WATCH & WatchKit

A Talk for CocoaHeads DC on April 2, 2015

Arthur Ariel Sabintsev

@ArtSabintsev

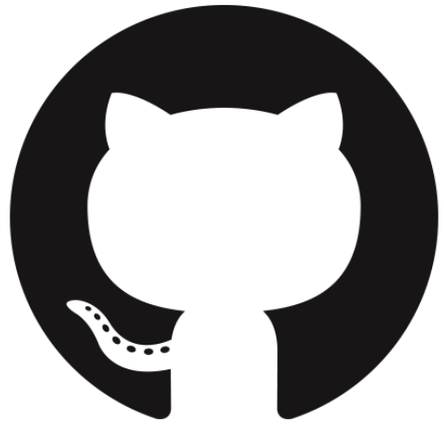
AGENDA

- **Introduction**
- **WATCH**
  - Everything you need to know
- **WatchKit**
  - Framework, UI Design Basics, UI Elements
- **Demo**

Introduction

Teach

Build



Tinker

@ArtSabintsev

WATCH

@ArtSabintsev

Disclaimer

- I have never played with an WATCH
- How I prepared:
  - I read a book (WatchKit by Tutorials)
  - I watched a set of video tutorials (WatchKit Series)
  - I attended a conference (RWDevCon)
  - I read Apple's Watch Programming Guide
  - I read Apple's Watch Human Interface Guidelines

WATCH - Models

WATCH

Stainless steel or space black stainless steel cases. Sapphire crystal. A range of stylish bands.

Learn more ›

WATCH SPORT

Anodized aluminum cases in silver or space gray. Strengthened Ion-X glass. Colorful, durable bands.

Learn more ›

WATCH EDITION

18-karat gold cases in yellow or rose. Sapphire crystal. Exquisitely crafted bands and closures.

Learn more ›

@ArtSabintsev

WATCH - Sizes

38mm (height)

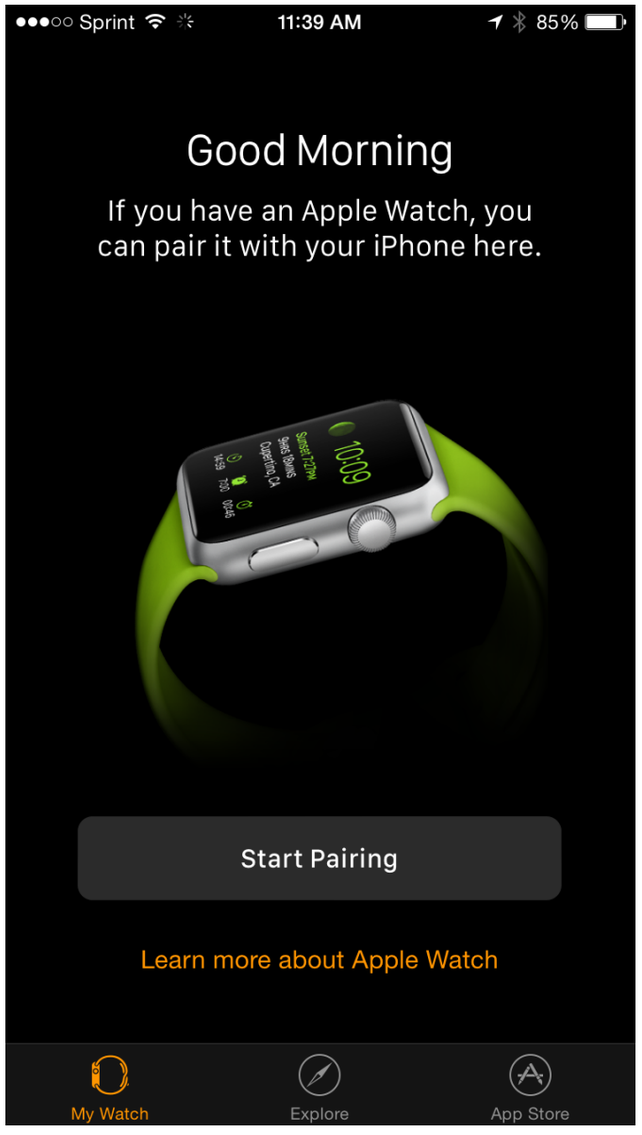
42mm (height)

