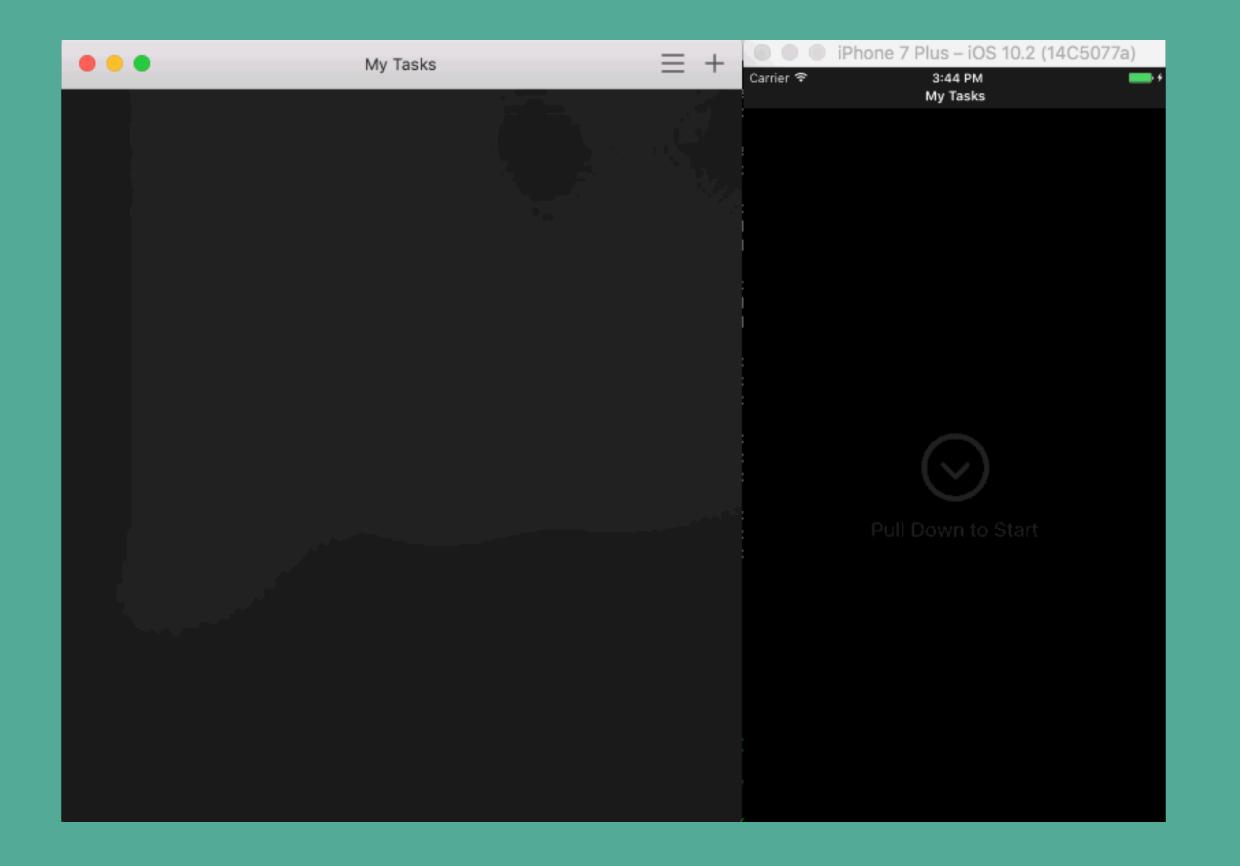
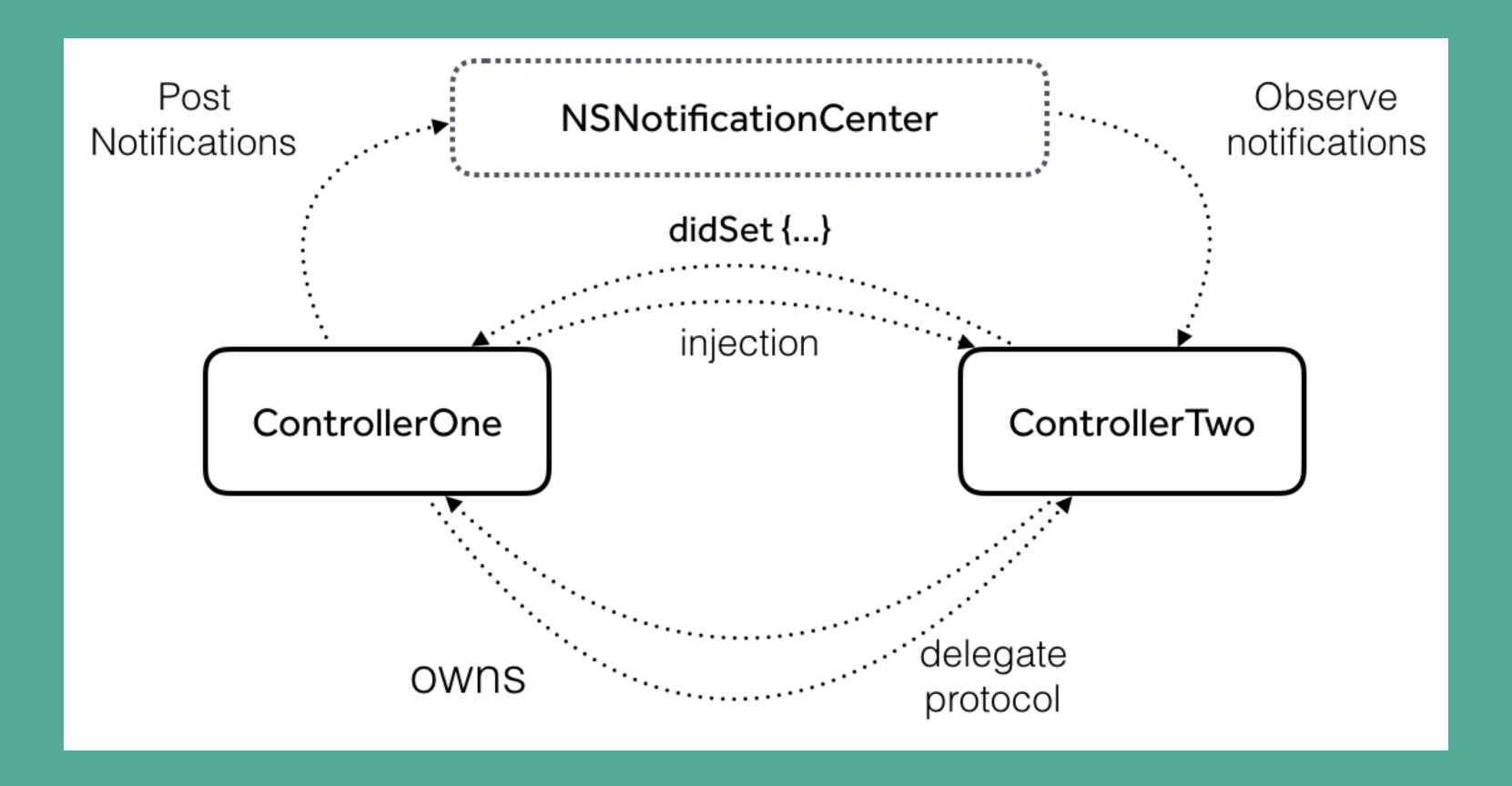
Mastering Notifications

