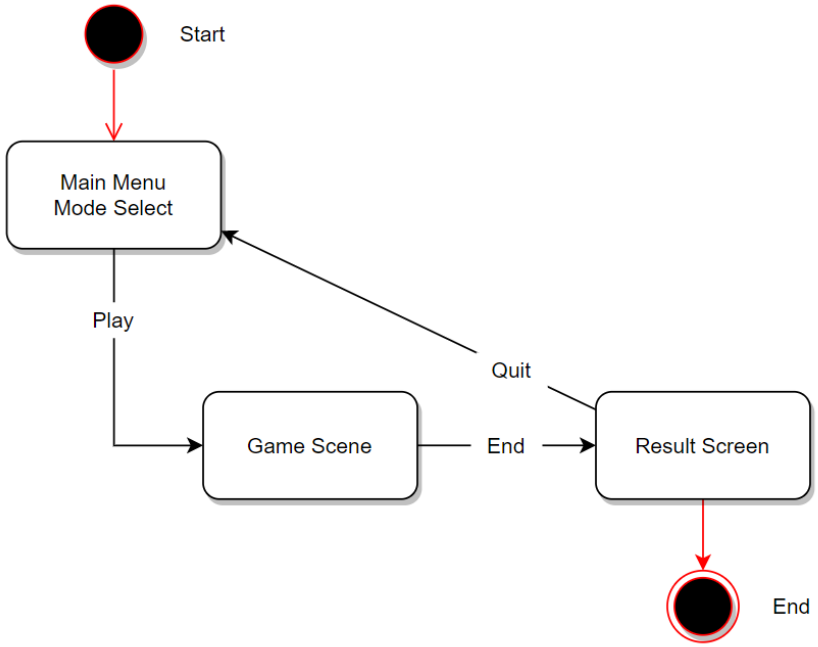
First steps

I'm a big fan of diagrams. They are fun and easy to make, they help visualize how things are going to work, and act as a point of reference when building a program or game loop. Along with setting up a repository, this will be the first thing I do. I'm going with Sourcetree for source control. I've committed a gitignore file and an empty Unity Project as a start.

Gameplay Loop

The project requirements are simple. A lot of things are open to interpretation, and besides the core game loop, everything else about the project is up to me. Awesome!

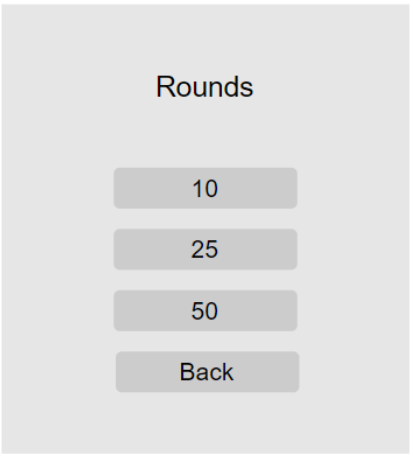
The game is based on The Stroop Color and Word Test (SCWT), or simply the Stroop Test. The core loop requires a start and end screen and a fixed number of rounds where the player is shown a colored word. Each time a word is shown, the player must input the color that matches the visual color of the word. At the end of the game, the player should be shown their score and may start a new game.



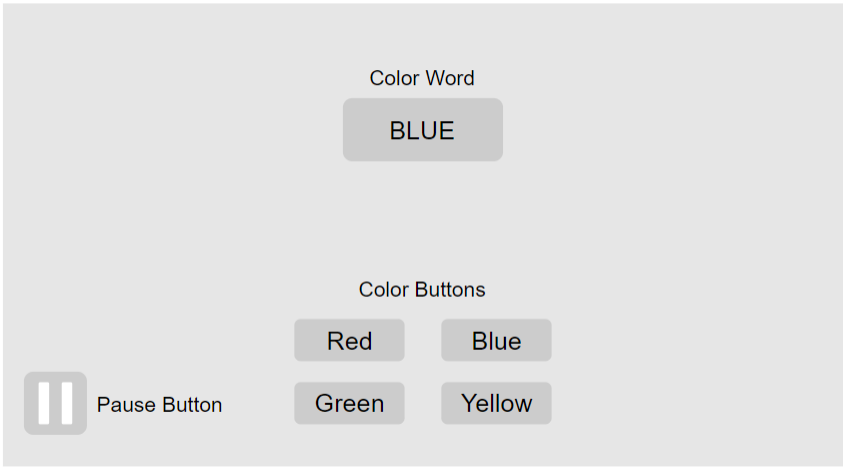
User Interface Mockups

Main Menu

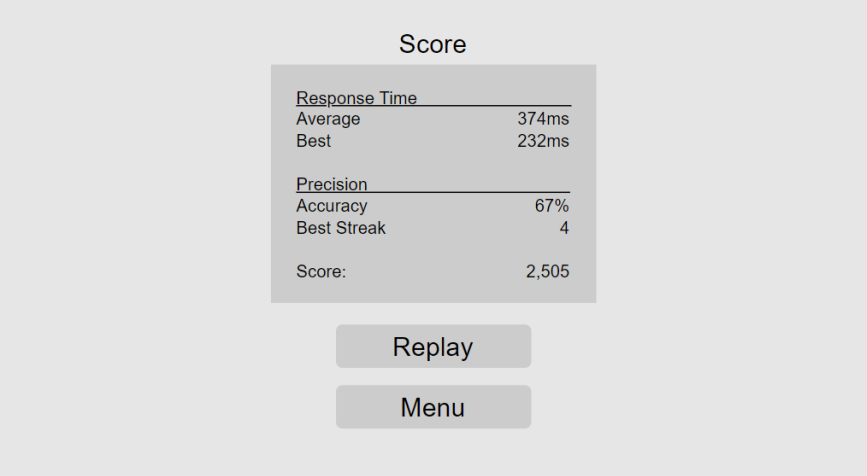
Mode Select



Game Scene



Result Screen



UI mockups aren't to scale, and the final result will no doubt vary

My plan is to have multiple modes with varying number of rounds. Brief sessions for those looking for a quick test, and longer ones for more accurate measurements.