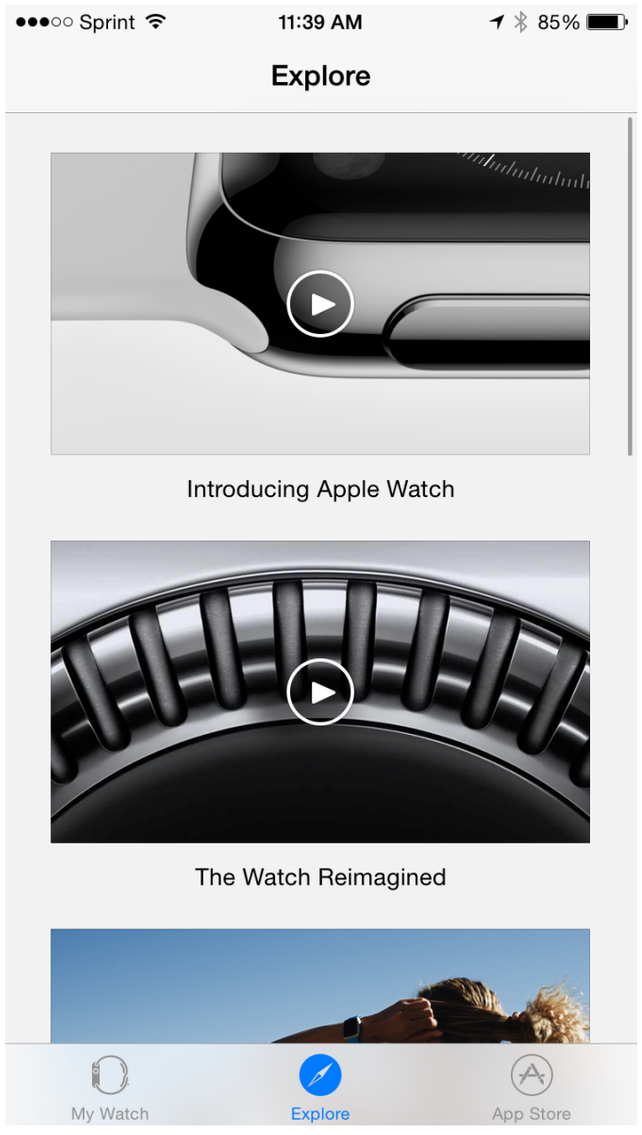
@ArtSabintsev

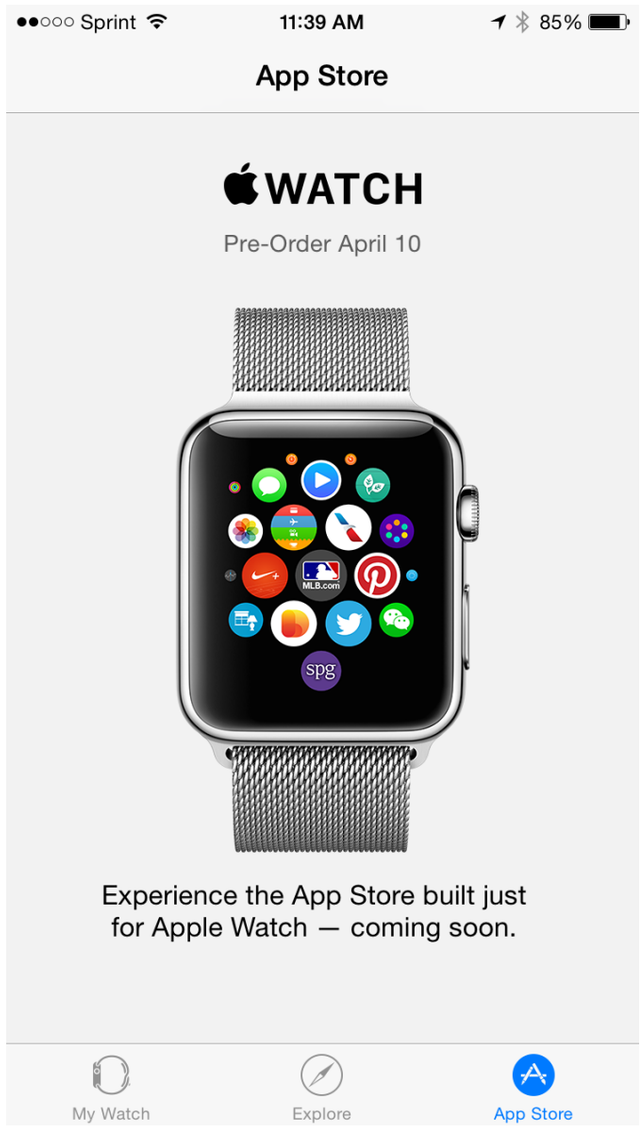
WATCH - Materials

@ArtSabintsev

WATCH - App Store





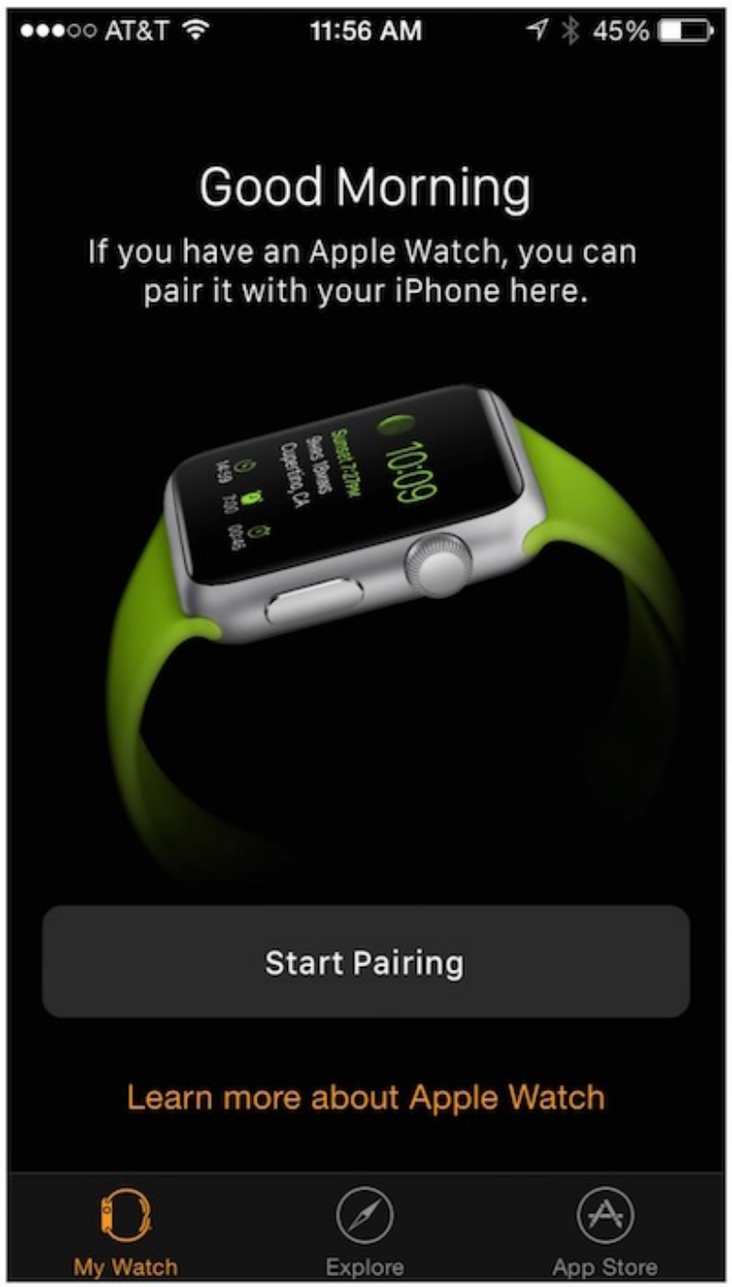


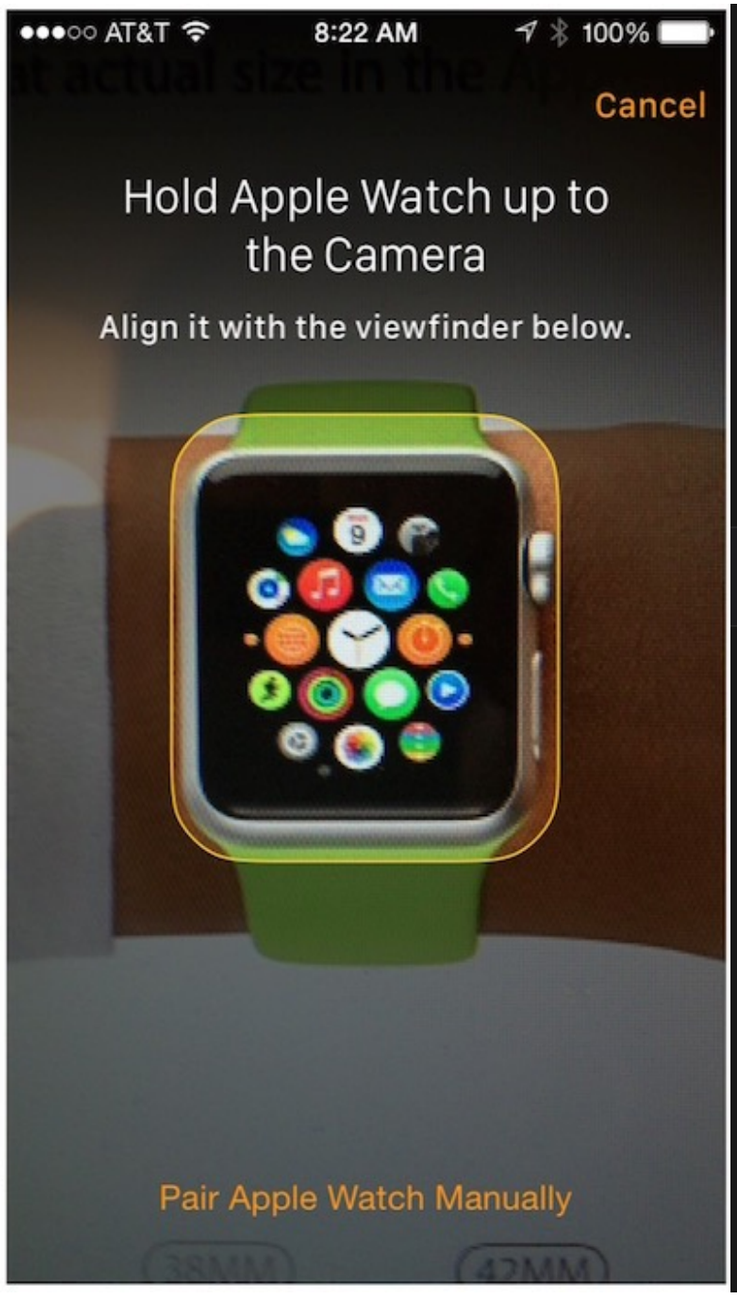
@ArtSabintsev

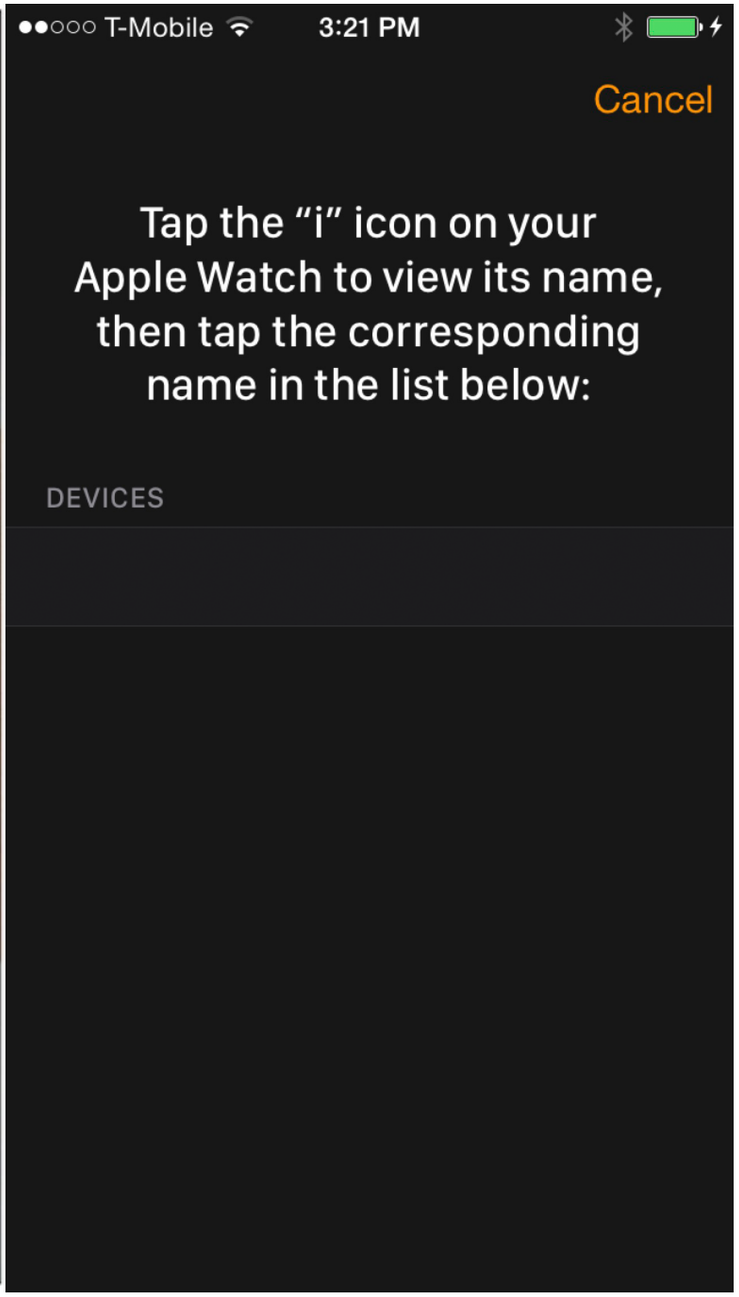
WATCH - Features (Pt. 1)

- Runs on iOS 8+
- Compatible with iPhone 5, 5c, 6s, 6, 6+
- Pairs to iPhone via Bluetooth
- If out of range, can still be connected to iPhone via Wi-Fi, if both are on the same network.
  - Third-Party apps can only be used if Apple Watch is in range of iPhone (Bluetooth or Wi-Fi)

WATCH - Features (Pt. 2)







@ArtSabintsev

WATCH - Features (Pt. 3)

- 24 hr Battery Life
- Sweat Resistant
- Sapphire Crystal (Scratch Resistant)
- Siri Integration (Hands-free interaction)
- Apple Pay Integration
- Health & Fitness Integration
- Phone, SMS, Push Notifications, etc.

