

# UnitConverter

## Lesson 2



### Description

Set the main view controller as the picker view data source.

### Learning Outcomes

- Recognize the dependencies of interface components, such as UIPickerView.
- Define Swift protocols and relate them to object-oriented interfaces.
- Apply a protocol adoption in Swift.
- Experiment with protocol method implementations to observe how they affect the caller.
- Recognize the Swift external parameter name syntax.



### Vocabulary

connection well	UIPickerView	protocol
object-oriented interface	protocol adoption	Issue Navigator
UIPickerViewDataSource	method prototype	implementation
external parameter name		

### Materials

- **UnitConverter Lesson 2** Xcode project
- **Picker Views** presentation
- **Protocols** presentation

## Opening

What does the picker view need in order to do its job?

## Agenda

- Run the app (⌘R) and observe that the picker is unpopulated with the California city names.
- Present the picker view, and the need for a data source and delegate.
- Using Interface Builder, set the main View Controller as the picker view datasource by Control-clicking on the picker view, and dragging a connection from the `dataSource` connection well to the View Controller in the Document Outline (⌘⇧L).
- Run the app, observe the crash, and inspect the console output:  
"[UnitConverter.ViewController numberOfComponentsInPickerView:]: unrecognized selector sent to instance."
- Discuss how the picker view's data source is the view controller, but the `ViewController` class does not yet implement the methods that conform to the `UIPickerViewDataSource` protocol.
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `UIPickerViewDataSource Protocol Reference` and the methods `numberOfComponentsInPickerView:` and `pickerView:numberOfRowsInComponent:`.
- Add the `UIPickerViewDataSource` protocol declaration to the controller class.

```
class ViewController: UIViewController, UIPickerViewDataSource {
```

- Present the concept of protocols.
- View the Issue Navigator (⌘4), and notice the warnings indicating the methods necessary for conforming to the `UIPickerViewDataSource` protocol.
- Implement `numberOfComponentsInPickerView:` and `pickerView:numberOfRowsInComponent:`.

```
func numberOfComponentsInPickerView(pickerView: UIPickerView) -> Int {  
    return 1  
}  
  
func pickerView(pickerView: UIPickerView, numberOfRowsInComponent  
    component: Int) -> Int {  
    return 10  
}
```

- Explain the use of the external parameter name in the `pickerView:numberOfRowsInComponent:` method and the need to include the name when calling the method.

- Explain how the picker view will call these methods to determine how many flickable components it has and how many rows are in the picker view.
- Run the app (⌘R), and observe that the picker has one scrollable element that contains ten rows.
- Experiment with different return values for `numberOfComponentsInPickerView:` (e.g., 3) and `pickerView:numberOfRowsInComponent:` (e.g., 2 or 20). Run the app (⌘R) to observe how the picker view changes based on the return values from these two protocol methods.
- Discuss the relationship between the two protocol methods and the picker view.

## Closing

Why do you think the picker view itself does not determine how many components and rows it contains?

## Modifications and Extensions

- Add the picker view as a `UIPickerView` property in the `ViewController` class, and set its `dataSource` with code in `viewDidLoad`.

## Resources

UIKit User Interface Catalog: Picker Views <https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/UIKitUICatalog/UIPickerView.html>

UIPickerView Class Reference [https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIPickerView\\_Class/](https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIPickerView_Class/)

UIPickerViewDataSource Protocol Reference [https://developer.apple.com/library/ios/documentation/iPhone/Reference/UIPickerViewDataSource\\_Protocol/](https://developer.apple.com/library/ios/documentation/iPhone/Reference/UIPickerViewDataSource_Protocol/)

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

Core Cocoa Competencies: Protocol <https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/Protocol.html>

The Swift Programming Language: External Parameter Names [https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift\\_Programming\\_Language/Functions.html#//apple\\_ref/doc/uid/TP40014097-CH10-ID167](https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/Functions.html#//apple_ref/doc/uid/TP40014097-CH10-ID167)

Start Developing iOS Apps Today: Finding Information <https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/FindingInformation.html>