

TDD

(Test Driven Development)

iOS D조 쌍쌍바

TDD란?

Test Driven Development
테스트 주도 개발이라는 개발 방법론

TDD를 하는 이유

소프트웨어 개발 시 작은 부분이 수정될 경우
관련된 모든 부분에 있어서 테스트가 다시 이루어져야 한다.

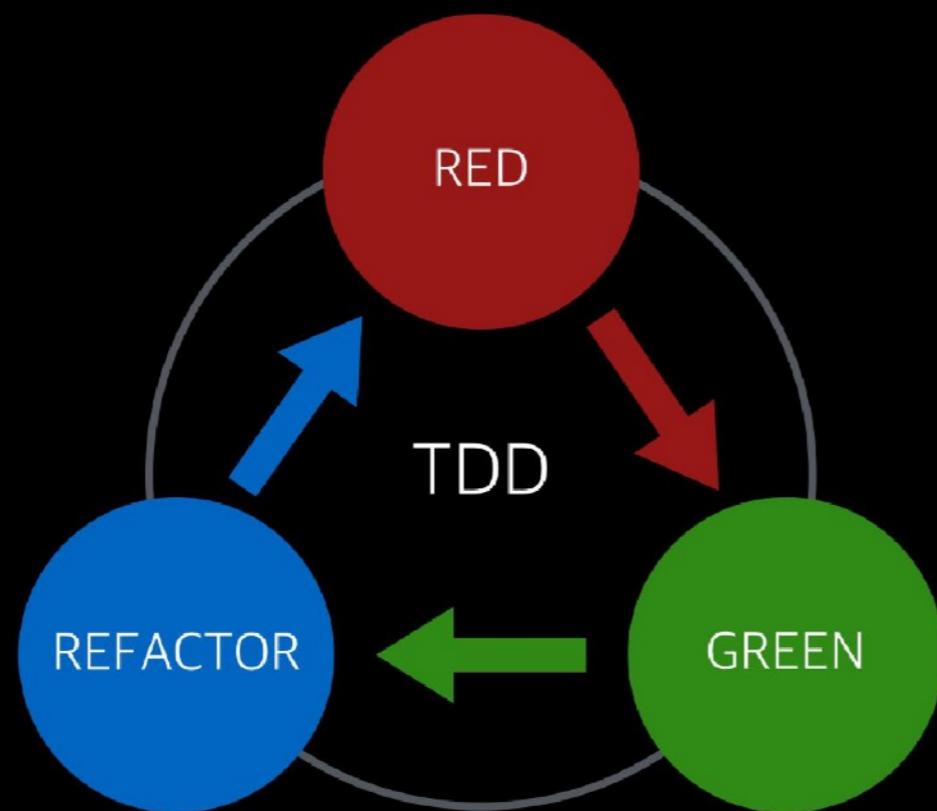
이전에 정상적으로 동작 하던 기능이 기능 추가 후에도
여전히 정상적으로 동작할 것이라고 감히 예상할 수 없기 때문이다.

TDD는 이러한 문제를 테스트 코드 작성을
통해 해결하는 방법을 제시한다.

TDD란?

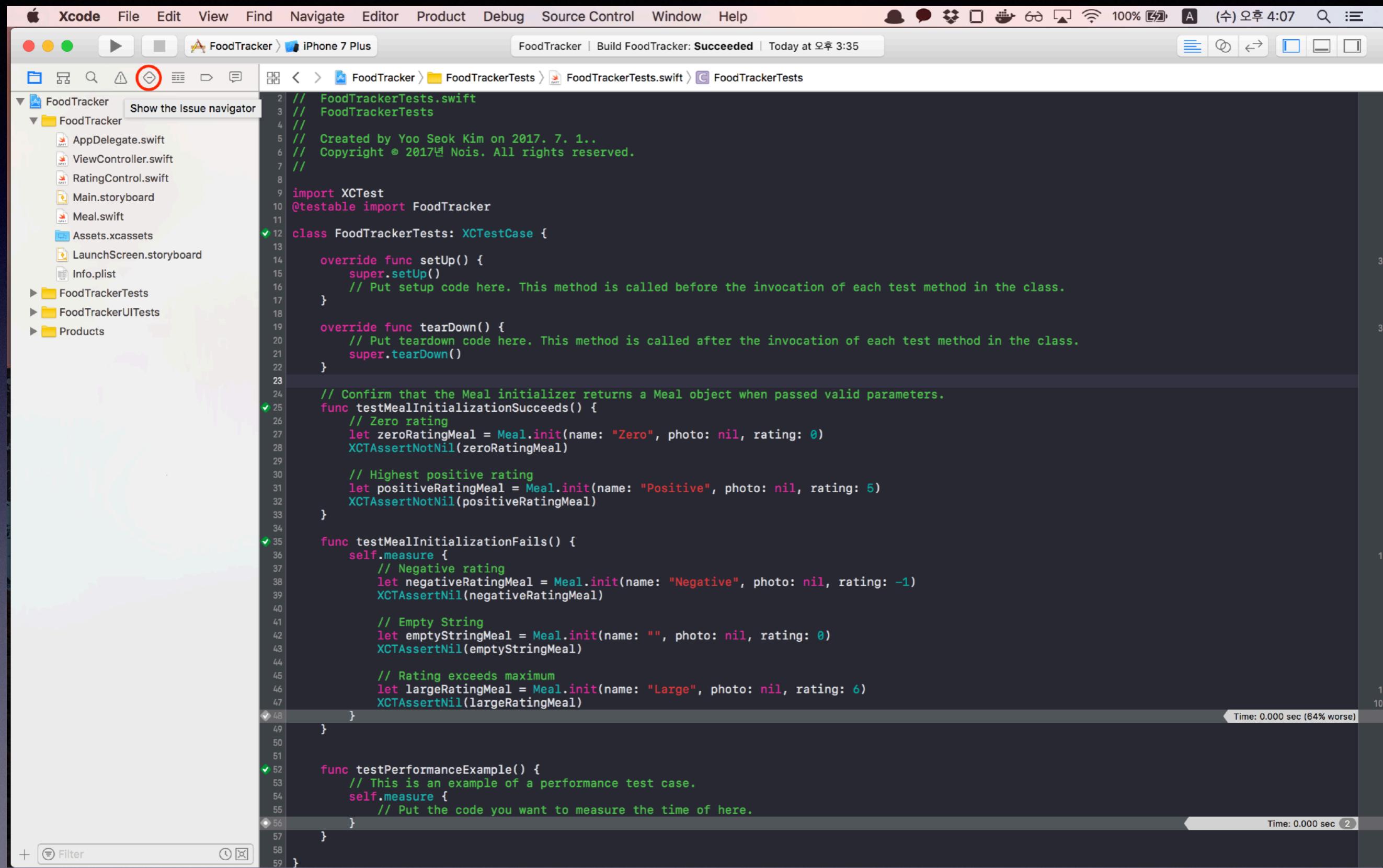
THE THREE LAWS OF TDD

Write failing test



1. 실패하는 테스트 코드 작성
2. 성공하는 테스트 코드 작성
3. 작성한 코드 리팩토링

TDD Unit Test 적용해보기



The screenshot shows the Xcode interface with the following details:

- Menu Bar:** Xcode, File, Edit, View, Find, Navigate, Editor, Product, Debug, Source Control, Window, Help.
- Toolbar:** Includes icons for Run, Stop, Build, and others.
- Project Navigator:** Shows the project structure under "FoodTracker".
- Issue Navigator:** A red circle highlights the "Show the Issue navigator" button.
- Editor:** Displays the code for `FoodTrackerTests.swift`. The code is a unit test case for the `FoodTracker` application, using `XCTest` and `@testable import FoodTracker`.
- Build Status:** Shows "Build FoodTracker: Succeeded | Today at 오후 4:07".
- Bottom Status Bar:** Shows "Time: 0.000 sec (64% worse)" and "Time: 0.000 sec 2".

```
// FoodTrackerTests.swift
// FoodTrackerTests
//
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

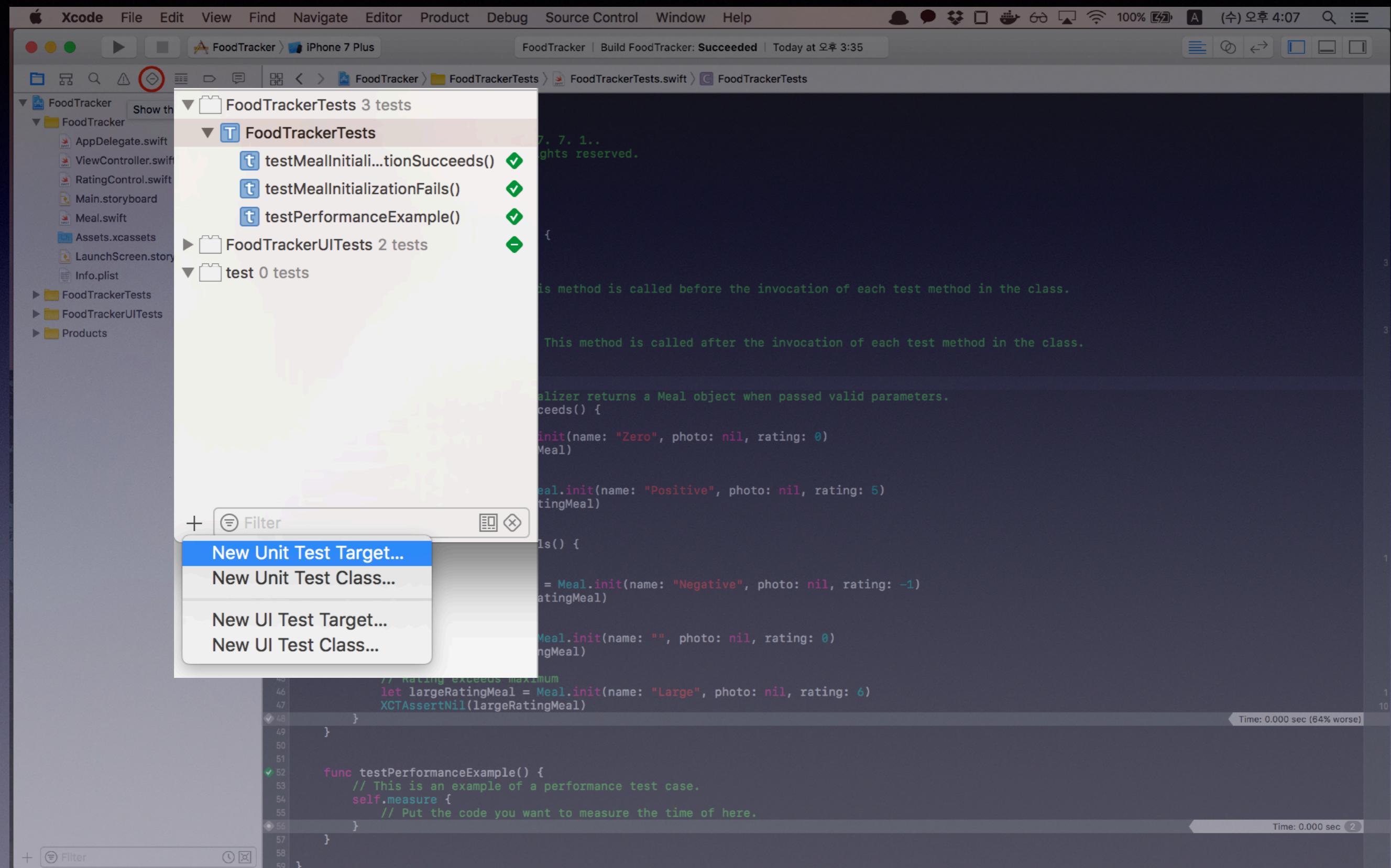
    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

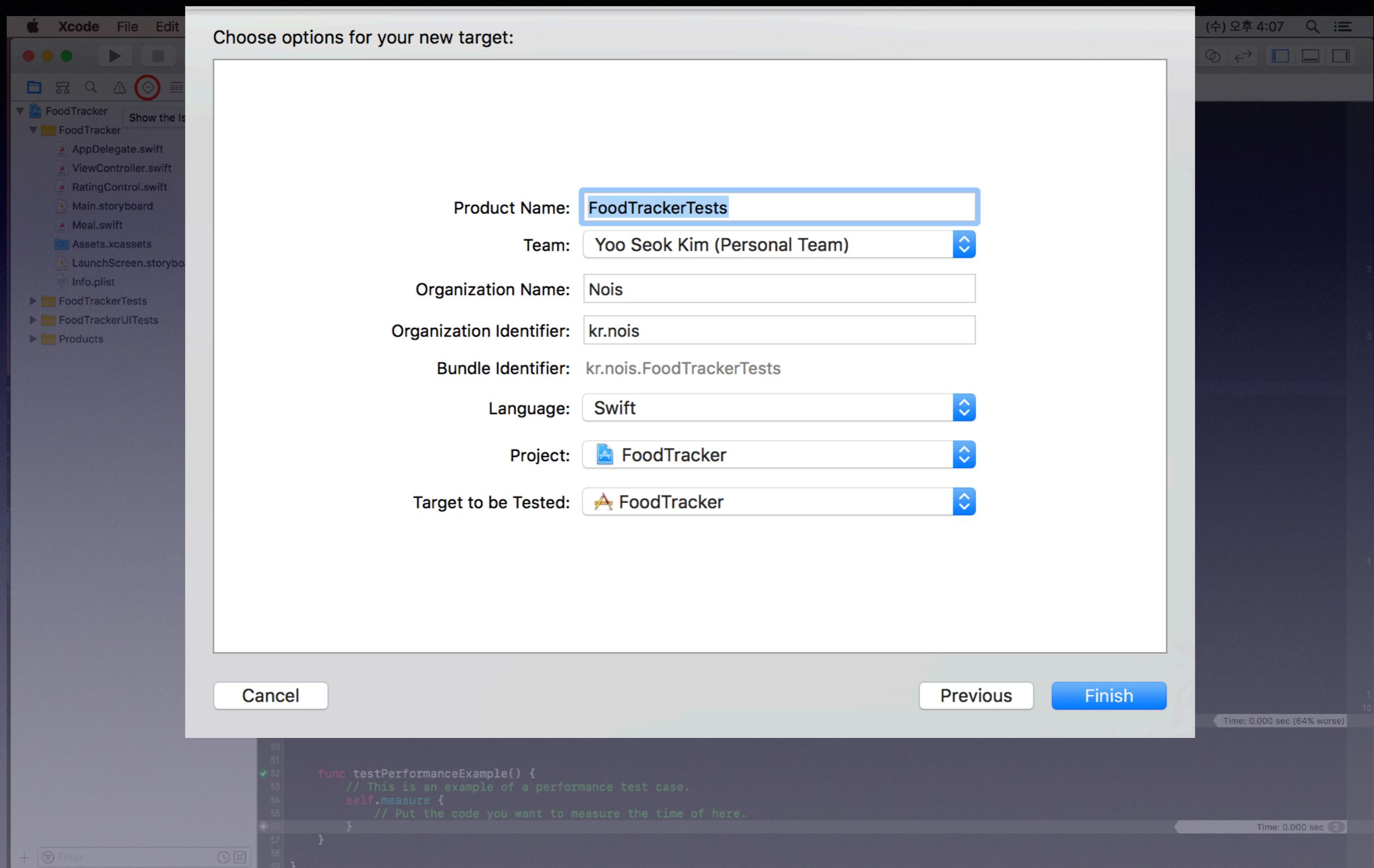
            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }

    func testPerformanceExample() {
        // This is an example of a performance test case.
        self.measure {
            // Put the code you want to measure the time of here.
        }
    }
}
```

TDD Unit Test 적용해보기



TDD Unit Test 적용해보기



TDD Unit Test 적용해보기

```
import XCTest

class Tests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    func testExample() {
        // This is an example of a functional test case.
        // Use XCTAssert and related functions to verify your tests produce the correct results.
    }

    func testPerformanceExample() {
        // This is an example of a performance test case.
        self.measure {
            // Put the code you want to measure the time of here.
        }
    }
}
```

TDD Unit Test 적용해보기

```
import XCTest

class Tests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    func testExample() {
        // This is an example of a functional test case.
        // Use XCTAssert and related functions to verify your tests produce the correct results.
    }

    func testPerformanceExample() {
        // This is an example of a performance test case.
        self.measure {
            // Put the code you want to measure the time of here.
        }
    }
}
```

TDD Unit Test 적용해보기

```
import XCTest

class Tests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    func testExample() {
        // This is an example of a functional test case.
        // Use XCTAssert and related functions to verify your tests produce the correct results.
    }

    func testPerformanceExample() {
        // This is an example of a performance test case.
        self.measure {
            // Put the code you want to measure the time of here.
        }
    }
}
```

TDD Unit Test 적용해보기

```
import UIKit

class Meal {

    //MARK: Properties

    var name: String
    var photo: UIImage?
    var rating: Int

    //MARK: Initialization

    init?(name: String, photo: UIImage?, rating: Int) {
        // Initialization should fail if there is no name or if the rating is negative.
        if name.isEmpty || rating < 0 {
            return nil
        }

        // Initialize stored properties.
        self.name = name
        self.photo = photo
        self.rating = rating
    }
}
```

TDD Unit Test 적용해보기

```
// Confirm that the Meal initializer returns a Meal object when passed valid parameters.
func testMealInitializationSucceeds() {
    // Zero rating
    let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
    XCTAssertNotNil(zeroRatingMeal)

    // Highest positive rating
    let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
    XCTAssertNotNil(positiveRatingMeal)
}

func testMealInitializationFails() {
    self.measure {
        // Negative rating
        let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
        XCTAssertNil(negativeRatingMeal)

        // Empty String
        let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
        XCTAssertNil(emptyStringMeal)

        // Rating exceeds maximum
        let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
        XCTAssertNil(largeRatingMeal)
    }
}
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with a expanded "FoodTrackerTests" group containing "FoodTrackerTests" and "Tests".
- Editor:** Displays the code for `FoodTrackerTests.swift`. The code is a unit test for the `FoodTracker` library, specifically testing the `Meal` initializer.
- Context Menu:** A context menu is open over the code editor, with the "Build for Testing" option selected. Other options visible in the menu include "Build for Running", "Build for Profiling", and "Analyze".
- Toolbar:** Standard Xcode toolbar items like "File", "Edit", "View", "Find", etc., are at the top.
- Status Bar:** Shows battery level, signal strength, and the date/time: "(수) 오후 4:23".

```
// FoodTrackerTests.swift
// FoodTrackerTests
//
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with a project named "FoodTracker". The "FoodTrackerTests" target is selected, showing two test cases: "testMealInitializationSucceeds()" and "testMealInitializationFails()". A context menu is open over the "FoodTrackerTests" target, with "Test" selected. The status bar at the top right indicates "FoodTracker: Failed | Today at 오후 4:13". The main editor area displays the source code for "FoodTrackerTests.swift". The code includes a class definition for "FoodTrackerTests" with overridden "setUp()" and "tearDown()" methods. It also contains two test functions: "testMealInitializationSucceeds()" and "testMealInitializationFails()", which use "Meal" objects and "XCTAssertNil" to verify meal initialization.

```
// FoodTrackerTests.swift
// Created by Yoo on 2017. 10. 13.
// Copyright © 2017 Yoo. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {
    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with the following details:

- File Menu:** Xcode, File, Edit, View, Find, Navigate, Editor, Product, Debug, Source Control, Window, Help.
- Toolbar:** Standard Xcode toolbar with icons for play, stop, run, etc.
- Project Navigator:** Shows the project structure with **FoodTrackerTests** selected. A red circle highlights the play button icon next to the test target.
- Search Bar:** FoodTracker | Build FoodTracker: Failed | Today at 오후 4:13
- Editor:** Displays the **FoodTrackerTests.swift** file content. The code implements a **XCTestCase** for **FoodTracker**, defining **setUp()** and **tearDown()** methods, and two test functions: **testMealInitializationSucceeds()** and **testMealInitializationFails()**.
- Annotations:** Four red circles highlight specific parts of the code:
 - Line 12: The opening brace of the **FoodTrackerTests** class definition.
 - Line 25: The opening brace of the **testMealInitializationSucceeds()** function.
 - Line 35: The opening brace of the **testMealInitializationFails()** function.
 - Line 52: The closing brace of the **FoodTrackerTests** class definition.
- Bottom Bar:** Includes a '+' button, a **Filter** button, and other standard Xcode navigation buttons.