JP Simard, @simjp, realm.io



Messaging



Reactive Programming

Beactive, Le Functiona

Sample of Cithlub Desktop Display: | Somple Text Process with pld 8692 sampled 2185 times 2185 noin (in 6164b Sesistry) = 33 (8x3896b5d57) 2185 NiApplicationMain (in AppKit) = 1176 [Bu7fffEsbufarba] 2185 -[NiApplication run] (in AppKit) = 682 [Bu7fffEsbufarba] 2185 - DKApplication _maxifuentMatchingfountMask:untilBate:inMode:dequase:] (in Applit) = 448 [Bx7fffEedfdas5] 2185 _DPSMext[vent (in AppKit) + 1876 [Bx7fffScoline7] 235 JPSERILLORI (IA AGRILI) - IBTO [BLTTTBOOLDOR] 2185 BlockintilRestiventPutchingListLiModeRithFilter (in HITsolben) - 71 [BLTTTB24297bb] 2185 ReceivaRestiventCommon (in HITsolben) - 432 (BLTTTB24257b) 2185 RestiventLiventLoopCoMode (in HITsolben) - 235 [BLTTTPS242xbb5] 2185 (Thurtophuripecific (in Corefoundation) = 196 [\$e/7ff867ech*E] 2185 _(Thurtophuri (in Corefoundation) = 1949 [\$e/7ff867ech*E] 2185 _CREALODP_IL_SIRVICING_THE_MAIN_RESPACELQUEST__ (in Corefoundation) = 9 [Bx7fff868025c9] 2185 _Gispotch_nain_ourse_collbock_ACF (in libdispotch_dylib) = 422 [Bx7fff912b06e3] 2185 _dispetch_queue_invoise (in libdispetch.dylib) = 540 [Bu7ff928f28f3] 2185 _dispetch_queue_druin (in libdispetch.dylib) = 754 [Bu7ff938f2f11] 2185 _dispetch_client_callout (in libdispetch.dylib) = 8 [Bu7ff928f3480] 2185 _dispetch_call_block_and_release (in libdispetch.dylib) = 12 [Bx7FFFSBFdScF] 2155 -[RKCscheduler performReCurrentScheduler:] (in ResctiveCocos) = 375 [Bx2BSdS62c] | 2355 -(NACPassthroughSubscriber sendMext:) (in ReactiveCocoe) = 325 [8x289cr2x89] | 2355 -(NACSubscriber sendMext:) (in ReactiveCocoe) = 119 [8x289cr4[cd]

