

# SpaceAdventure

## Lesson 11

### Description

Prompt the user for the name of a planet to travel to, and print the planet's description.

```
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.  
Saturn  
Traveling to Saturn...  
Arrived at Saturn. This planet has beautiful rings around it.
```

### Learning Outcomes

- Practice defining parameterized methods within a class definition.
- Relate the Swift `for` loop syntax to similar constructs in other familiar languages.
- Apply subscript syntax to access specific objects within an array.
- Practice logic and control flow with an `if` statement.
- Discover the Swift `for-in` loop, and compare them with traditional `for` loops.

### Vocabulary

method call	method implementation	type annotation
<code>for</code> loop	counter variable	array subscripting
bracket	iterate	<code>for-in</code> loop

## Materials

- **SpaceAdventure Lesson 11** Xcode project

## Opening

How can we ask the traveler which planet he or she would like to visit, and then display that planet's description?

## Agenda

- Examine the `if` statement in the implementation of `determineDestination` within the `SpaceAdventure` class.
- Replace the `TODO` and `print` call with a prompt to capture a planet's name that the user will type, and a call to a private `visit:` method.

```
...
} else if decision == "N"{
    let planetName = responseToPrompt("Ok, name the planet you would
    like to visit.")
    visit(planetName)
} else {
...

```

- Implement a simple version of the `visit:` method.

```
private func visit(planetName: String) {
    print("Traveling to \(planetName)...")
}
```

- Explain the method definition syntax, emphasizing the parameter name and type annotation.
- Discuss how one might print the `description` of the `Planet` in the `planetarySystem.planets` array whose name matches the value of `planetName`.
- Discuss the drawbacks of using a long, explicit `if` statement, such as `if planetName == "Mercury"`.
- Complete an implementation of `visit:` that uses a traditional `for` loop.

```
private func visit(planetName: String) {
    print("Traveling to \(planetName)...")
    for var i = 0; i < planetarySystem.planets.count; ++i {
        let planet = planetarySystem.planets[i]
        if planetName == planet.name {
            print("Arrived at \(planet.name). \(planet.description)")
        }
    }
}
```

- Explain the traditional `for` loop syntax.
- Discuss the the idiom of array subscripting using a `for` loop counter variable.
- Run the program (⌘R), enter a name, choose N, type a valid planet name, and observe the results displayed in the console (⇧⌘C).
- Discuss the first two lines of the `for` loop.

```
for var i = 0; i < planetarySystem.planets.count; ++i {
    let planet = planetarySystem.planets[i]
```

- Discuss how the loop iterates over each item in the array by using the counter variable to retrieve a `Planet` object out of the array, assigning the object to a `planet` constant.
- Replace the traditional `for` loop with a `for-in` loop.

```
for planet in planetarySystem.planets {
    if planetName == planet.name {
        print("Arrived at \(planet.name). \(planet.description)")
    }
}
```

- Discuss how the `for-in` loop manages the iteration, assigning each `Planet` object to the implicit `planet` constant during each repetition of the loop.
- Run the program (⌘R), enter a name, choose N, type a planet name, and observe the results displayed in the console (⇧⌘C).

## Closing

What happens when the traveler types something else besides a valid planet name?

## Modifications and Extensions

- Enhance the `visit:` method to handle cases where the traveler types an invalid planet name, and add logic to make certain planets unvisitable.

## Resources

The Swift Programming Language: About Swift [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/](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](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](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/TheBasics.html)

The Swift Programming Language: Methods [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/Methods.html](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/Methods.html)

The Swift Programming Language: Subscripts [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/Subscripts.html](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/Subscripts.html)

The Swift Programming Language: Collection Types [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/CollectionTypes.html](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/CollectionTypes.html)

The Swift Programming Language: Control Flow [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/ControlFlow.html](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/ControlFlow.html)