6.2 Matching Game Brief

Overview

As an app developer, you've been asked to design an iOS word matching game for young children to help them associate words with images. The objective of the game is for the child to match a word with an image.

To start, create a list of words and a set of corresponding images you will use in the game. All of the images should display on the screen, but only one word should display at a time.

The user must select the image that corresponds to the word on the screen. If they select the correct image, they are awarded one point, but no points are awarded or deducted if they select an incorrect image.

Once the user has made a selection, their score updates and a new word appears. The image positions and states remain static. The game ends when each word has been displayed once. At the end of the game, the user sees their score and can start the game again.

Criteria for success

This task consolidates the knowledge you've gained so far in the course. It will test your ability to address the following user requirements:

- The project compiles and runs in Xcode.
- The app shuffles the words.
- All words are presented.
- The user receives a score when they select the correct image.
- There is a logical end to the game: it ends when all of the words have been displayed.
- When the game is over, the user is notified that it's over and their score is displayed.
- Auto Layout has been used and the app layout is consistent across a recent release iPad 10.5 an iPhone in Portrait mode.
- The user's current score displays on screen throughout the game.

App criteria and game logic

- Shuffle the list of words.
- Present each word to the user.
- Display the user's current score on the screen throughout the game.
- If the user selects the correct image, award them 1 point, and update the current score.
- If the user selects an incorrect image, award them 0 points.
- Once the user has made a selection, move on to the next word. Do not repeat the word.
- The game is over when there are no more words.
- Let the user know when the game is over and display their score on the screen.
- Prompt the user to play again.
- Use Auto Layout layout and ensure the layout is consistent across an iPad 10.5 and an iPhone 8 in portrait mode.

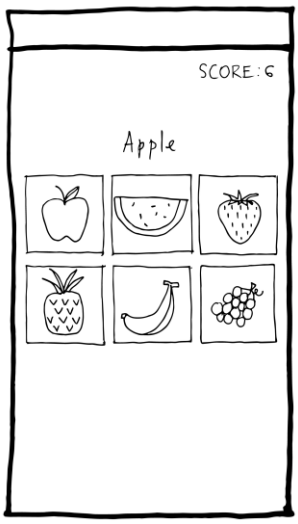
References and hints

This builds on the skills you gained from the Guided Project: Apple Pie but you have to extend your code.

When you work as an app developer, a designer will sometimes provide a wireframe which shows you exactly how they expect the app UI design and workflow to be developed.

In other instances, you are provided with a rough sketch, which is designed to visually communicate the key features and workflow of the app, but does not prescribe the design or navigation. In a workplace setting, you would clarify this expectation with the designer before you begin developing.

For this project, you have been provided with the rough sketch shown below. Create your app using this sketch as a guide only. As long as it meets the criteria, the design and navigation of your app is completely up to you.



Help code

Your main source of information should be the work you have completed throughout the course. Other sources are websites and tutorials.

- Take some time to sketch out your design.
- To confirm your game mechanics, get a friend to play the game.
- Keep it simple.
- Remember the Guided Project: Apple Pie demonstrates the Auto Layout use of Stack Views, which you'll need to build your app.

Buttons can be images

Here's some code to help you set an image on a button. Read up more about these methods in the XCode documentation on UIButton

```
button.setImage(UIImage(named:word), for: .normal)
button.setTitle(word, for: .normal)
```

Fixing stretchy looking buttons

Your image will look good on your button if the image's aspect ratio will fits correctly. To do this, configure each button using this code.

```
myButton.imageView?.contentMode = UIContentMode.scaleAspectFit
```

Shuffling an array

An array is a great structure to store your words in. On every new game, you'll need to shuffle your words. Here's a function to do this:

```
let stringArray = ["apple", "peach", "banana", "fig"]

// `shuffled()` will shuffle an existing array and return it

let shuffledArray = stringArray.shuffled()
```