```
// FoodTrackerTests.swift
// FoodTrackerTests
//
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.
//
import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with the following details:

- Project Structure:** The left sidebar shows a project named "FoodTracker" with two test targets: "FoodTrackerTests" and "FoodTrackerUITests".
- Code Editor:** The main editor window displays `FoodTrackerTests.swift`. The code defines a `XCTTestCase` named `FoodTrackerTests` with two test methods: `testMealInitializationSucceeds()` and `testMealInitializationFails()`.
- Test Results:** The status bar at the top indicates "2 tests, 1 failing". The `testMealInitializationSucceeds()` method is marked with a green checkmark, indicating it has passed. The `testMealInitializationFails()` method is marked with a red X, indicating it has failed.
- Console Output:** A red bar at the bottom right shows the error message: `XCTAssertNil failed: "FoodTracker.Meal"`.
- Bottom Bar:** The Xcode toolbar at the bottom includes icons for file operations like New, Open, Save, and Print, along with other standard Mac OS X-style icons.

```
// FoodTrackerTests.swift
// FoodTrackerTests
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}

// caught "NSInternalInconsistencyException", "Performance Metrics must provide 10 measurements."
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with the project 'FoodTracker' open. The main window displays the file 'FoodTrackerTests.swift' under the 'FoodTrackerTests' target. The code implements unit tests for the 'Meal' class, specifically focusing on its initialization logic.

```
// FoodTrackerTests.swift
// FoodTrackerTests
//
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    init?(name: String, photo: UIImage?, rating: Int) {
        // Initialization should fail if there is no name or if the rating is negative.
        if name.isEmpty || rating < 0 {
            return nil
        }
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}
```

A red rectangular box highlights the section of code where the `init?` constructor is defined. The code checks for three invalid initialization cases: a negative rating, an empty string for the meal name, and a rating value that exceeds the maximum allowed (which is implicitly 5 based on the code). Each of these cases results in a `XCTAssertNil` assertion, which is failing as indicated by the red status bar message:

XCTAssertNil failed: "FoodTracker.Meal" -

TDD Unit Test 적용해보기

```
import UIKit

class Meal {

    //MARK: Properties

    var name: String
    var photo: UIImage?
    var rating: Int

    //MARK: Initialization

    init?(name: String, photo: UIImage?, rating: Int) {
        // The name must not be empty
        guard !name.isEmpty else {
            return nil
        }

        // The rating must be between 0 and 5 inclusively
        guard (rating >= 0) && (rating <= 5) else {
            return nil
        }

        // Initialize stored properties.
        self.name = name
        self.photo = photo
        self.rating = rating
    }
}
```

TDD Unit Test 적용해보기

The screenshot shows the Xcode interface with the following details:

- Menu Bar:** Xcode, File, Edit, View, Find, Navigate, Editor, Product, Debug, Source Control, Window, Help.
- Toolbar:** Standard Xcode toolbar items.
- Project Navigator:** Shows the project structure with two test targets: FoodTrackerTests and FoodTrackerUITests. The FoodTrackerTests target has two passing tests: testMealInitializationSucceeds() and testMealInitializationFails().
- Editor:** Displays the code for `FoodTrackerTests.swift`. The code defines a `XCTestCase` subclass `FoodTrackerTests` with two test methods: `testMealInitializationSucceeds()` and `testMealInitializationFails()`. The failing test `testMealInitializationFails()` is currently selected.
- Output Area:** Shows the message "Finished running FoodTracker on iPhone 7 Plus" and a status bar indicating 100% battery, A (수) 오후 4:30, and a search bar.
- Bottom Status Bar:** Shows "Time: 0.000 sec (78% better)".

```
// FoodTrackerTests.swift
// FoodTrackerTests
//
// Created by Yoo Seok Kim on 2017. 7. 1..
// Copyright © 2017년 Nois. All rights reserved.

import XCTest
@testable import FoodTracker

class FoodTrackerTests: XCTestCase {

    override func setUp() {
        super.setUp()
        // Put setup code here. This method is called before the invocation of each test method in the class.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
    func testMealInitializationSucceeds() {
        // Zero rating
        let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
        XCTAssertNotNil(zeroRatingMeal)

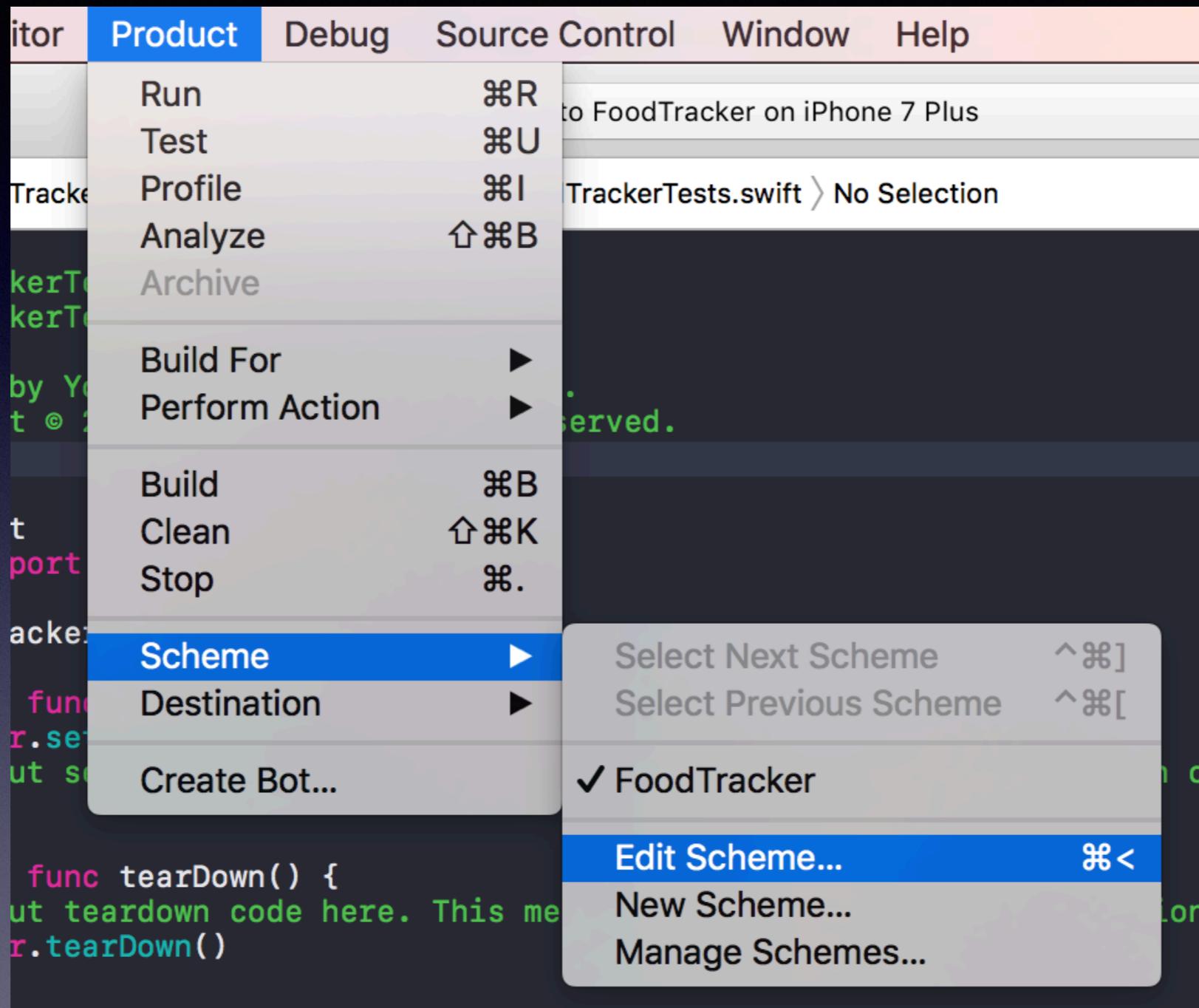
        // Highest positive rating
        let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
        XCTAssertNotNil(positiveRatingMeal)
    }

    func testMealInitializationFails() {
        self.measure {
            // Negative rating
            let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
            XCTAssertNil(negativeRatingMeal)