Score is calculated based on your average response time for the session and how accurate you were. Players will be given a starting score of 1,000 (one second). At the end of the session, we'll subtract our average response time, then multiply by our accuracy. Additionally, I don't want the final number to be too big, so we'll just divide the result by 10.
Formula:
(1,000 - Avg response time) * (accuracy / 10)
Example:
(1,000ms - 287ms) * (80% / 10) = 5,704.

The lower your average response time and the more accurate you are, the better your score will be. Randomly choosing buttons will give you a high response time, but a low accuracy. Taking your time to choose right will give you greater accuracy, but at the cost of response time. For example:

Low average response time of 164ms, low accuracy of 25%
Score: 2,090

High average response time of 412ms, reasonable accuracy of 60%
Score: 3,528

Reasonable Average response time of 253ms, high accuracy of 80%
Score: 5,976

The average human response time is about 250ms and a response time of 150ms or less is considered superhuman by some, so we should discount any rounds from the final score should

the average response time be below 50-100ms or so, as this response time is very unrealistic, and was likely achieved by spamming one of the color buttons.

To make up for it, we could add another round to the session as the game plays to ensure we have our desired sample size for the final result.

The Stroop test is meant to measure more than just response time, so this won't be 100% accurate, but for the purpose of this project, it works just fine.

I'll be taking the bottom-up approach as I build this project. With the building blocks established, my next goal will be to head in-engine and sort out the UI.

A Quick Note on Hierarchy and Naming Standards

My goal is always to ensure readability, primarily because I have own eyesight problems, but also because good standards make for a neat project. Consistency and order mean a lot to me as they help with navigating and streamlining my workflow. That being said, not everything needs to be renamed. For example, I'll name a button after it's intended function, such as "Start Button", but the child text object can remain named as is. I follow this pattern throughout all my projects. This usually means I can tell what something is and what it is for immediately.

User Interface

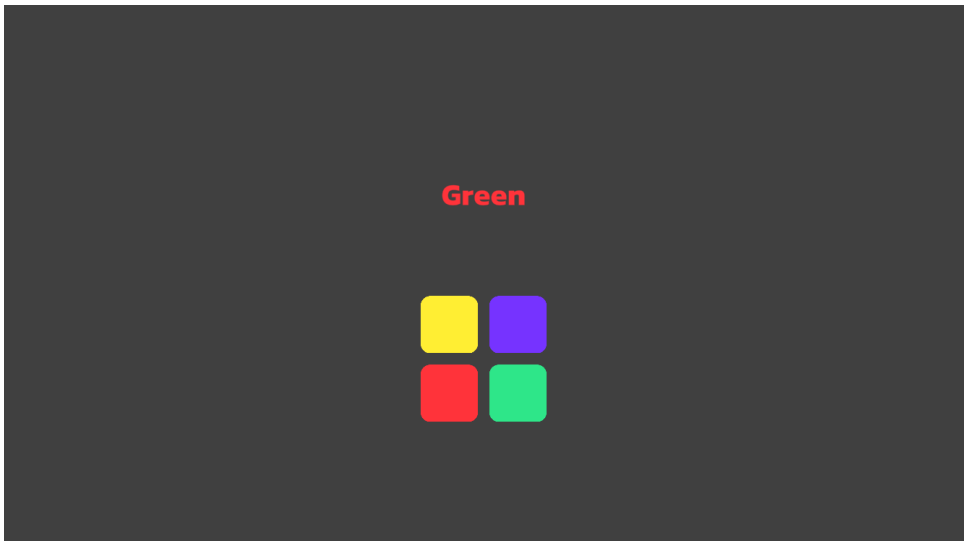
Seeing as assets from the asset store are allowed, I'll be importing LeanTween (and TextMeshPro, but that one doesn't really count).

LeanTween lets me quickly set up good looking UI transitions without much effort. Using the mockups as a reference, I've set up the main menu.

I would have loved to have used a Gif here, but Word didn't like that.
I've also already set up the color changing background, but that isn't visible here.

For the game screen, I made the following choices.

- The color word should be the first thing players see, as it is the focus of the game. So that was placed near the center of the screen.
- Because the aim of the game is to try for the best response time, I've grouped the color buttons together as to minimize the time it takes for players to move their mouse and click on one. Additionally, I've sized them so it's harder to miss one than it is to hit it. Lastly, a small gap between them all is there to help prevent hitting the wrong button by undershooting.
- I initially planned to randomize what color was shown on the buttons each round, but that would work against the point of the test. We only want to test the response time for recognizing the visual color of the word, not the buttons. Randomizing the colors of the buttons would mean we're now testing for both the word and the button. This also simplifies some of the code for the game.
- For colors, we'll go with Yellow, Purple, Red, Green. Keeping that order in mind will also simplify what I need to do in code. We don't need anything especially fancy here.



The logic of the game is very simple. I will utilize two lists, one for the colors and one for the words. We'll get a random index of each and ensure they aren't the same. Then we simply compare the button's color with the one at the random index of the color list. After a set number of rounds, we'll calculate results and display them to an end screen.

I'd like the reader to note that this was not my best response score, and in fact was a really, poor attempt on my part.

Summary

I enjoyed the whole process of making this game. It was fun and I even spent some time seeing what my best time could be.

I hope you enjoy your holidays!