WATCH - Features (Pt. 4)

The Crown

- Scroll lists
- Zoom in and out of photos
- Press to return home

@ArtSabintsev

WATCH - Features (Pt. 5)

Stay closer to the people you care about.

Now your inner circle is always nearby. Press the side button to access Friends, where you'll see thumbnails of the people you like to stay in touch with most. Tap to send a message, make a call, or reach out in one of the new ways only Apple Watch makes possible.

- The 'Button'
  - Tap to access Friends app
- Friends App
  - Contact List
  - Tap to call
  - Tap to text
  - Tap to *Digital Touch*

@ArtSabintsev

WATCH - Features (Pt. 6)

Digital Touch

Tap.

Let friends or loved ones know you're thinking of them with silent, gentle tap patterns they'll feel on the wrist. You can even customize taps for different people.

@ArtSabintsev

WATCH - Dates

Pre-Order	Release
April 10, 2015	April 24, 2015

@ArtSabintsev

WATCH - Submit Apps

Developer

Submit your WatchKit apps now.

It's time. Apple Watch will be in the hands of customers on April 24. Get your WatchKit apps ready and submit them for review now.

Once your WatchKit app is approved and released by Apple, your existing iPhone users will receive the app update and customers will see your WatchKit extension icon and description on the App Store. A small group of people who currently have an Apple Watch will be able to use your WatchKit app before April 24, so make sure your back end systems are ready. If you prefer your WatchKit app to be available only after Apple Watch is available to all customers, set your release date to "manual" and update it on April 24.

Read Preparing Your App Submission for Apple Watch to learn more.

@ArtSabintsev

WATCH - Am I getting one?

