

Testing with IOS

1. Setup a UI Test
2. Do a Code Coverage Test

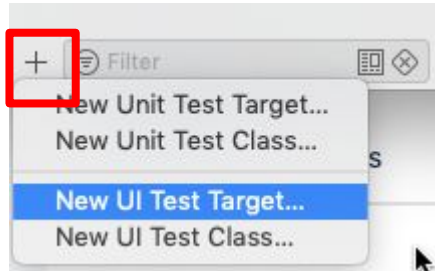
UI Testing

Setup

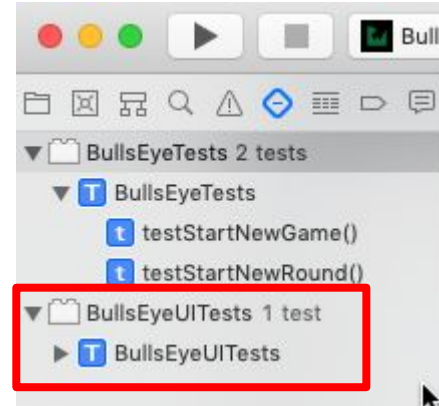
1. Go to Test Navigator window



2. Add new Unit Test



3. Expected Result



Process

1. Use the RECORDER to get the names of the elements
 - a. Click on an empty test case
 - b. Press the RECORD button
 - c. When emulator opens, click on the elements you want
 - d. Press the STOP button
 - e. Names of elements will appear in function
2. Write your test cases using XCTAssert functions

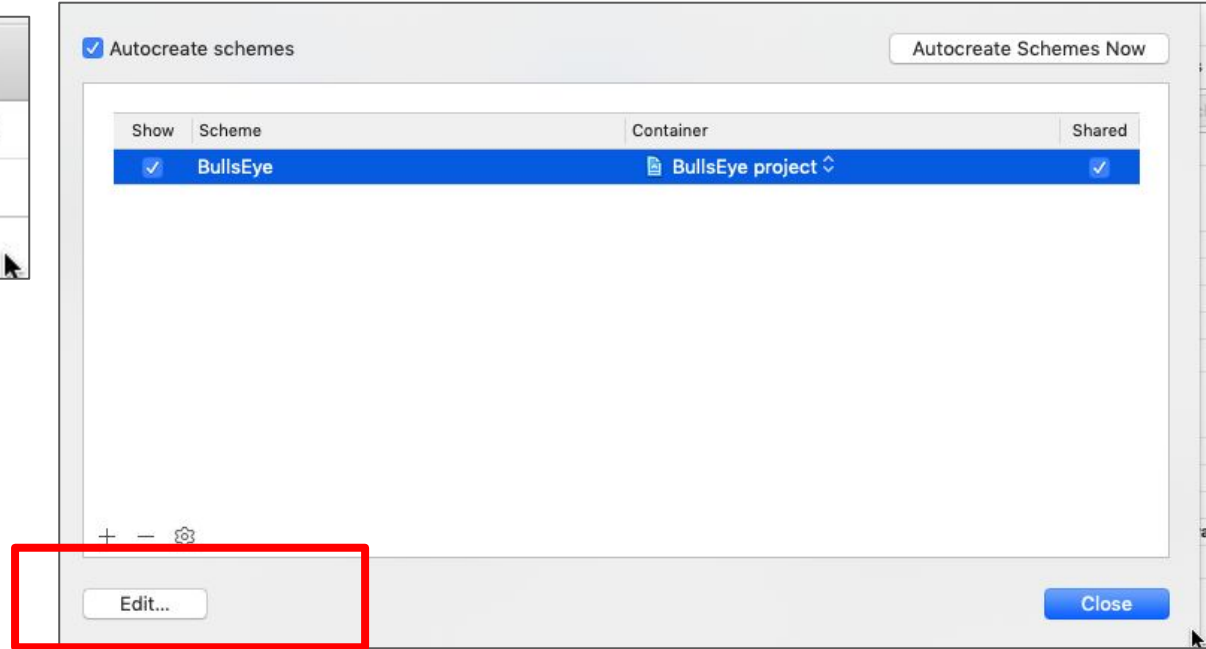
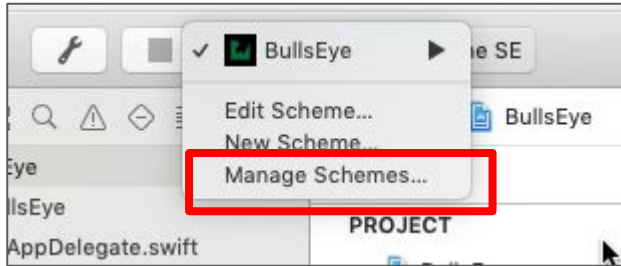


```
// Score and Round label text  
func testStartOverButton() {  
|  
}  
}
```