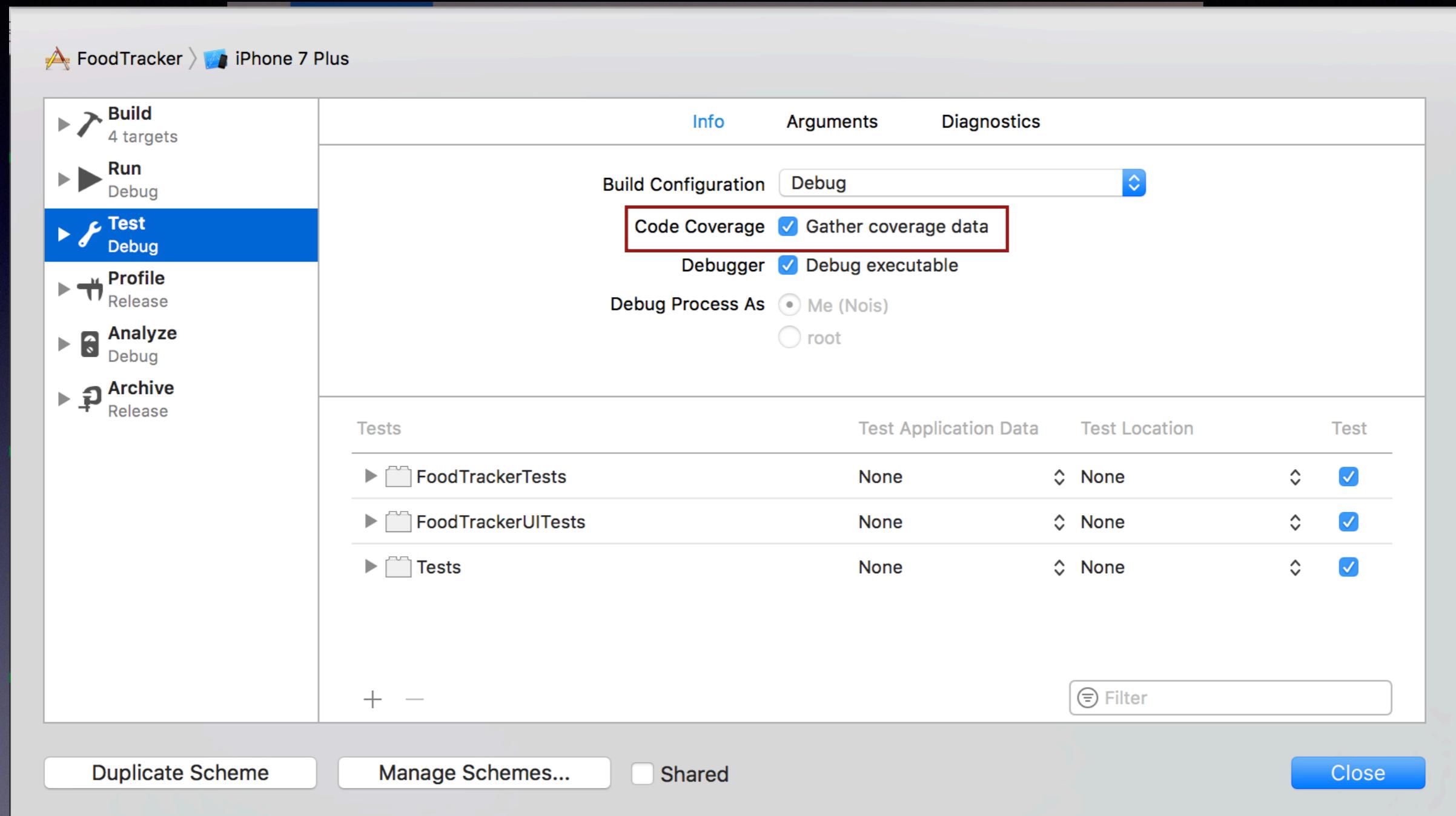
            // Empty String
            let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
            XCTAssertNil(emptyStringMeal)

            // Rating exceeds maximum
            let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
            XCTAssertNil(largeRatingMeal)
        }
    }
}
```

TDD Unit Test 적용해보기



TDD Unit Test 적용해보기



TDD Unit Test 적용해보기

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

FoodTracker > iPhone 7 Plus Finished running FoodTracker on iPhone 7 Plus A (수) 오후 4:40

By Group By Time

Debug FoodTracker 오늘 오후 4:29
Debug FoodTracker 오늘 오후 4:29
Test FoodTracker 오늘 오후 4:28
Build FoodTracker 오늘 오후 4:28
Debug FoodTracker 오늘 오후 4:26
Debug FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:25
Build FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:24
Build FoodTracker 오늘 오후 4:24
Debug FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:12
Test FoodTracker 오늘 오후 3:35
Build FoodTracker 오늘 오후 3:35
Debug FoodTracker 오늘 오후 3:16
Update Signing 오늘 오후 2:38

ViewController.swift

```
5 // Created by Yoo Seok Kim on 2017. 7. 1..
6 // Copyright © 2017년 Nois. All rights reserved.
7 //
8 import UIKit
9
10 class ViewController: UIViewController, UITextFieldDelegate, UIImagePickerControllerDelegate, UINavigationControllerDelegate {
11
12     // MARK: properties
13     @IBOutlet weak var nameTextField: UITextField!
14     @IBOutlet weak var mealNameLabel: UILabel!
15     @IBOutlet weak var photoImageView: UIImageView!
16     @IBOutlet weak var ratingControl: RatingControl!
17
18     override func viewDidLoad() {
19         super.viewDidLoad()
20
21         // Handle the text field's user input through delegate callbacks.
22         nameTextField.delegate = self
23     }
24
25     // MARK: UITextFieldDelegate
26     func textFieldShouldReturn(_ textField: UITextField) -> Bool {
27         // Hide the keyboard.
28         textField.resignFirstResponder()
29         return true
30     }
31
32     func textFieldDidEndEditing(_ textField: UITextField) {
33         mealNameLabel.text = textField.text
34     }
35
36     // MARK: Actions
37     func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {
38         // Dismiss the picker if the user canceled.
39         dismiss(animated: true, completion: nil)
40     }
41
42     func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
43
44         // The info dictionary may contain multiple representations of the image. You want to use the original.
45         guard let selectedImage = info[UIImagePickerControllerOriginalImage] as? UIImage else {
46             fatalError("Expected a dictionary containing an image, but was provided the following: \(info)")
47         }
48
49         // Set photoImageView to display the selected image.
50         photoImageView.image = selectedImage
51
52         // Dismiss the picker.
53         dismiss(animated: true, completion: nil)
54     }
55
56     @IBAction func selectImageFromPhotoLibrary(_ sender: UITapGestureRecognizer) {
57
58         // Hide the keyboard.
59         nameTextField.resignFirstResponder()
60
61         // UIImagePickerController is a view controller that lets a user pick media from their photo library.
62     }
}
```

TDD Unit Test 적용해보기

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

FoodTracker > iPhone 7 Plus Finished running FoodTracker on iPhone 7 Plus

By Group By Time

Debug FoodTracker 오늘 오후 4:29
Debug FoodTracker 오늘 오후 4:29
Test FoodTracker 오늘 오후 4:28
Build FoodTracker 오늘 오후 4:28
Debug FoodTracker 오늘 오후 4:26
Debug FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:25
Build FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:24
Build FoodTracker 오늘 오후 4:24
Debug FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:12
Test FoodTracker 오늘 오후 3:35
Build FoodTracker 오늘 오후 3:35
Debug FoodTracker 오늘 오후 3:16
Update Signing 오늘 오후 2:38

ViewController.swift

```
5 // Created by Yoo Seok Kim on 2017. 7. 1..  
6 // Copyright © 2017년 Nois. All rights reserved.  
7 //  
8 import UIKit  
9  
10 class ViewController: UIViewController, UITextFieldDelegate, UIImagePickerControllerDelegate, UINavigationControllerDelegate {  
11  
12     // MARK: properties  
13     @IBOutlet weak var nameTextField: UITextField!  
14     @IBOutlet weak var mealNameLabel: UILabel!  
15     @IBOutlet weak var photoImageView: UIImageView!  
16     @IBOutlet weak var ratingControl: RatingControl!  
17  
18     override func viewDidLoad() {  
19         super.viewDidLoad()  
20  
21         // Handle the text field's user input through delegate callbacks.  
22         nameTextField.delegate = self  
23     }  
24  
25     // MARK: UITextFieldDelegate  
26     func textFieldShouldReturn(_ textField: UITextField) -> Bool {  
27         // Hide the keyboard.  
28         textField.resignFirstResponder()  
29         return true  
30     }  
31  
32     func textFieldDidEndEditing(_ textField: UITextField) {  
33         mealNameLabel.text = textField.text  
34     }  
35  
36     // MARK: Actions  
37     func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {  
38         // Dismiss the picker if the user canceled.  
39         dismiss(animated: true, completion: nil)  
40     }  
41  
42     func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {  
43  
44         // The info dictionary may contain multiple representations of the image. You want to use the original.  
45         guard let selectedImage = info[UIImagePickerControllerOriginalImage] as? UIImage else {  
46             fatalError("Expected a dictionary containing an image, but was provided the following: \(info)")  
47         }  
48  
49         // Set photoImageView to display the selected image.  
50         photoImageView.image = selectedImage  
51  
52         // Dismiss the picker.  
53         dismiss(animated: true, completion: nil)  
54     }  
55  
56     @IBAction func selectImageFromPhotoLibrary(_ sender: UITapGestureRecognizer) {  
57  
58         // Hide the keyboard.  
59         nameTextField.resignFirstResponder()  
60  
61         // UIImagePickerController is a view controller that lets a user pick media from their photo library.  
62 }
```

TDD Unit Test 적용해보기

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

FoodTracker > iPhone 7 Plus Finished running FoodTracker on iPhone 7 Plus A (수) 오후 4:40

By Group By Time

Debug FoodTracker 오늘 오후 4:29
Debug FoodTracker 오늘 오후 4:29
Test FoodTracker 오늘 오후 4:28
Build FoodTracker 오늘 오후 4:28
Debug FoodTracker 오늘 오후 4:26
Debug FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:25
Build FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:24
Build FoodTracker 오늘 오후 4:24
Debug FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:12
Test FoodTracker 오늘 오후 3:35
Build FoodTracker 오늘 오후 3:35
Debug FoodTracker 오늘 오후 3:16
Update Signing 오늘 오후 2:38