Yes, ***but*** I don't want it. I need it for my career.
I feel like the **WATCH *literally*** shackles me to technology

@ArtSabintsev

WatchKit

@ArtSabintsev

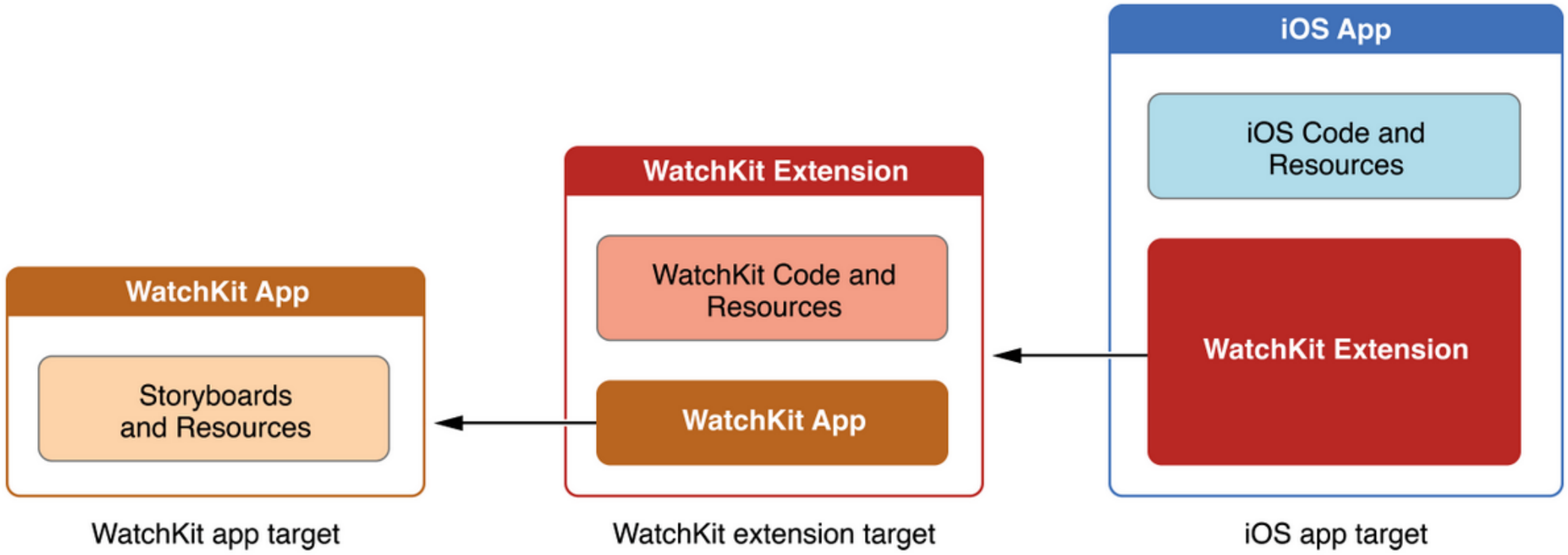
WatchKit - App Architecture (Pt. 1)

- **How do I make and ship an WATCH App?**
  - iPhone App + WatchKit Extension + WatchKit App

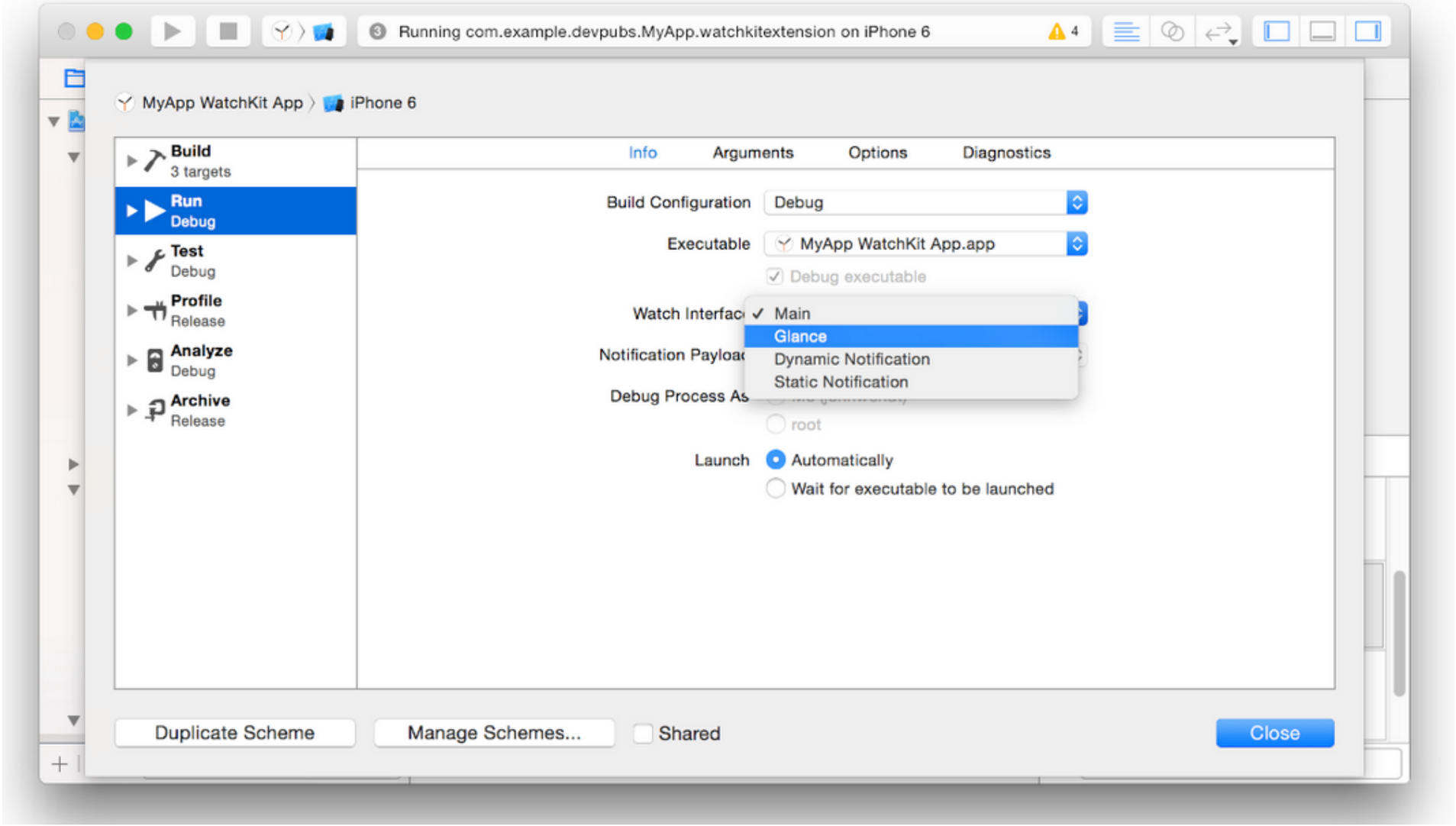
- **iPhone App**
  - Your standard iPhone app that you would like make accessible on the **WATCH**
- **WatchKit Extension**
  - Code to manage WatchKit App UI and interactions
- **WatchKit App**
  - The UI: Storyboards and Resource files

