Asynchrone Tests

XCTestExpectation

```
func test_asynTask_waitForList() {
  let asyncExpectation = expectation(description: "async")

  var result: Bool? = nil
  sut.asyncTask { success in
    result = success
    asyncExpectation.fulfill()
  }

  wait(for: [asyncExpectation], timeout: 1)
  XCTAssertTrue(result ?? false)
}
```

XCTKVOExpectation

```
func test_kvo() {
  let kvoExpectation =
    keyValueObservingExpectation(
      for: sut,
      keyPath: "name",
      expectedValue: "Foo Bar")

  sut.firstname = "Foo"
  sut.lastname = "Bar"

  wait(for: [kvoExpectation], timeout: 1)
}
```

XCTKVOExpectation With Handler

```
func test_kvo() {
  let kvoExpectation =
    keyValueObservingExpectation(
      for: sut as Any,
      keyPath: "name") { observed, changes -> Bool in

        guard let new = changes["new"] as? String else {
            return false
        }

        return new == "Foo Bar"
  }

  sut.firstname = "Foo"
  sut.lastname = "Bar"

  wait(for: [kvoExpectation], timeout: 1)
}
```

XCTNSNotificationExpectation

```
func test_settingScore_sendsNotification() {
  expectation(
    forNotification: notificationName,
    object: nil,
    handler: nil)

  sut.score = 5

  waitForExpectations(timeout: 1, handler: nil)
}
```

XCTNSPredicateExpectation

```
func test_predicate() {
  let predicate = NSPredicate(format: "name = %@", "Foo Bar")
  let predicateExp = expectation(
  for: predicate, evaluatedWith: sut) { () -> Bool in
    return true
  }

  sut.firstname = "Foo"
  sut.lastname = "Bar"

  wait(for: [predicateExp], timeout: 1)
}
```