```
5 // Created by Yoo Seok Kim on 2017. 7. 1..  
6 // Copyright © 2017년 Nois. All rights reserved.  
7 //  
8 import UIKit  
9  
10 class ViewController: UIViewController, UITextFieldDelegate, UIImagePickerControllerDelegate, UINavigationControllerDelegate {  
11  
12     // MARK: properties  
13     @IBOutlet weak var nameTextField: UITextField!  
14     @IBOutlet weak var mealNameLabel: UILabel!  
15     @IBOutlet weak var photoImageView: UIImageView!  
16     @IBOutlet weak var ratingControl: RatingControl!  
17  
18     override func viewDidLoad() {  
19         super.viewDidLoad()  
20  
21         // Handle the text field's user input through delegate callbacks.  
22         nameTextField.delegate = self  
23     }  
24  
25     // MARK: UITextFieldDelegate  
26     func textFieldShouldReturn(_ textField: UITextField) -> Bool {  
27         // Hide the keyboard.  
28         textField.resignFirstResponder()  
29         return true  
30     }  
31  
32     func textFieldDidEndEditing(_ textField: UITextField) {  
33         mealNameLabel.text = textField.text  
34     }  
35  
36     // MARK: Actions  
37     func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {  
38         // Dismiss the picker if the user canceled.  
39         dismiss(animated: true, completion: nil)  
40     }  
41  
42     func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {  
43  
44         // The info dictionary may contain multiple representations of the image. You want to use the original.  
45         guard let selectedImage = info[UIImagePickerControllerOriginalImage] as? UIImage else {  
46             fatalError("Expected a dictionary containing an image, but was provided the following: \(info)")  
47         }  
48  
49         // Set photoImageView to display the selected image.  
50         photoImageView.image = selectedImage  
51  
52         // Dismiss the picker.  
53         dismiss(animated: true, completion: nil)  
54     }  
55  
56     @IBAction func selectImageFromPhotoLibrary(_ sender: UITapGestureRecognizer) {  
57  
58         // Hide the keyboard.  
59         nameTextField.resignFirstResponder()  
60  
61         // UIImagePickerController is a view controller that lets a user pick media from their photo library.
```

TDD Unit Test 적용해보기

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

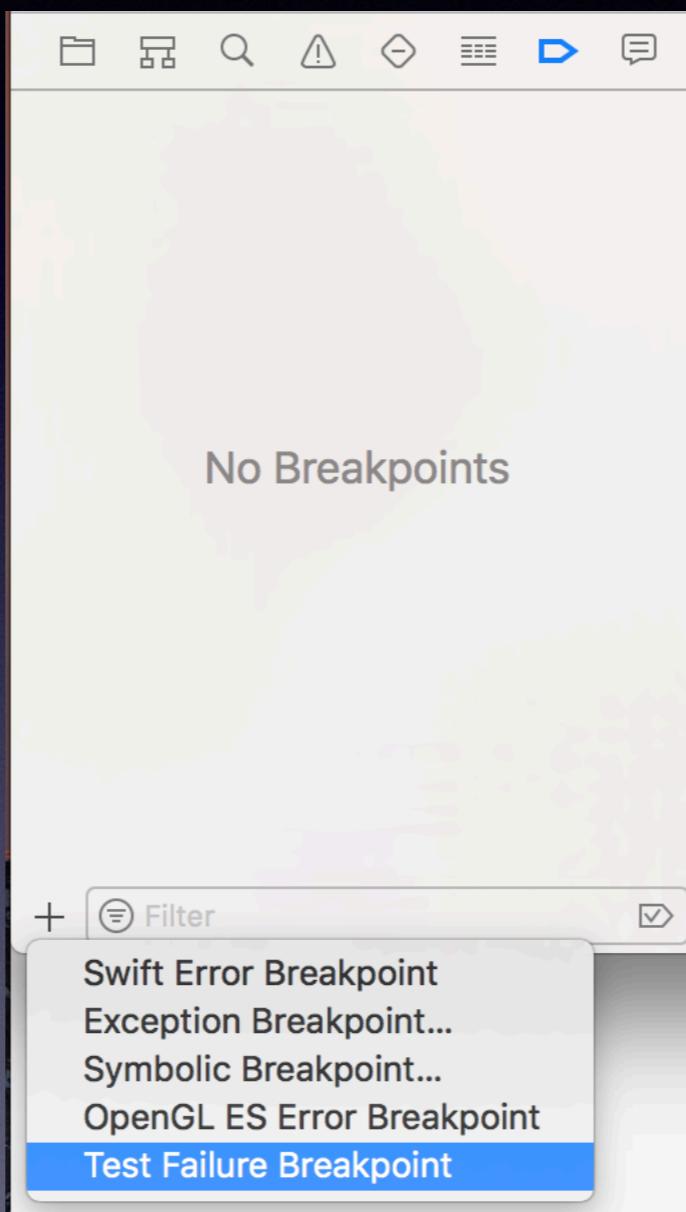
FoodTracker > iPhone 7 Plus Finished running FoodTracker on iPhone 7 Plus

By Group By Time

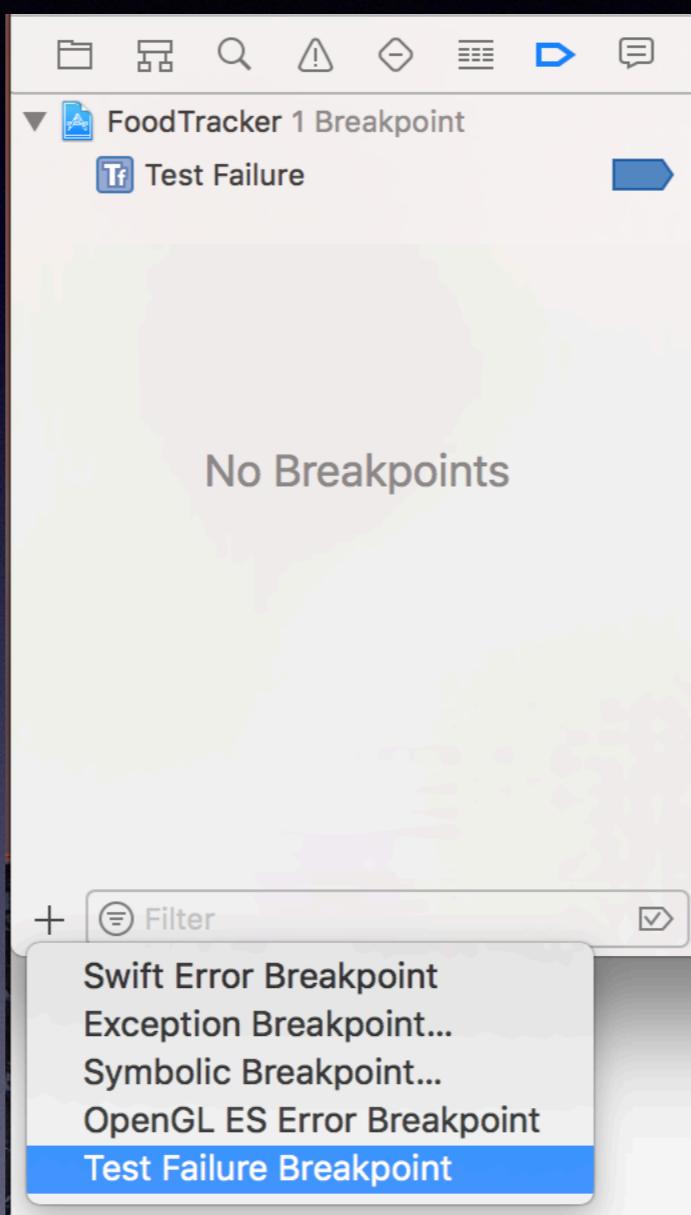
Debug FoodTracker 오늘 오후 4:29
Debug FoodTracker 오늘 오후 4:29
Test FoodTracker 오늘 오후 4:28
Build FoodTracker 오늘 오후 4:28
Debug FoodTracker 오늘 오후 4:26
Debug FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:25
Build FoodTracker 오늘 오후 4:25
Test FoodTracker 오늘 오후 4:24
Build FoodTracker 오늘 오후 4:24
Debug FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:23
Build FoodTracker 오늘 오후 4:12
Test FoodTracker 오늘 오후 3:35
Build FoodTracker 오늘 오후 3:35
Debug FoodTracker 오늘 오후 3:16
Update Signing 오늘 오후 2:38