WatchKit - App Architecture (Pt. 2)

Figure 2-1 The target structure for a WatchKit app



WatchKit - Interface Types (Pt. 1)

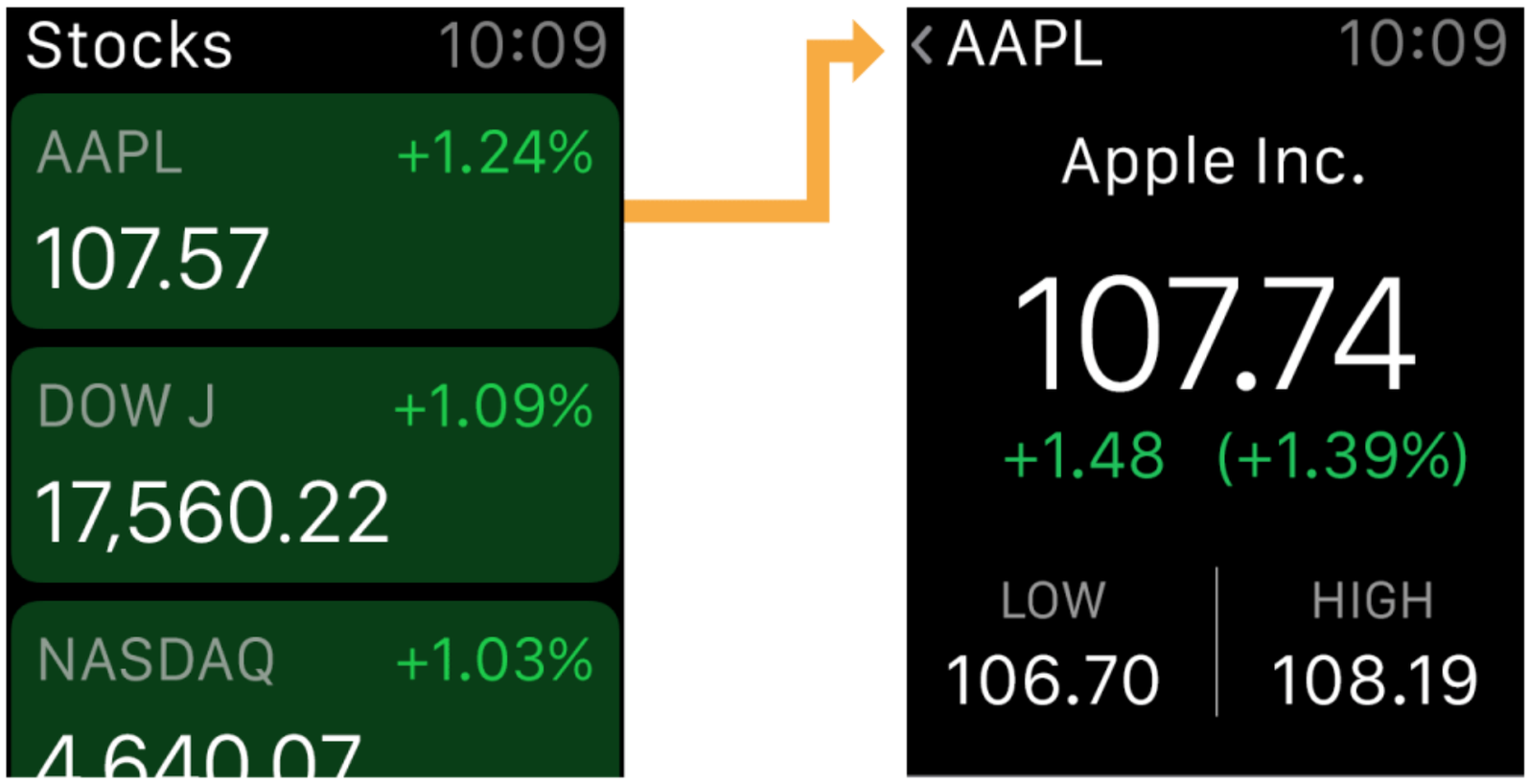


@ArtSabintsev

WatchKit - Interface Types (Pt. 2)

Main (Default) Interface

- **Two types of Apps**
  - Hierarchical
  - Page-based
- **Responds to**
  - Gestures
  - Force Touch
  - Digital Crown



