SUMMARY

In this chapter, we have worked on these topics:

- have an introduction to UML
- set up app specification
- create class diagram for the model
- declare enumeration
- declare class attribute
- declare class initializer
- use static attributes

If any of these is unclear to you, please make sure to go back and read the related part or parts before moving on the the next chapter.

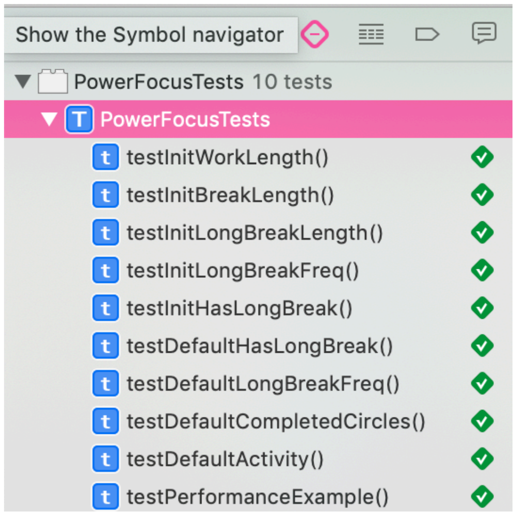
If you can't figure out the exercises on your own, you can check the answers at

https://github.com/CarlistleZ/MyiOSTutorial/blob/master/Chapter_Ten_Exercises.zip

For reference, you can download the complete Xcode project from

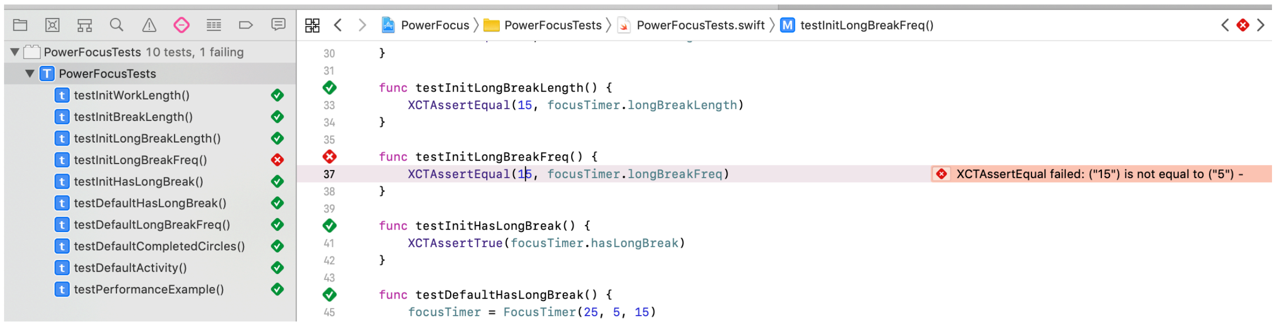
https://github.com/CarlistleZ/MyiOSTutorial/blob/master/Chapter_Ten_PowerFocus.zip

Open the test navigator in the left pane or press **Command(⌘) + 6**. Start the PowerFocusTests class by pressing the start button on the right, you will see all the results of the test methods with assertion.



If your model is correctly written, all tests should pass. If not, go to the model and try to fix the issue. You can check fail details in the popup on the right. For example, change the expected value of long break frequency from 5 to 15. Re-run this test and you will see the following error message: XCAssertEqual failed: ("15") is not equal to ("5").