```
5 // Created by Yoo Seok Kim on 2017. 7. 1..  
6 // Copyright © 2017년 Nois. All rights reserved.  
7 //  
8 import UIKit  
9  
10 class ViewController: UIViewController, UITextFieldDelegate, UIImagePickerControllerDelegate, UINavigationControllerDelegate {  
11  
12     // MARK: properties  
13     @IBOutlet weak var nameTextField: UITextField!  
14     @IBOutlet weak var mealNameLabel: UILabel!  
15     @IBOutlet weak var photoImageView: UIImageView!  
16     @IBOutlet weak var ratingControl: RatingControl!  
17  
18     override func viewDidLoad() {  
19         super.viewDidLoad()  
20  
21         // Handle the text field's user input through delegate callbacks.  
22         nameTextField.delegate = self  
23     }  
24  
25     // MARK: UITextFieldDelegate  
26     func textFieldShouldReturn(_ textField: UITextField) -> Bool {  
27         // Hide the keyboard.  
28         textField.resignFirstResponder()  
29         return true  
30     }  
31  
32     func textFieldDidEndEditing(_ textField: UITextField) {  
33         mealNameLabel.text = textField.text  
34     }  
35  
36     // MARK: Actions  
37     func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {  
38         // Dismiss the picker if the user canceled.  
39         dismiss(animated: true, completion: nil)  
40     }  
41  
42     func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {  
43  
44         // The info dictionary may contain multiple representations of the image. You want to use the original.  
45         guard let selectedImage = info[UIImagePickerControllerOriginalImage] as? UIImage else {  
46             fatalError("Expected a dictionary containing an image, but was provided the following: \(info)")  
47         }  
48  
49         // Set photoImageView to display the selected image.  
50         photoImageView.image = selectedImage  
51  
52         // Dismiss the picker.  
53         dismiss(animated: true, completion: nil)  
54     }  
55  
56     @IBAction func selectImageFromPhotoLibrary(_ sender: UITapGestureRecognizer) {  
57  
58         // Hide the keyboard.  
59         nameTextField.resignFirstResponder()  
60  
61         // UIImagePickerController is a view controller that lets a user pick media from their photo library.
```

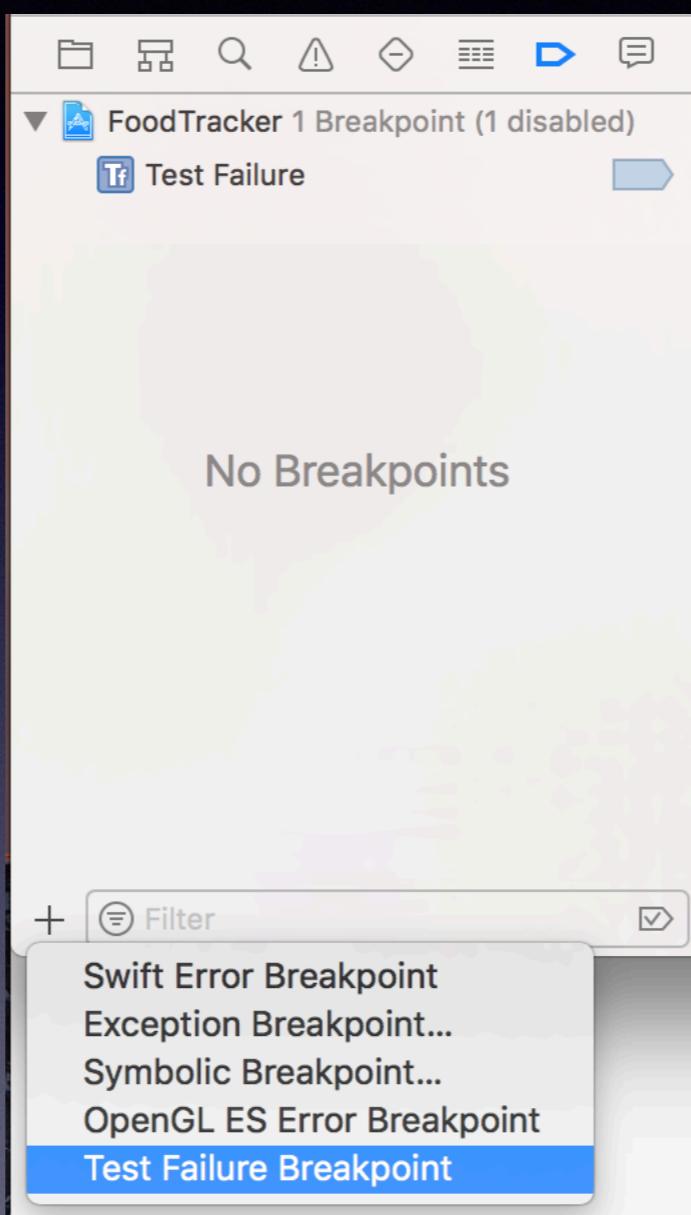
TDD Unit Test 적용해보기



TDD Unit Test 적용해보기



TDD Unit Test 적용해보기



TDD Unit Test 적용해보기

The screenshot shows the Xcode interface during a unit test session for the FoodTracker project. The top menu bar includes File, Edit, View, Find, Navigate, Editor, Product, Debug, Source Control, Window, and Help. The title bar indicates the project is 'FoodTracker' and the target is 'iPhone 7 Plus'. A status bar at the top right shows the date and time as '(수) 오후 4:47'.

The left sidebar displays the 'FoodTrackerTests' process (PID 57559) with various monitoring tabs: CPU, Memory, Disk, and Network, all showing 'Zero KB/s'. Below these are threads: Thread 1 (selected), Thread 2, Thread 3, Thread 4, com.apple.uikit.eventfetch-thread, Thread 7, Thread 8, Thread 9, and Thread 10.

The main editor area shows the code for `FoodTrackerTests.swift`:

```
8
9 import XCTest
10 @testable import FoodTracker
11
12 class FoodTrackerTests: XCTestCase {
13
14     override func setUp() {
15         super.setUp()
16         // Put setup code here. This method is called before the invocation of each test method in the class.
17     }
18
19     override func tearDown() {
20         // Put teardown code here. This method is called after the invocation of each test method in the class.
21         super.tearDown()
22     }
23
24     // Confirm that the Meal initializer returns a Meal object when passed valid parameters.
25     func testMealInitializationSucceeds() {
26         // Zero rating
27         let zeroRatingMeal = Meal.init(name: "Zero", photo: nil, rating: 0)
28         XCTAssertNotNil(zeroRatingMeal)
29
30         // Highest positive rating
31         let positiveRatingMeal = Meal.init(name: "Positive", photo: nil, rating: 5)
32         XCTAssertNotNil(positiveRatingMeal)
33     }
34
35     func testMealInitializationFails() {
36         self.measure {
37             // Negative rating
38             let negativeRatingMeal = Meal.init(name: "Negative", photo: nil, rating: -1)
39             XCTAssertNil(negativeRatingMeal)
40
41             // Empty String
42             let emptyStringMeal = Meal.init(name: "", photo: nil, rating: 0)
43             XCTAssertNil(emptyStringMeal)
44
45             // Rating exceeds maximum
46             let largeRatingMeal = Meal.init(name: "Large", photo: nil, rating: 6)
47             XCTAssertNil(largeRatingMeal)
48         }
49     }
50 }

```

A tooltip 'Thread 1: Test Failure' is visible near the bottom of the code editor. The status bar at the bottom right shows 'Time: 0.000 sec (78% better)'.

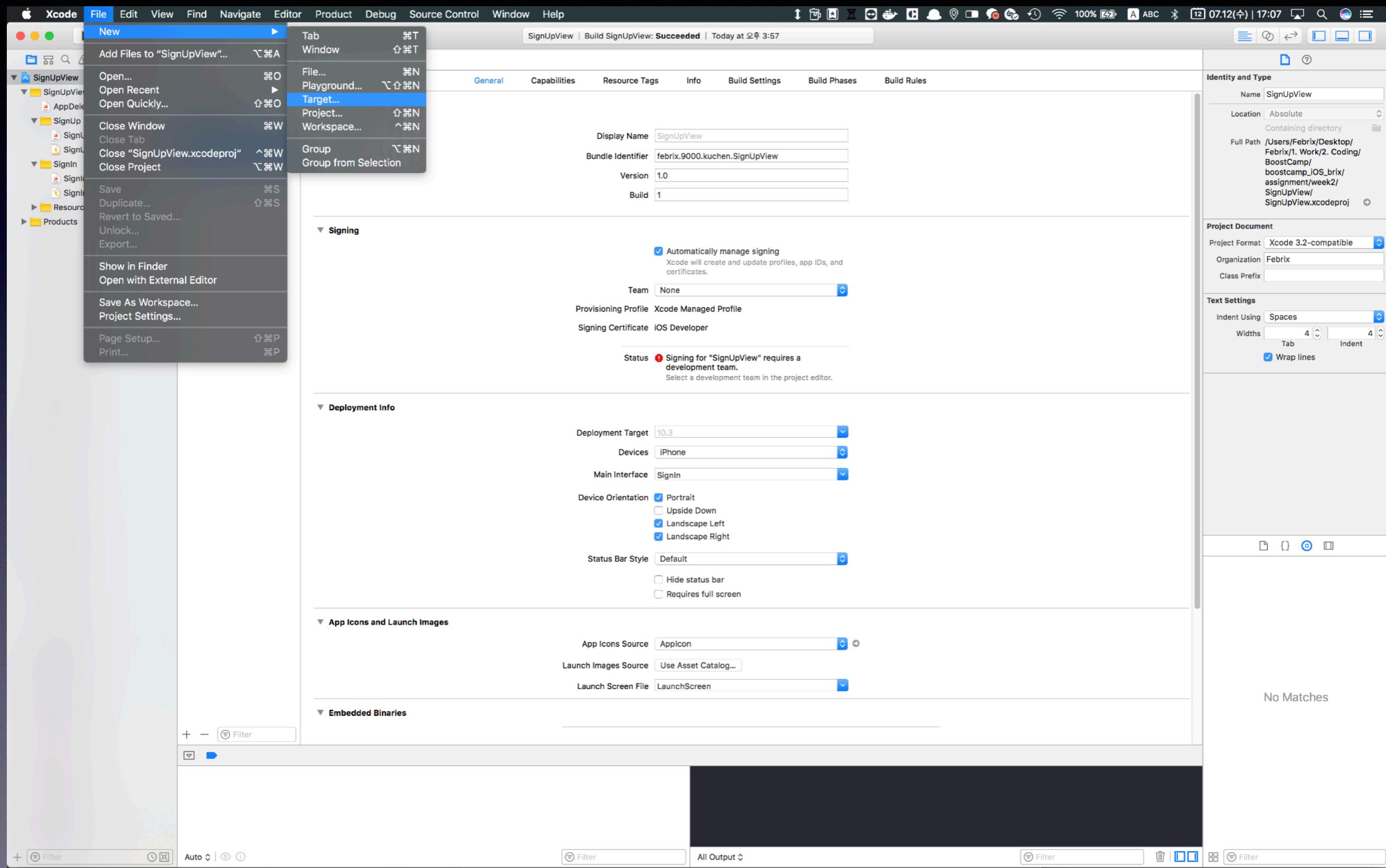
The bottom section shows the debugger stack trace for Thread 1, specifically for the failing test case:

```
negativeRatingMeal = (FoodTracker.Meal?) nil
emptyStringMeal = (FoodTracker.Meal?) nil
largeRatingMeal = (FoodTracker.Meal?) 0x000060800026ec00
name = (String) "Large"
photo = (UIImage?) nil
rating = (Int) 6

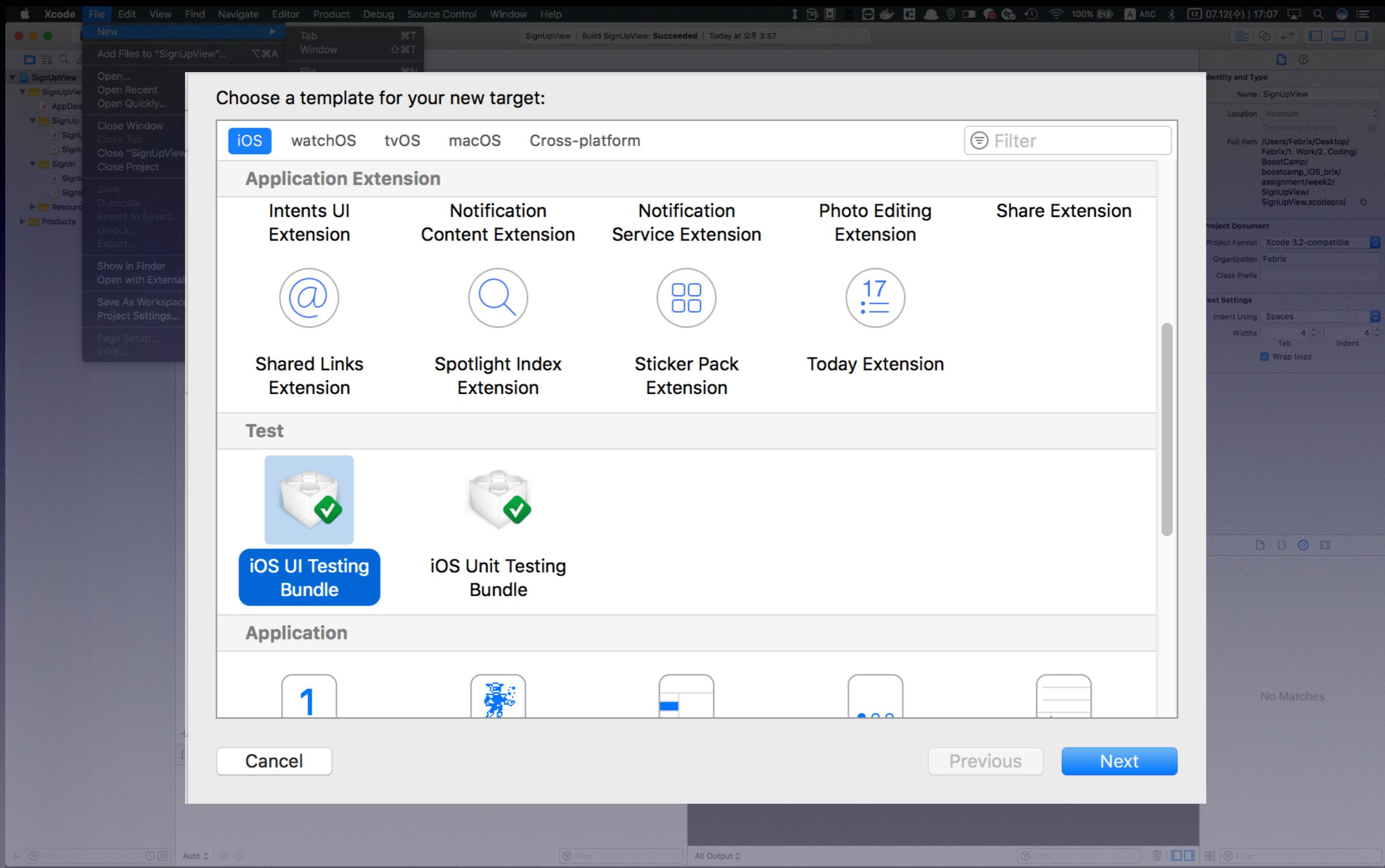
bb
Test Suite 'Selected tests' started at 2017-07-12 16:46:37.980
Test Suite 'FoodTrackerTests.xctest' started at 2017-07-12 16:46:37.981
Test Suite 'FoodTrackerTests' started at 2017-07-12 16:46:37.982
Test Case '-[FoodTrackerTests.FoodTrackerTests testMealInitializationFails]' started.
(lldb)
```

At the very bottom, there are three filter dropdowns labeled 'Auto', 'All Output', and another 'All Output'.

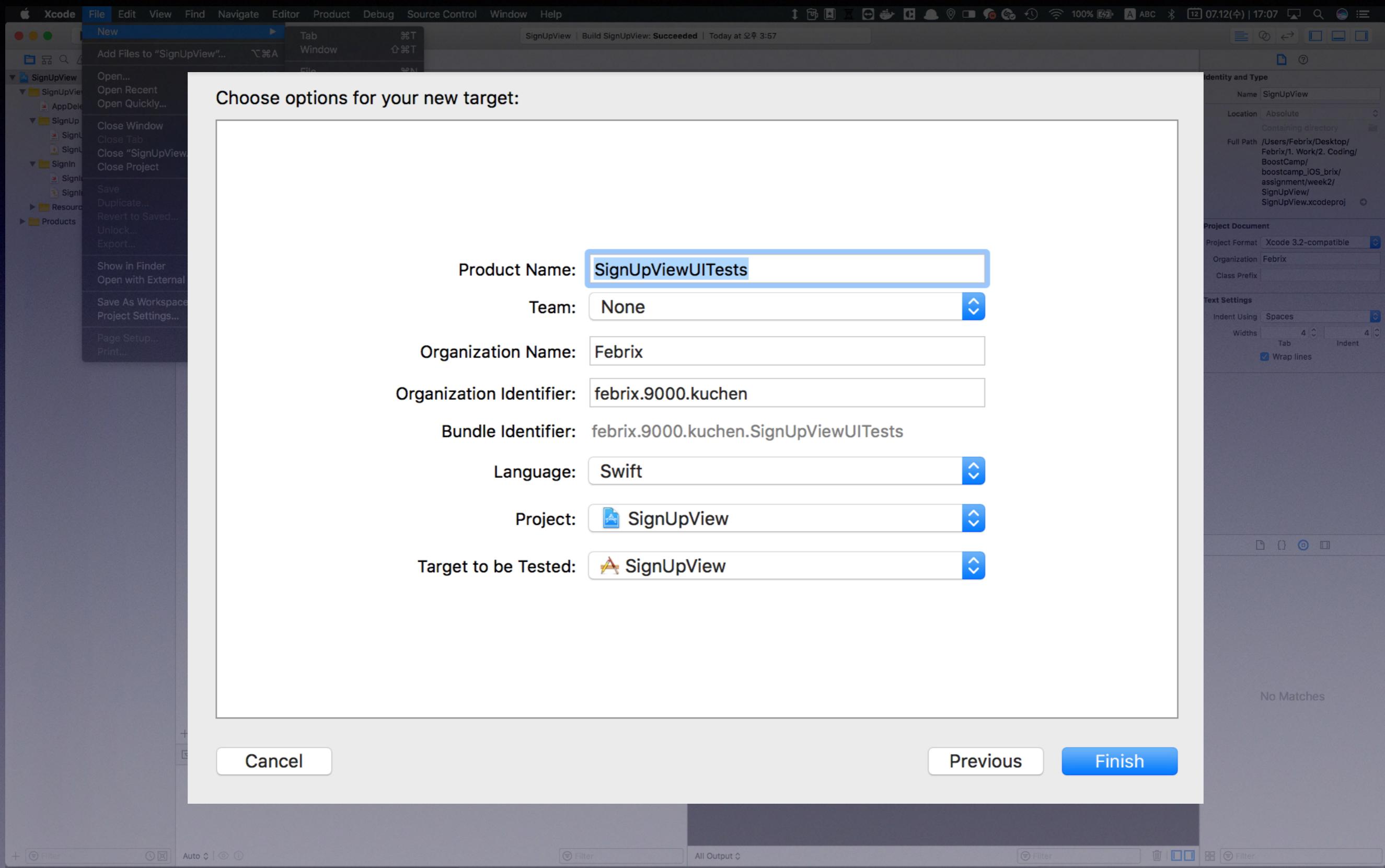
TDD UI Test 적용해보기



TDD UI Test 적용해보기



TDD UI Test 적용해보기



TDD UI Test 적용해보기

The screenshot shows the Xcode interface with the following details:

- Project Structure:** The project is named "SignUpView". Inside it, there is a "SignUpViewUITests" folder containing a file named "SignUpViewUITests.swift".
- File Path:** The current file path is "SignUpView > SignUpViewUITests > SignUpViewUITests.swift".
- Code Editor:** The main window displays the "SignUpViewUITests.swift" file.
- Code Content:** The code is a Swift file for UI testing. It includes imports for XCTest and defines a class `SignUpViewUITests` that inherits from `XCTestCase`. It overrides `setUp()` and `tearDown()` methods and contains a test function `testExample()`.
- Code Lines:** The code spans from line 1 to line 38.
- Toolbars:** The bottom toolbar shows standard Xcode icons for file operations, including a red circle and a blue arrow.

```
// SignUpViewUITests.swift
// SignUpViewUITests
//
// Created by Febrix on 2017. 7. 12..
// Copyright © 2017년 Febrix. All rights reserved.

import XCTest

class SignUpViewUITests: XCTestCase {

    override func setUp() {
        super.setUp()

        // Put setup code here. This method is called before the invocation of each test method.