WatchKit - Interface Types (Pt. 3)

Main (Default) Interface with Force-Touch Context Menu

- Force touch the default interface WatchKit app
- Displays up to four actions
- Actions are view/interface-specific
  - Ex. Displaying time in multiple formats
  - Ex. Displaying options on a Map Screen, like Satellite, Vector, or Hybrid view

WatchKit - Interface Types (Pt. 4)

Glance Interface

- Template based
- Not scrollable
- One Action
  - Tapping anywhere opens up your WatchKit app

WatchKit - Interface Types (Pt. 5)

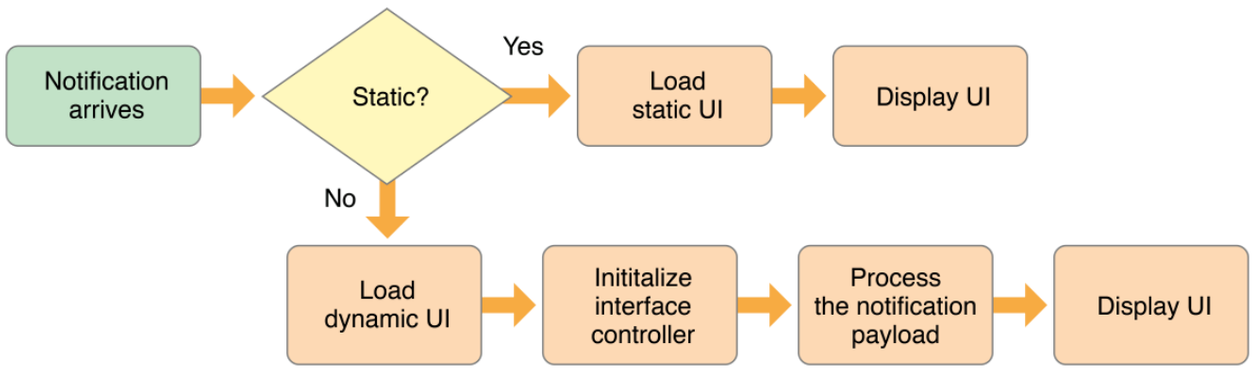
Short Look Notification Interface

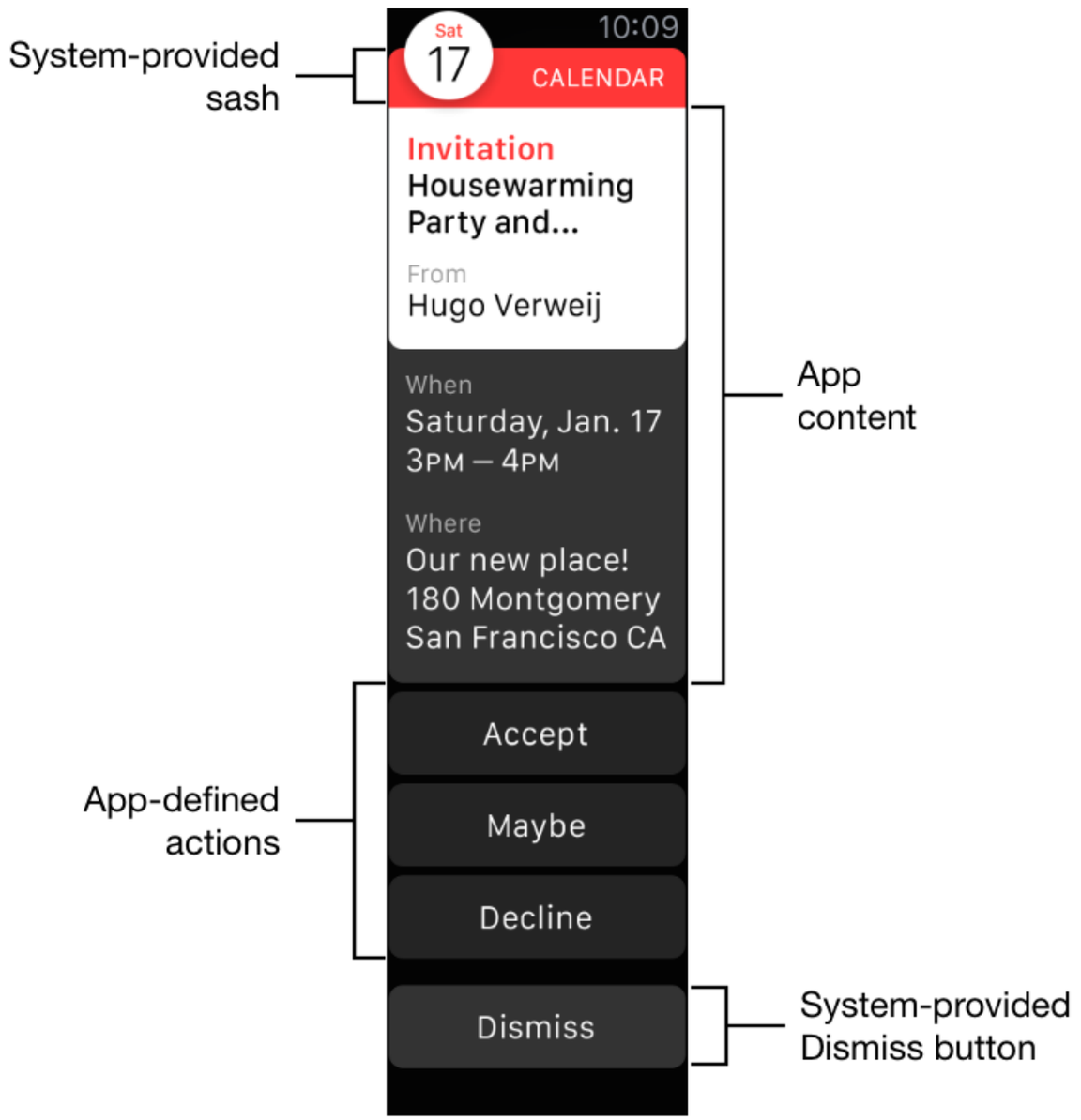
- Template based
- Not scrollable
- One Action
  - Tapping anywhere opens up your Watc app
- Customize Title