2130 _67-(DF) indiagnositorization controller scorner did indiagnository (ASS) block invoked (in Cittab Desires) = 75 (Ba3806866a) 2118 - (G-Repository initWithfileIRL:) (in Gittle Desktop) + 534 (B-18950e7c5) 2587 - [MTUModel initWithDistinguryjerror:] (in Wortle) = 79 [8x18x4218313] | 653 -[OREspository init] (in GiVAub Desktop) = 3309 [Bi3955035c] | = 653 -[OREspository communitities[lusiveOperation]] (in GiVAub Desktop) = 153 [Bi39548831] 658 -[RACDynamicSignal subscribe:] (in ReactiveCocca) + 521 (8x189cm20bc) 650 -[RAChabscriptionScheduler scheduler] (in ReactiveCocae) = 276 [RedD0caceF] 650 _30-[RAChymanicSignal makeriber],block_invalue (in ReactiveCocae) + 50 (Be280ca225c) 658 ... 29-(RACSignal (RACStream) bind:].block_invoke (in ReactiveCoose) + 947 [Bx289cz346c 850 [McSignel(Sabsoription) subscribeRest:error:completed:] (in ResctiveCocos) = 721 [Bul89cs6790]
650 [McSignel(Sabsoription) subscribeRest:error:completed:] (in ResctiveCocos) = 721 [Bul89cs670c]
650 [McSigner(SabsoriptionChecktor schedule:] (in ResctiveCocos) = 275 [Bul89cs6047]
650 _Ne.[McSigner(SabsoriptionChecktor) = 500 [Mul89cs25c] 650 _35-CRACS(protCRACS(rean) concet(3_block_invoke (in ReactiveCacox) + 346 (0x509cr4be) 658 -[MACSignal(Subscription) subscribeNext error completed:] (in ReactiveCooms) = 723 [0=120cu6730] 658 -[MACDynomicSignal subscribe:] (in ReactiveCooms) = 521 [0=120cu6730] 650 [RACSubscriptionScheduler schedule] (in ResctiveCocoe) = 276 [Bc200cate47] 658 _30-(RACSymanicSignal subscribe:]_block_invoke (in ReactiveCocoe) = 59 [Bc200cc215c] 1 . 438 __29-(RACSignal(RACStream) bind:]_block_invoke (in ReactiveCocoo) = 947 [Bx289cx376c 438 - (RACSignal(Subscription) subscribeNext:error:completed:] (in ReactiveCoco) = 721 [9x109csE790] 438 - (DACSymanicSignal subscribe:) (in ReactiveCoco) = 521 [9x109csE790] 438 - (DACSubscriptionScheduler statedule:) (in ReactiveCoco) = 276 (Bx309csec47) 1: 438 _39-[RACDynamicSignal subscribe:]_block_invoke (in ResctiveCocos) = 59 [8x395c215c] 438 _29-(DACSignal(DACStream) bind:]_block_invoke (in ResctiveCocos) = 947 (Bx395c215d) 438 -(RACS)grain(Subscription) subscribeRext:error:completes) (in ReactiveCocou) + 721 (Bx189coE790)
438 -(RACDynamicSignal subscribes) (in ReactiveCocou) + 521 (Bx189co200c) 1 . 438 -[MACSubscriptionScheduler scheduler] (in ReactiveCocos) = 276 (Bal9Ecdock7) 438 _30-[MACSymanicSignal subscriber].block.invoke (in ReactiveCocos) = 59 (Bal9Ecal254) 438 ...29-(RACSignol(RACStream) bind:]_block_invoke (in ReactiveCocos) = 947 (Bx389cd3F6d) 438 -[MCSignal(Subscription) subscribeSextrarror.completed;] (in ReactiveCocca) = 721 [BcS9ccE706] 1 . 48 - (MATDynamicSignal subscriber) (in ReactiveScoop > SI2 (MaBBACADE)
438 - (MATDynamicSignal subscriber) (in ReactiveScoop > SI2 (MaBBACADE)
438 - (MATDynamicSignal subscriber) (in ReactiveScoop) = 276 (MatBACACE)
438 - (MATDynamicSignal subscriber).block.invoke (in ReactiveScoop) = 59 (MatBACATE)
437 - 43 (MATSLynamicSignal subscriber).block.invoke (in ReactiveScoop) = 1189 (MatBACATE)
