SpaceAdventure
Lesson 15

Description

Refactor the repetitive `Planet` object instantiation with a dictionary, `for-in` loop, and `map`.

```
Welcome to the Solar System!
There are 8 planets to explore.
What is your name?
Jane
Nice to meet you, Jane. My name is Eliza, I'm an old friend of Siri.
Let's go on an adventure!
Shall I randomly choose a planet for you to visit? (Y or N)
N
Name the planet you would like to visit.
Neptune
Traveling to Neptune...
Arrived at Neptune. A very cold planet, furthest from the sun.
```

Learning Outcomes

- Analyze repetitive patterns in code, and discover how to reduce repetition with data structures.
- Discover the Swift `Dictionary` collection type, and describe key-value data structures.
- Apply a `for-in` loop to iterate over values in a data structure.
- Discover the Swift `map` function, and recognize transformation as the creation of a new collection based on the values within an existing collection.

Vocabulary

dictionary	key-value data structure	`Dictionary`
`for-in` loop	iterate	`map`
closure	`in`	

Materials

- **SpaceAdventure Lesson 15** Xcode project

Opening

How might we reduce the repetitive code in **main.swift**?

Agenda

- Discuss how the code in **main.swift** uses pairs of planet names and descriptions to create each `Planet` object, one statement at a time.
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the Swift Standard Library documentation for the `Dictionary` collection type.
- Explain that dictionaries are data structures consisting of key-value pairs.
- Replace the `TODO` comment and the individual instantiations of `Planet` objects with a dictionary of planet names and descriptions.

```
let planetData = [
   "Mercury": "A very hot planet, closest to the sun.",
   "Venus": "It's very cloudy here!",
   "Earth": "There is something very familiar about this planet.",
   "Mars": "Known as the red planet.",
   "Jupiter": "A gas giant, with a noticeable red spot.",
   "Saturn": "This planet has beautiful rings around it.",
   "Uranus": "Strangely, this planet rotates around on its side.",
   "Neptune": "A very cold planet, furthest from the sun."
]
...
```

- Explain the `Dictionary` literal syntax, and how each planet name serves as a key to the corresponding planet description.
- Replace the repetitive statements that append each planet object to the `planets` array with a `for-in` loop.

```
var planets = [Planet]()
for (name, description) in planetData {
   planets.append(Planet(name: name, description: description))
}
...
```

- Explain how the `for-in` loop iterates over each key-value pair in the `planetData` dictionary; assigns each key and value to the `name` and `description` constants; and uses these values to instantiate a `Planet` and append it to the `planets` array.

- Discuss how the code is less repetitive and more succinct.
- Run the app (⌘R), and observe that the functionality remains the same.
- Discuss how the program starts with a dictionary of planet data and ends with an array of `Planet` objects.
- Explain the concept of the Swift `map` function.
- Replace the array instantiation and `for-in` loop with a call to `map`.

```
let planets = planetData.map { name, description in
   Planet(name: name, description: description)
}
```

- Explain how the `map` function iterates over each key-value pair in the `planetData` dictionary; passes each key and value to the closure as `name` and `description`; and invokes the closure, which implicitly returns a `Planet` object.
- Discuss how using `map` removes the need to create a mutable array, and succinctly describes the transformation of data in the dictionary to a `Planet` array.
- Run the app (⌘R), and observe that the functionality remains the same.

Closing

What happens when we type the name of a planet that is not in the planetary system? Can you think of a way that lets us keep the planet data entirely separate from our code?

What else can we add to our `Planet` and `PlanetarySystem` classes? How might you add a star to the planetary system? What about moons? What about other planetary information? What about… martians?

Modifications and Extensions

- Investigate what Swift closures are, and explain how they work in your own words.
- Revise **main.swift** to load the planetary system name and planet data from a file, from a database, or from an external Web service.
- Add additional planetary systems to the program, and enable travelers to visit these systems.
- Add an association between planets and neighboring planets, and enable the traveler to hop from planet to planet.
- Incorporate a `Spaceship` class into the program.

Resources

The Swift Programming Language: About Swift https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/

The Swift Programming Language: A Swift Tour https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/GuidedTour.html

The Swift Programming Language: The Basics https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/TheBasics.html

The Swift Programming language: Dictionary https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/CollectionTypes.html#//apple_ref/doc/uid/TP40014097-CH8-ID113

Swift Standard Library Reference: Dictionary https://developer.apple.com/library/ios/documentation/General/Reference/SwiftStandardLibraryReference/Dictionary.html

The Swift Programming Language: Mutability of Collections https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/CollectionTypes.html#//apple_ref/doc/uid/TP40014097-CH8-ID106

The Swift Programming Language: For-In https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/ControlFlow.html#//apple_ref/doc/uid/TP40014097-CH9-ID122

Swift Standard Library Reference: Array https://developer.apple.com/library/ios/documentation/General/Reference/SwiftStandardLibraryReference/Array.html

The Swift Programming Language: Closures https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/Closures.html