WatchKit - Interface Types (Pt. 6)

Long Look Notification Interface

- Template based
- Scrollable
- Up to four actions
- Customizable
- Can be *Static* or *Dynamic*





@ArtSabintsev

WatchKit - Framework

Classes

NSObject	`NSObject` is the root class of most Objective-C class hierarchies.
WKAccessibilityImageRegion	A `WKAccessibilityImageRegion` object defines a portion of an image that you want to call out separately to an assistive app.
WKInterfaceController	The `WKInterfaceController` class is the main class for implementing your WatchKit app's interface.
WKUserNotificationInterfaceController	The `WKUserNotificationInterfaceController` class manages a custom user interface for a local or remote notification.
WKInterfaceDevice	A `WKInterfaceDevice` object encapsulates information about the user-specific configuration of the paired Apple Watch.
WKInterfaceObject	The `WKInterfaceObject` class is the base class for all interface objects in your app.
WKInterfaceButton	A `WKInterfaceButton` object represents a tappable area on the screen.
WKInterfaceDate	A `WKInterfaceDate` object is a custom label that displays the current date or time.
WKInterfaceGroup	A `WKInterfaceGroup` object is a container for one or more interface objects.
WKInterfaceImage	A `WKInterfaceImage` object lets you manipulate an image in your WatchKit app's interface.
WKInterfaceLabel	A `WKInterfaceLabel` object lets you manipulate the contents of an onscreen label at runtime.
WKInterfaceMap	A `WKInterfaceMap` object displays a noninteractive map for the location you specify.
WKInterfaceSeparator	A `WKInterfaceSeparator` object lets you manipulate a separator at runtime.
WKInterfaceSlider	A `WKInterfaceSlider` object represents a visual control used to select a single floating-point value from a range of values.
WKInterfaceSwitch	A `WKInterfaceSwitch` object represents a control that toggles between an On and Off state.
WKInterfaceTable	A `WKInterfaceTable` object creates and manages the contents of a single-column table interface.
WKInterfaceTimer	A `WKInterfaceTimer` object is a special type of label that displays a countdown timer.

@ArtSabintsev

WatchKit - Classes (Pt. 1)

- As we're short on time, I'll only be covering WKInterfaceController and the WKInterfaceObject subclasses

- WKInterfaceController
  - Analog to UIViewController in UIKit
- WKInterfaceObject
  - Analog to UIView in UIKit

WatchKit - Classes (Pt. 2)

- **WKInterfaceController Methods**
  - *Initialization*
    - init
    - awakeWithContext(_:)
  - *Interface*
    - willActivate() analog to viewWillAppear(_:)
    - willDeactivate() analog to viewWillDisappear(_:)
  - *Navigation*
    - pushControllerWithName(_:context:)
    - popController()
    - popToRootViewController()
  - *Modal*
    - presentControllerWithName(_:context:)
    - dismissController()

WatchKit - Classes (Pt. 3)

- Contexts in **WKInterfaceController**
  - Contexts are defined as **AnyObject?**
  - Used to pass data you want loaded on the screen before willActivate() is called

WatchKit - Classes (Pt. 4)

- **WKInterfaceObject**, specifically **WKInterfaceGroup**
  - It's a UI object that is a container.
  - Defines bounds for layout of all WKInterfaceObject's that it contains within its boundaries
  - Groups can be nested

WatchKit - Classes (Pt. 5)

- **What Exists?**
  - WKInterfaceLabel (analog to UILabel)
  - WKInterfaceImage (analog to UIImage and UIImageView)
  - WKInterfaceTable (analog to UITableView and its delegates)
  - WKInterfaceButton (analog to UIButton)
  - WKInterfaceSwitch (analog to UISwitch)
  - WKInterfaceSlider (analog to UISlider)
  - WKInterfaceMap (analog to MKMapView)
  - WKInterfaceDate (no analog)
    - Pretend that UILabel, NSDate, and NSDateFormatter had a View-Controller paradigm-breaking baby.

WatchKit - Design Guidelines (Pt. 1)

38mm

42mm

50px larger height

40px larger width

@ArtSabintsev

WatchKit - Design Guidelines (Pt. 2)

- WatchKit Apps automatically grow vertically when they run out of space to present content
- Presented WKInterfaceControllers come with back buttons
- Modal InterfaceControllers come with close buttons
- WKInterfaceLabel fonts are limited to system fonts and pre-determined sizes
- Watch Apps should mostly be used in a *read-only* or *tell-me-more-information* state
  - That's why objects like UITextField, UITextView, UISearchBar, and UIKeyboard don't have WatchKit analogs

Demo

(if you want one)