437 - 43 (MATSLynamicSignal subscriber).block.invoke (in ReactiveScoop) = 1189 (MatBACATE) 1: | 437 -[UKCSignel(Subscription) subscribeRext:error:completed:] (in ReactiveCocee) + 721 [Bx180co679b] | 437 -[UKLDynamicSignel subscribe:] (in ReactiveCocee) + 521 [8x180co28bc] 1: 437 -[McClabscriptionSchodular schodular] (in ResctiveCocoe) = 276 [Bc180ccoc47]
437 _30-(MACDynamicSignal subscriber]_block_invoke (in ResctiveCocoe) = 59 [Bc180cc225c) 1 . 437 _30-(MAChignel(Spenetions) smitchfolatess)_block_invois (in RectiveCocoe) = 302 [8x200cbfd62] 437 -(MAChilicastConnection connect) (in RectiveCocoe) = 326 [0x200cc839f] 1 . 437 -[RACSubscriptionScheduler schedule:] (in ReactiveCocce) = \$21 (Bx189ce280x) 437 -{RACSubscriptionScheduler schedule:] (in ReactiveCocce) = 276 (Bx189cdc47) 1 . 437 _30-[DACDynamicSignal subscribe:] block invoke (in BeastiveCoos) = 50 [Bx190x215c]
437 __29-[DACSignalCHCStreen) bind:].block_invoke (in ReactiveCoos) = 947 [Bx190x215c] 437 -[BACSignal(Subscription) subscribeMest:error:completed] (in ReactiveCocoa) + 721 [BuldScaE790]
437 -(UACBymanicSignal subscribe:) (in ReactiveCocoa) + 521 (BuldScaE80c) 437 -[MACSubscriptionScheduler schedule:] (in ResctiveCocse) = 276 [Bx389cdsc47] 437 __NO-[MACSymanicSignel subscribe:]_block_invoke (in ResctiveCocse) = 59 [Bx189cs215c] 1: 437 _29-(RACSignel(RACStreen) bind:].block.invoke (in ResctiveCocos) = 947 [8x389cd976d]
437 -[RACSignel(Subscription) subscribeNext:error:completed:] (in ResctiveCocos) = 723 [8x349cd679b] 1 . 437 - (NACOynericSignal subscribe:) (in ReactiveCocoo) + 521 (Bx389ccDBc) 437 - (NACSubscriptionScheduler schedule:) (in ReactiveCocoo) - 276 (Bx389cdcc47) 437 _30-[RACDynamicSignal subscribe:]_block_invoke (in RescriveCocoo) + 59 (0x109cc215c) 437 _35-[RACSignal(Operations) tobsChtil:]_block_invoke (in RescriveCocoo) + 678 [0x109cc412d] 1 . 437 -(MCSignal(Substriction) substribe(entrerror(completed)) (in ReactiveCocos) = 721 (BuidScoRF9b)
437 -(MCDynamicSignal substribe() (in ReactiveCocos) = 521 [BuidScoRP8c] 1: 437 -[MAClabscriptineEcheller stretcher] (in ReactiveCooo) = 276 (BalBhcaceF)
437 _38-[MAClabscriptineEcheller stretcher] (in ReactiveCooo) = 276 (BalBhcaceF)
437 _38-[MAClabscriber],Block_invoke (in ReactiveCooo) = 150 (BalBhcaceF)
437 _38-[MAClabscriber],Block_invoke (in ReactiveCooo) = 246 (BalBhcaceFb)
437 -[MAClabscriptineCheller],Block_invoke41 (in ReactiveCooo) = 246 (BalBhcaceFb)
437 -[MAClabscriptineCheller],Block_invoke41 (in ReactiveCooo) = 246 (BalBhcaceFb) 437 _84-(MSDE)ect(MA(PropertySubscribing) rec.veluesAndChangesForKeyPathcoptions:observer:),block_invoke48 (in ReactiveCocoe) = 275 (BsSM2cee52) 437 -(RAChesthroughSubscriber sendRext:) (in ReactiveCocoe) = 325 (BsSM2cee52) 1: 437 -[UACSubscriber sandMext:] (in ReactiveCocoe) = 119 (0x100co4lod] 437 -[RACPosstbroughSubscriber sandMext:] (in ReactiveCocoe) = 325 (0x100cf2x00) 437 -[RACSubscriber sandkest:] (in ReactiveCocce) = 129 [BalBDceSice] 437 __25-[MACSignab(MACStreen) bind:]_bbock_invokeL25 (in ReactiveCocce) = 105 [BalBBcoMice]

