RXSwift for UI

The tips and tricks your views were waiting for!

About me

Stefano Mondino

Mobile Technical Leader @ Synesthesia srl

Contacts

stefano.mondino.dev@gmail.com

stefano.mondino@synesthesia.it

github: **stefanomondino**

twitter: **@puntoste**

What's going on today

What's going on today

- RxSwift

What's going on today

- **RxSwift**

- **RxCocoa**

What's going on today

- RxSwift

- RxCocoa

- Live coding cool stuff

What's going on today

- RxSwift

- RxCocoa

- Live coding cool stuff

- Make cool stuff also reusable

RXSwift overview

- Handles streams of values over time.

- Clean syntax for FRP

- Chain, combine, manipulates events

Observable<Something>

- Wraps some asynchronous "job" that emits one or many elements

- Job starts when `subscribe()` method is manually invoked

- A job can be canceled by disposing its subscription

- `onNext:{ something in ... }` **every time job produces viable output**

- `onError:{ error in ... }` **if any error occurs in the process (only one per subscription)**

- `onComplete:{}` **if job is done**

Operators

- `.map { $0.makeMeMeaningful()}` -> **transform**

- `.filter { $0.isMeaningful() }` -> **filter out**

- `.flatMapLatest { someObservableThatDependsOn($0)}` -> **chaining**

- a **LOT** of others

RxCocoa overview

- **Available only on [i,tv,mac,watch]OS systems (no linux)**

- **Extensions related to Cocoa objects**

- **Obj-C runtime, swizzling, reflection, selectors**

- **Each `NSObject` has an associated `.rx` proxy**

.rx proxy: Howto

```
extension Reactive where Base: MyClass {
    ...
}
```

.rx proxy: Observables

```swift
extension Reactive where Base: UIScrollView {
    var xOffset: Observable<CGFloat> {
        return contentOffset.map { $0.x }
    }
}
```

.rx proxy: Binders

```swift
extension Reactive where Base: UICollectionView {
    func scrollToIndexPath(animated: Bool = true)
    -> Binder<IndexPath> {
        return Binder(base) { base, indexPath in
            base.scrollToItem(at: indexPath,
            at: .centeredHorizontally,
            animated: animated)
        }
    }
}
```

Let's play

Let's play

- Code

Let's play

- Code
- Test

Let's play

- Code
- Test
- Refactor

Thank you <3