Code Coverage Tools

Activate the Code Coverage screen



Info

Arguments

Options

Diagnostics

▶ **Build**
3 targets

▶ **Run**
Debug

▶ **Test**
Debug

▶ **Profile**
Release

▶ **Analyze**
Debug

▶ **Archive**
Release

Application Language System Language

Application Region System Region

UI Testing ☒ Capture screenshots automatically
☒ Delete when each test succeeds

Attachments ☒ Delete when each test succeeds

Code Coverage ☒ Gather coverage for all targets

Duplicate Scheme

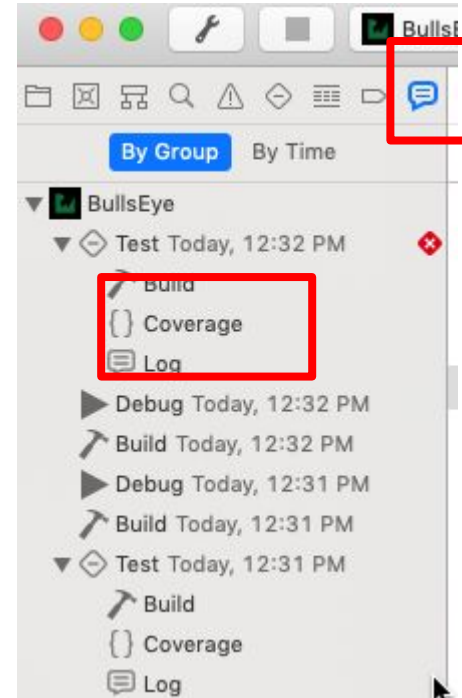
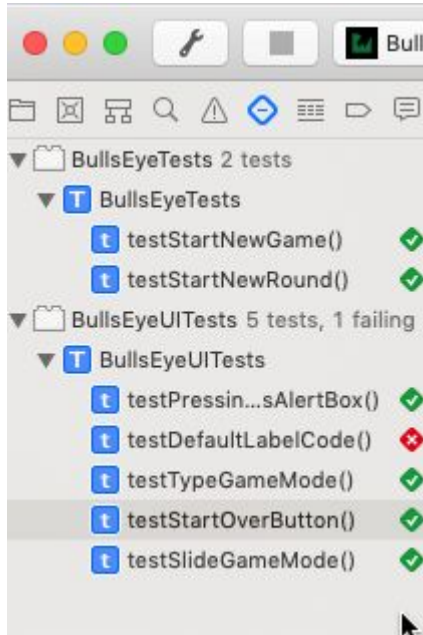
Manage Schemes...

☒ Shared

Close













Expected Result

Click these buttons to open Coverage window



Expected Result

You will see a summary of which files and functions were tested by your test cases.

Name	Coverage
▼ BullsEye.app	55.5%
▼ BullsEyeGame.swift	61.9%
	100.0%
 BullsEye.BullsEyeGame.startNewGame() -> ()	100.0%
 BullsEye.BullsEyeGame.startNewRound() -> ()	100.0%
 BullsEye.BullsEyeGame.check(guess: Swift.Int) -> Swift.Int	0.0%
▼ ViewController.swift	53.9%
 BullsEye.ViewController.viewDidLoad() -> ()	85.7%
 BullsEye.ViewController.chooseGameStyle(_C.UISegmentedControl) -> ()	0.0%
 BullsEye.ViewController.updateView() -> ()	75.0%
 BullsEye.ViewController.checkGuess(Any) -> ()	86.7%
 BullsEye.ViewController.showScoreAlert(difference: Swift.Int) -> ()	0.0%
 closure #1 (_C.UIAlertAction) -> () in BullsEye.ViewController.showScoreA	0.0%
 BullsEye.ViewController.showNaNAlert() -> ()	100.0%
 BullsEye.ViewController.startOver(Any) -> ()	0.0%

Coverage also shows the functions that are NOT tested by your TCs

Name	Coverage
▼ BullsEye.app	<div><div></div></div> 55.5%
▼ BullsEyeGame.swift	<div><div></div></div> 61.9%
📄 BullsEye.BullsEyeGame.startNewGame() -> ()	<div><div></div></div> 100.0%
📄 BullsEye.BullsEyeGame.startNewRound() -> ()	<div><div></div></div> 100.0%
📄 BullsEye.BullsEyeGame.check(guess: Swift.Int) -> Swift.Int	<div><div></div></div> 0.0%
▼ ViewController.swift	<div><div></div></div> 53.9%
📄 BullsEye.ViewController.viewDidLoad() -> ()	<div><div></div></div> 85.7%
📄 BullsEye.ViewController.chooseGameStyle(__C.UISegmentedControl) -> ()	<div><div></div></div> 0.0%
📄 BullsEye.ViewController.updateView() -> ()	<div><div></div></div> 75.0%
📄 BullsEye.ViewController.checkGuess(Any) -> ()	<div><div></div></div> 86.7%
📄 BullsEye.ViewController.showScoreAlert(difference: Swift.Int) -> ()	<div><div></div></div> 0.0%
📄 closure #1 (__C.UIAlertAction) -> () in BullsEye.ViewController.showScoreA	<div><div></div></div> 0.0%
📄 BullsEye.ViewController.showNaNAlert() -> ()	<div><div></div></div> 100.0%
📄 BullsEye.ViewController.startOver(Any) -> ()	<div><div></div></div> 0.0%

To fix coverage, add tests cases to cover the missing functions

Example - Write an automated test case to test the check() function

(check() is in BullsEyeGame.swift)

- Note - this is a nonsense test case. We are using it just to show that adding tests cases will increase coverage.

```
func testCheckFunction() {  
    let diff = game.check(guess: 5)  
    XCTAssertEqual(95, diff)  
}
```

Rerun tests and look at new coverage

Test Coverage will increase to 62.7%

And you get 100% coverage on the check() function