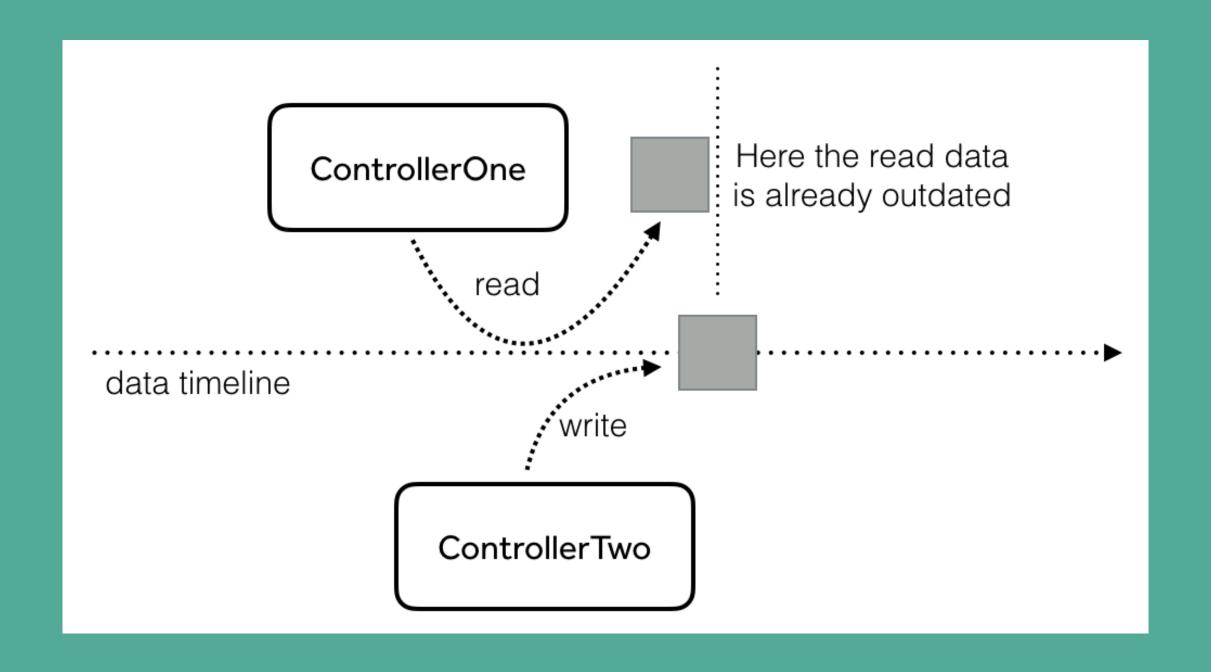


Peter Steinberger @steipete · 12 Aug 2015

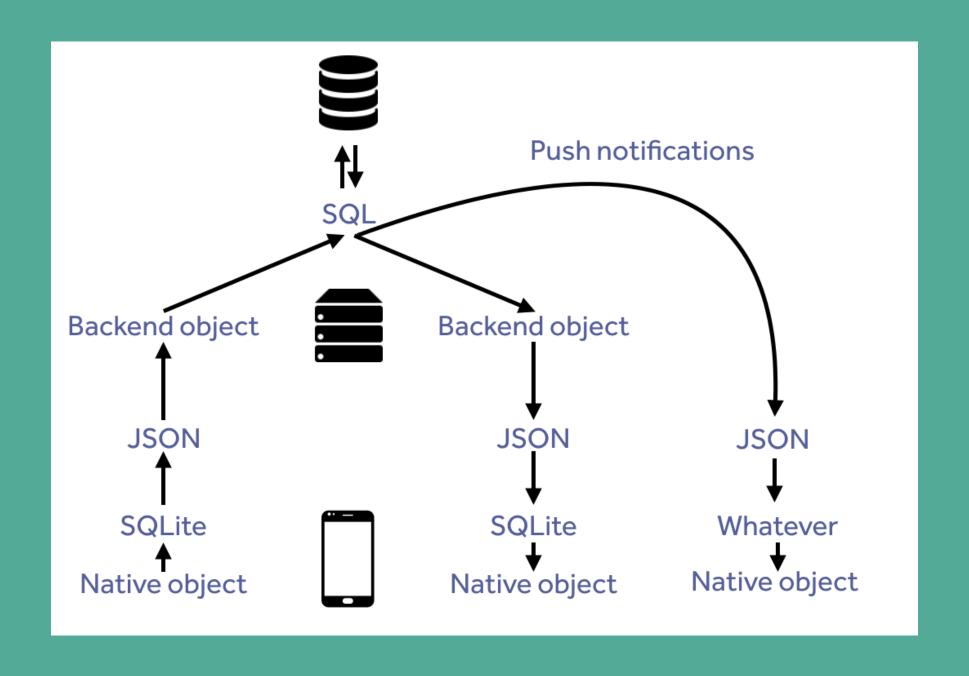
Tried GitHub Desktop. Freezes after setup complete, stack trace somewhere deeeeeeeep in ReactiveCocoa.

Reactive Programming just means reacting to changes & minimizing the explicit messaging about changes.