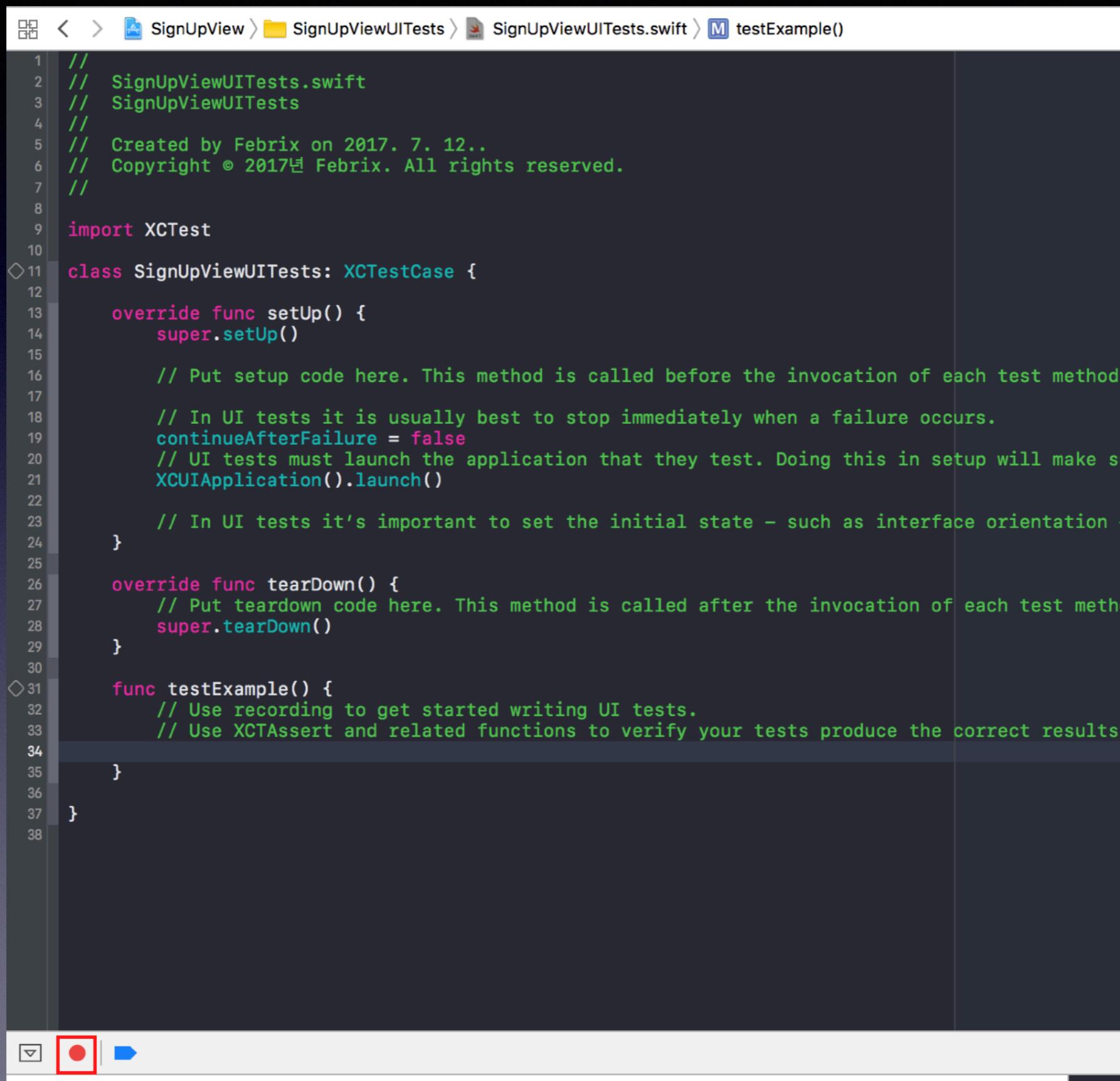
        // In UI tests it is usually best to stop immediately when a failure occurs.
        continueAfterFailure = false
        // UI tests must launch the application that they test. Doing this in setup will make sure it happens for each test method.
        XCUIApplication().launch()

        // In UI tests it's important to set the initial state – such as interface orientation – before each test.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method.
        super.tearDown()
    }

    func testExample() {
        // Use recording to get started writing UI tests.
        // Use XCTAssert and related functions to verify your tests produce the correct results.
    }
}
```

TDD UI Test 적용해보기



```
 SignUpView < > SignUpViewUITests > SignUpViewUITests.swift > testExample()

1 // 
2 //  SignUpViewUITests.swift
3 //  SignUpViewUITests
4 //
5 //  Created by Febrix on 2017. 7. 12..
6 //  Copyright © 2017년 Febrix. All rights reserved.
7 //

8 import XCTest
9
10 class SignUpViewUITests: XCTestCase {
11
12     override func setUp() {
13         super.setUp()
14
15         // Put setup code here. This method is called before the invocation of each test method.
16
17         // In UI tests it is usually best to stop immediately when a failure occurs.
18         continueAfterFailure = false
19         // UI tests must launch the application that they test. Doing this in setup will make sure it happens for
20         XCUIApplication().launch()
21
22         // In UI tests it's important to set the initial state – such as interface orientation – before each test
23     }
24
25
26     override func tearDown() {
27         // Put teardown code here. This method is called after the invocation of each test method.
28         super.tearDown()
29     }
30
31     func testExample() {
32         // Use recording to get started writing UI tests.
33         // Use XCTAssert and related functions to verify your tests produce the correct results
34     }
35
36
37 }
38
```

TDD UI Test 적용해보기

```
30
31 func testExample() {
32     // Use recording to get started writing UI tests.
33     // Use XCTAssert and related functions to verify your tests produce the correct results.
34     XCUIDevice.shared().orientation = .portrait
35     XCUIDevice.shared().orientation = .portrait
36     XCUIDevice.shared().orientation = .faceUp
37     XCUIDevice.shared().orientation = .portrait
38
39     let app = XCUIApplication()
40     let idTextField = app.textFields["ID"]
41     idTextField.tap()
42     idTextField.typeText("test")
43
44     let passwordSecureTextField = app.secureTextFields["Password"]
45     passwordSecureTextField.tap()
46
47     let moreKey = app.keys["more"]▼
48     moreKey.tap()
49     moreKey.tap()
50     passwordSecureTextField.typeText("1234")
51     app.buttons["Sign In"].tap()
52
53 }
54 }
55 }
```

TDD UI Test 적용해보기

UI Recording

The screenshot shows the Xcode interface with the following details:

- Title Bar:** Shows "SignUpView" and "iPhone 7".
- Toolbar:** Standard Xcode toolbar with icons for play, stop, record, and others.
- Project Navigator:** Shows the project structure under "SignUpView".
- Editor:** Displays the "SignUpViewUITests.swift" file content.
- Output Navigator:** Shows three tabs: "All Output", "Filter", and another "Filter".

```
// SignUpViewUITests.swift
// SignUpViewUITests
//
// Created by Febrix on 2017. 7. 12..
// Copyright © 2017W Febrix. All rights reserved.
//
import XCTest

class SignUpViewUITests: XCTestCase {

    override func setUp() {
        super.setUp()

        // Put setup code here. This method is called before the invocation of each test method in the class.
        continueAfterFailure = false
        // UI tests must launch the application that they test. Doing this in setup will make sure it happens for each test method.
        XCUIApplication().launch()

        // In UI tests it's important to set the initial state - such as interface orientation - required for your tests before they run. The
        // setUp method is a good place to do this.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    func testExample() {
        // Use recording to get started writing UI tests.
        // Use XCTAssert and related functions to verify your tests produce the correct results.

        |
    }
}
```

TDD UI Test 적용해보기

Test

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure for "SignUpView". It includes files like AppDelegate.swift, SignUpViewController.swift, SignUp.storyboard, SignInViewController.swift, SignIn.storyboard, and Info.plist.
- Editor:** Displays the code for "SignUpViewUITests.swift". The code defines a UI test case with setup and teardown methods, and a specific test method named "testExample()". The "testExample()" method interacts with the application's UI elements (text fields and buttons) using XCUIElement API.
- Run Bar:** Shows the target as "SignUpView" and the device as "iPhone 7".
- Output Navigator:** Shows the output of the test run. It includes a log message indicating the configuration profile path and a timestamp. Below that, it shows the tap operation on the "Sign In" button and the entered text "ID : test, PW : 1234".

```
// Created by Febrix on 2017. 7. 12..
// Copyright © 2017 Febrix. All rights reserved.

import XCTest

class SignUpViewUITests: XCTestCase {

    override func setUp() {
        super.setUp()

        // Put setup code here. This method is called before the invocation of each test method in the class.
        continueAfterFailure = false
        // UI tests must launch the application that they test. Doing this in setup will make sure it happens for each test method.
        XCUIApplication().launch()

        // In UI tests it's important to set the initial state - such as interface orientation - required for your tests before they run. The
        // setup method is a good place to do this.
    }

    override func tearDown() {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        super.tearDown()
    }

    func testExample() {
        // Use recording to get started writing UI tests.
        // Use XCTAssert and related functions to verify your tests produce the correct results.

        let app = XCUIApplication()
        let idTextField = app.textFields["ID"]
        idTextField.tap()
        idTextField.typeText("test")

        let passwordSecureTextField = app.secureTextFields["Password"]
        passwordSecureTextField.tap()
        passwordSecureTextField.tap()
        passwordSecureTextField.typeText("1234")
        app.buttons["Sign In"].tap()
    }
}
```

```
container for systemgroup.com.apple.configurationprofiles path is /Users/Febr...  
2017-07-12 17:44:17.668729+0900 SignUpView[33948:9376547] [NC] Reading from  
private effective user settings.  
touch up inside - sign in  
ID : test, PW : 1234
```

TDD의 장단점

장점

작성한 코드가 테스트를 반드시 통과해야 하므로 소스코드의 품질이 향상되고 디버깅 시간이 절약된다.
또한 유지보수와 리팩토링이 용이해진다.

단점

프로토타이핑에 부적합하다.
테스트 코드를 작성하는 비용이 듈다.

감사합니다