Reactive programming

(Used to drive multiple stuff from 1 change)

Streams are observable

If you dont subcribe they are gone for good

DisposeBag() -> for memory management gone with deinit of viewcontroller

debug() to check what’s printed as we cant print it

Oversable is just like array

Rx sends stuffs through stream at particular time so we can miss it in xctest

In RXTest the time at things start is fake , it happens in time 1 ,2 …

So this is great so our test doesnt have to be realtime

So if you testing your stream can receive data in over 20 minutes, so you dont have to wait 20 minutes for your test to run

struct CornerSorter{

let barnStream : Observable<String>

Init (tractorStream : Observable<String>){

//do something to filter corn

barnStream = tractorStream

.filter($0 == “🌽”)

}

}

**Testing //testing order at which it come through rather than time at whhich it comes through**

Import XCTest

Import RxSwift

Import RxTests

@testable import SillyDemos

class CornSorterTests : XCTestCase{

private let disposeBag = DisposeBag()

func canSortWheatFromTheChaff(){

let scheduler = TestScheduler(initialClock : 0)

let testObserver = schedular.createObserver(String)

//Given

let testInput = [“🌽”,”🐞”,”🐞”,”🐞”,”🌽”,”🐞”]

let observableInput = CornSorter(tractorStream : testInput)

let cornSorter = CornSorter(tractorStream : InputStream)

//When

cornSorter.barnStream

.subscribe(testObserver)

.addDisposableTo(disposeBag)

scheduler.start()

//Then (//time event start at 0 then next event at 1)

Let expectedEvents = [

next(1,”🌽”),

next(4,”🌽”),

completed(8)

]

XCAssertEqual(expectedEvents,testObserver.events)

}

}