It's about avoiding stale data

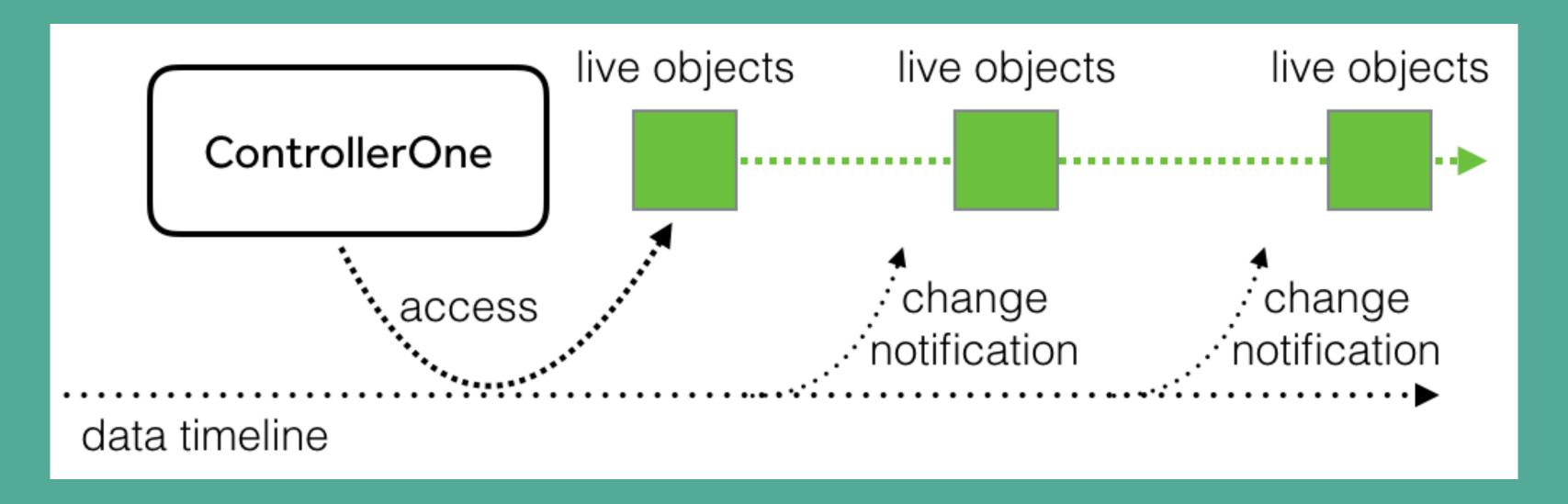


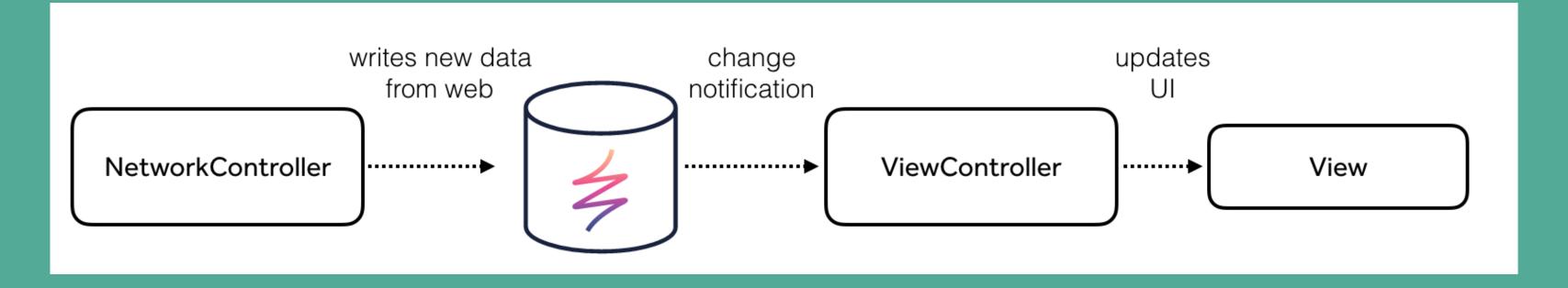
It's about avoiding duplicate code paths



Avoid different code paths

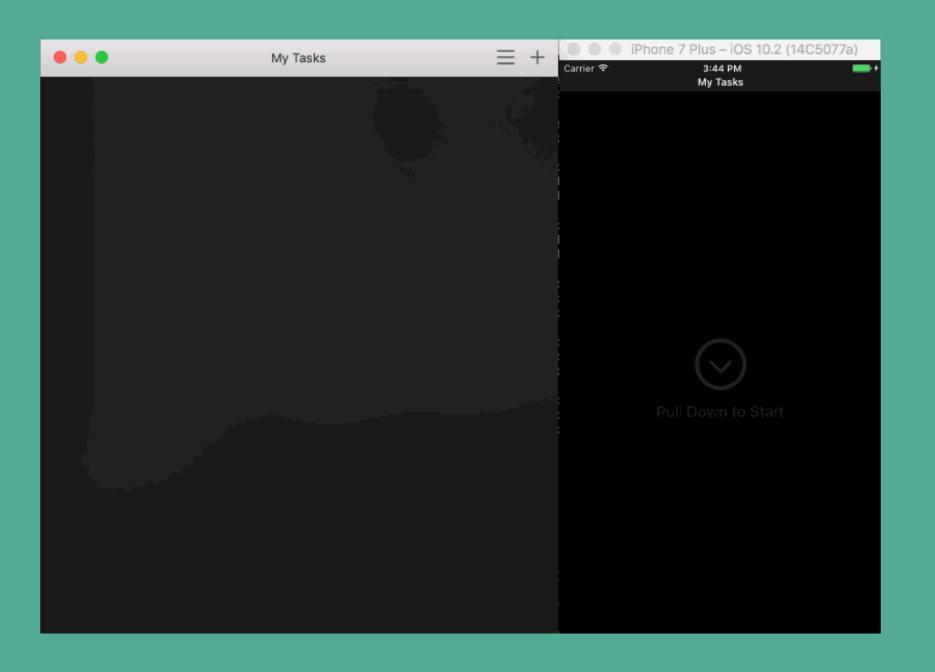
- network
- database
- user action
- push notification
- -system alerts
- other processes





Notifications

RealmTasks



Types of Realm Notifications

- Realm Notifications
- Collection Notifications
- **-KVO**
- Upcoming APIs...