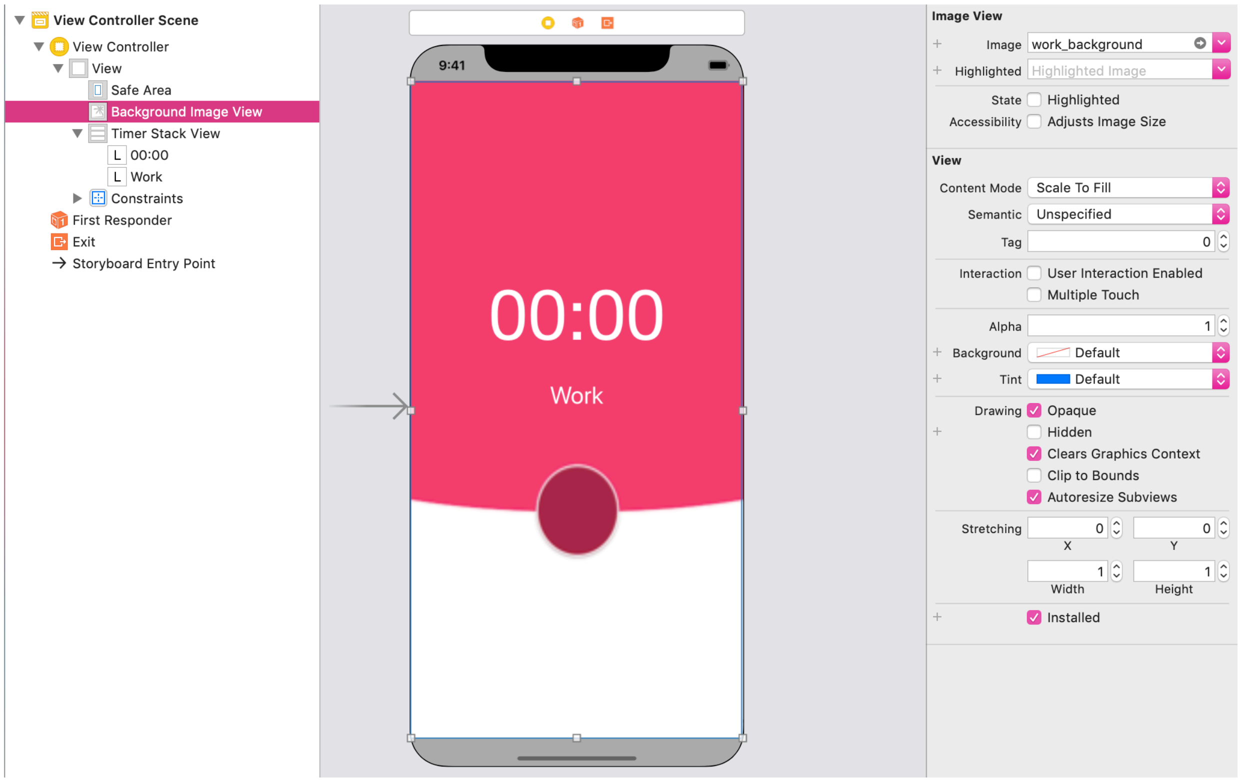
You can check the value during runtime and compare it to your expected value. You should bear it in mind that when a unit test fails, it doesn't always mean that your model has a bug. In this case, the expected value in test method is incorrect. Change 15 back to 5, the test should pass again.


PowerFocusTests 10 tests, 1 failing
PowerFocusTests
testInitWorkLength()
testInitBreakLength()
testInitLongBreakLength()
testInitLongBreakFreq()
testInitHasLongBreak()
testDefaultHasLongBreak()
testDefaultLongBreakFreq()
testDefaultCompletedCircles()
testDefaultActivity()
testPerformanceExample()
PowerFocus 〉 PowerFocusTests 〉 PowerFocusTests.swift 〉 testInitLongBreakFreq()

```
    }

    func testInitLongBreakLength() {
        XCTAssertEqual(15, focusTimer.longBreakLength)
    }

    func testInitLongBreakFreq() {
        XCTAssertEqual(15, focusTimer.longBreakFreq)      XCTAssertEqual failed: ("15") is not equal to ("5") -
    }

    func testInitHasLongBreak() {
        XCTAssertTrue(focusTimer.hasLongBreak)
    }

    func testDefaultHasLongBreak() {
        focusTimer = FocusTimer(25, 5, 15)
```

The background image is covering everything else, this is obviously not what we want. It is displayed in this way because the order in the view hierarchy is important. It always renders the widgets from top to bottom. If the current widget is not opaque (alpha is 1), it will cover all the previously rendered widget in its rectangle. The “**Timer Stack View**” is rendered before the “**Background Image View**” and since the image view isn’t opaque, we can no longer see the stack view even if it’s still there. Its graphic layout won’t even be calculated by the system to save CPU resource. This is a common reason why a widget appears to be “missing”.

To solve this, simply drag the image view above the stack view. Since the background of the stack view is transparent, you will see both of them at the same time.



```
▼ View Controller Scene
  ▼ View Controller
    ▼ View
        Safe Area
        Background Image View
      ▼ Timer Stack View
          L 00:00
          L Work
      ▶ Constraints
    First Responder
    Exit
  → Storyboard Entry Point
```

Image View

Image: work_background
Highlighted: Highlighted Image
State: ☐ Highlighted
Accessibility: ☐ Adjusts Image Size

View

Content Mode: Scale To Fill
Semantic: Unspecified
Tag: 0
Interaction: ☐ User Interaction Enabled
☐ Multiple Touch
Alpha: 1
Background: Default
Tint: Default
Drawing: ☑ Opaque
☐ Hidden
☑ Clears Graphics Context
☐ Clip to Bounds
☑ Autoresize Subviews
Stretching: X 0, Y 0
Width 1, Height 1
☑ Installed

Congratulations, now you know how to create storyboards for an iOS app! You can try the following exercises to test your self. If you are stuck somewhere, go back and read the corresponding section again. If you are really stuck, you can check out the solution at the end of this chapter.