Notifications

Realm Notifications

```
let realm = try Realm()
let token = realm.addNotificationBlock { notification, realm in
    viewController.updateUI()
}
// Later...
token.stop()
```

Realm Notifications

- Use when you want to react to everything that happens
- Asynchronously delivered
- Realm.refresh() waits for these notifications to be delivered
- There's usually a better way

Notification Token

- How Realm knows how long to deliver notifications for.
- Must hold a strong reference.
- Best to call stop() when you no longer want to be notified.
- Logs "released without unregistering a notification" if deallocated without calling stop().
 Harmless but the best way we know to inform

Notifications

Collection Notifications

```
let realm = try Realm()
let results = realm.objects(Item.self) // Auto-Updating Results
let token = results.addNotificationBlock(tableView.applyChanges)
```

Collection Notifications

```
extension UITableView {
 func applyChanges<T>(changes: RealmCollectionChange<T>) {
    switch changes {
      case .initial: reloadData()
      case .update(let results, let deletions, let insertions, let updates):
        let fromRow = { (row: Int) in return IndexPath(row: row, section: 0) }
        beginUpdates()
        insertRows(at: insertions.map(fromRow), with: .automatic)
        reloadRows(at: updates.map(fromRow), with: .automatic)
        deleteRows(at: deletions.map(fromRow), with: .automatic)
        endUpdates()
      case .error(let error): fatalError("\(error)")
```

Notifications may be coalesced! (grouped)

Notifications may be coalesced (grouped)

```
switch changes {
  case .initial: reloadData()
  case .update(let results, let deletions, let insertions, let updates):
    let fromRow = { (row: Int) in return IndexPath(row: row, section: 0) }
    beginUpdates()
    insertRows(at: insertions.map(fromRow), with: .automatic)
    reloadRows(at: updates.map(fromRow), with: .automatic)
    deleteRows(at: deletions.map(fromRow), with: .automatic)
    endUpdates()
  case .error(let error): fatalError("\(error)")
```

Synchronous vs Asynchronous Queries

Results are queried synchronously if requested, asynchronously otherwise.

Asynchronous Query

```
let realm = try Realm()
// Query never performed on the current thread
let results = realm.objects(Item.self)
let token = results.addNotificationBlock { _ in
   /* results available asynchronously here */
}
```

Synchronous Query

```
let realm = try Realm()
// Query performed synchronously to return first result
_ = results.first // <- query performed here
let token = results.addNotificationBlock { _ in
    /* results available asynchronously here */
}</pre>
```

Collection Notifications

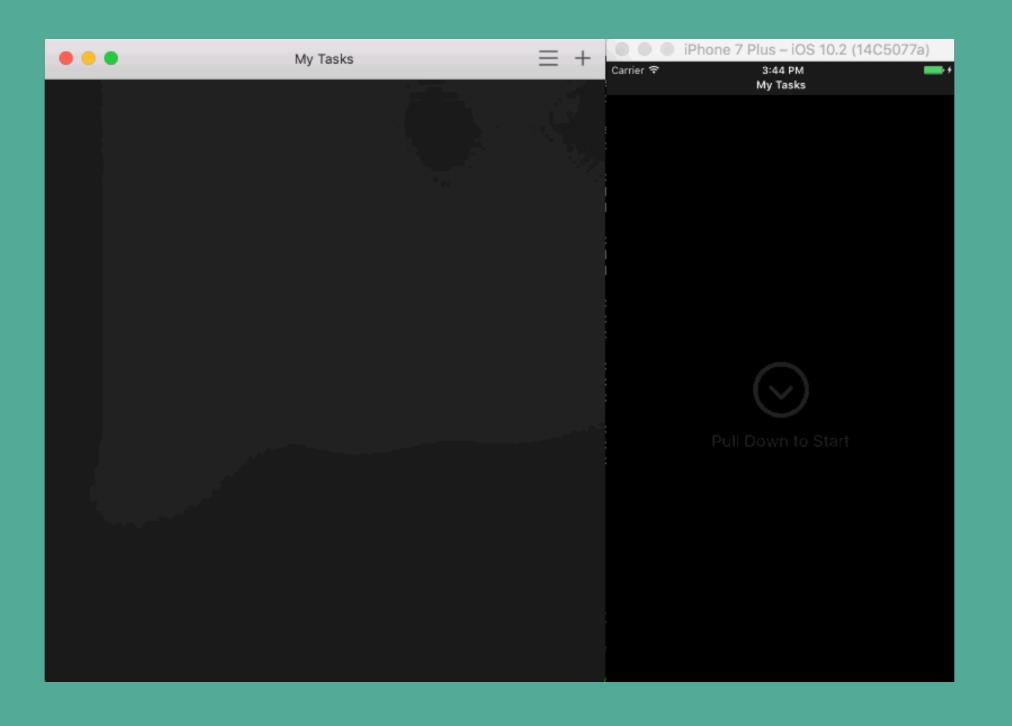
- May be coalesced
- Generally delivered on runloop iterations
- Query upkeep always performed on background thread
- Notifications delivered on the source thread
- Can avoid all source thread overhead if used correctly

nterface-Driven Mittes

Interface-Driven Writes

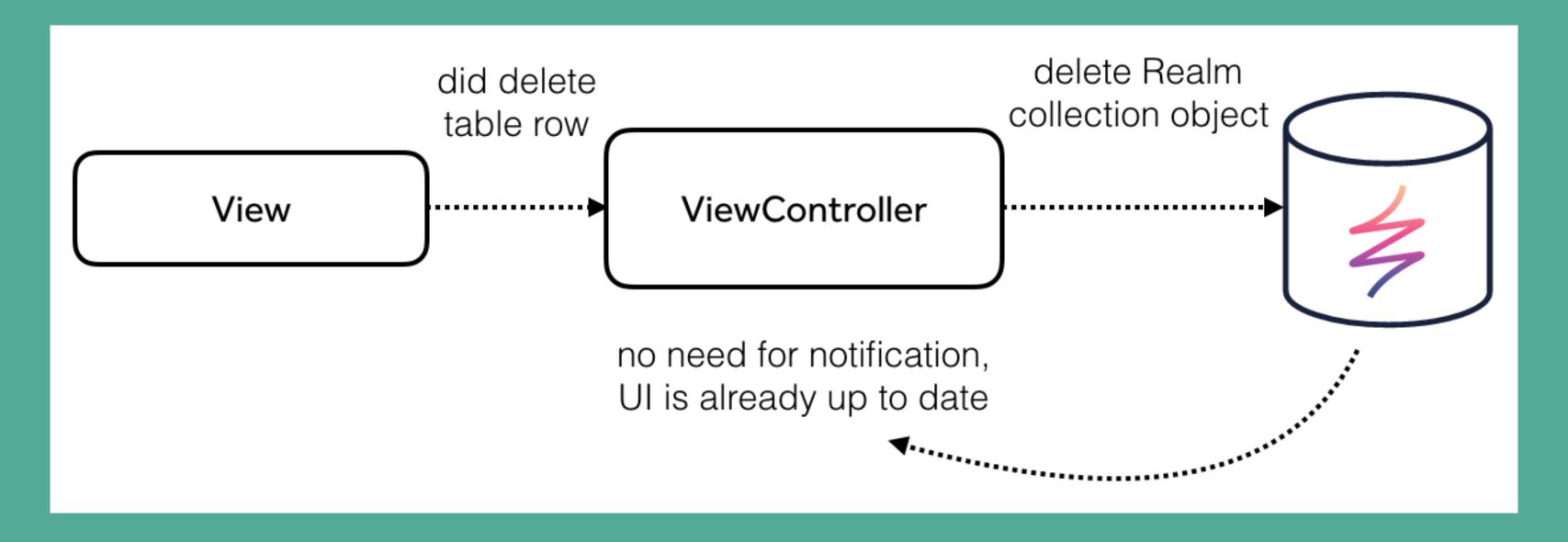
- Changeset notifications are all about applying state diffs.
- State must be up to date for the diff to make sense.
- Some changes should be applied to the UI synchronously.

Like reordering rows.



Interface-Driven Writes

```
func viewDidLoad() {
 messages = realm.objects(Message.self)
 self.token = messages.addNotificationBlock(tableView.applyChanges)
func tableView(_ tableView: UITableView,
               commit editingStyle: UITableViewCellEditingStyle,
               forRowAt indexPath: IndexPath) {
  guard editingStyle == .delete else { return }
  realm.beginWrite()
 messages.removeAtIndex(indexPath.row)
  tableView.deleteRows(at: [indexPath], with: .automatic)
  realm.commitWrite(withoutNotifying: [self.token])
```



Key-Value Observation

Key-Value Observation

- Enables Realm to work well with Cocoa APIs
- ReactiveCocoa
- Synchronously delivered
- Single property at a time

Obects as Messages

Inter-process data sharing/message passing

- Realm already has robust interprocess data sharing
- Could use to share data or notify main iOS app from app extension

Not limited to single-device

- Also for Realm Object Server communication
- And massively distributed apps

Objects as Messages

Objects as Messages

```
let collectionOfOne = managementRealm.objects(SyncPermissionChange.self)
                                     .filter("id = %@", permissionChange.id)
token = collectionOfOne.addNotificationBlock { notification in
 if case .update(let changes, _, _, _) = notification,
   let change = changes.first {
    // Object Server processed the permission change operation
   switch change.status {
   case .notProcessed: break // handle case
   case .success: break // handle case
   case .error: break // handle case
   print(change.statusMessage) // contains error or message
```

```
let realm = try Realm()
let jane = realm.objects(Person.self).filter("name == 'Jane'").first!
let token = jane.addNotificationBlock { change in
  switch change {
  case .change(let propertyChanges):
    for propChange in propertyChanges {
      print("'\(propChange.name)': \(propChange.oldValue) -> \(propChange.newValue)")
  case .deleted: print("object was deleted")
  case .error(let error): print("notification couldn't be delivered: \(error)")
// Later...
token.stop()
```

- Coming early 2017
- Asynchronously delivered
- Great for Objects-as-APIs/Messages and event handling

Resources

- Realm's docs on Notifications
 - realm.io/docs/swift#notifications
- Live Objects and Fine-Grained Notifications
 - realm.io/news
- RealmTasks
 - github.com/realm/RealmTasks
 - Removing workaround PR: RealmTasks#352
- RxRealm: RxSwiftCommunity/RxRealm

Questions?

JP Simard, @simjp, realm.io