

Independent watchOS:

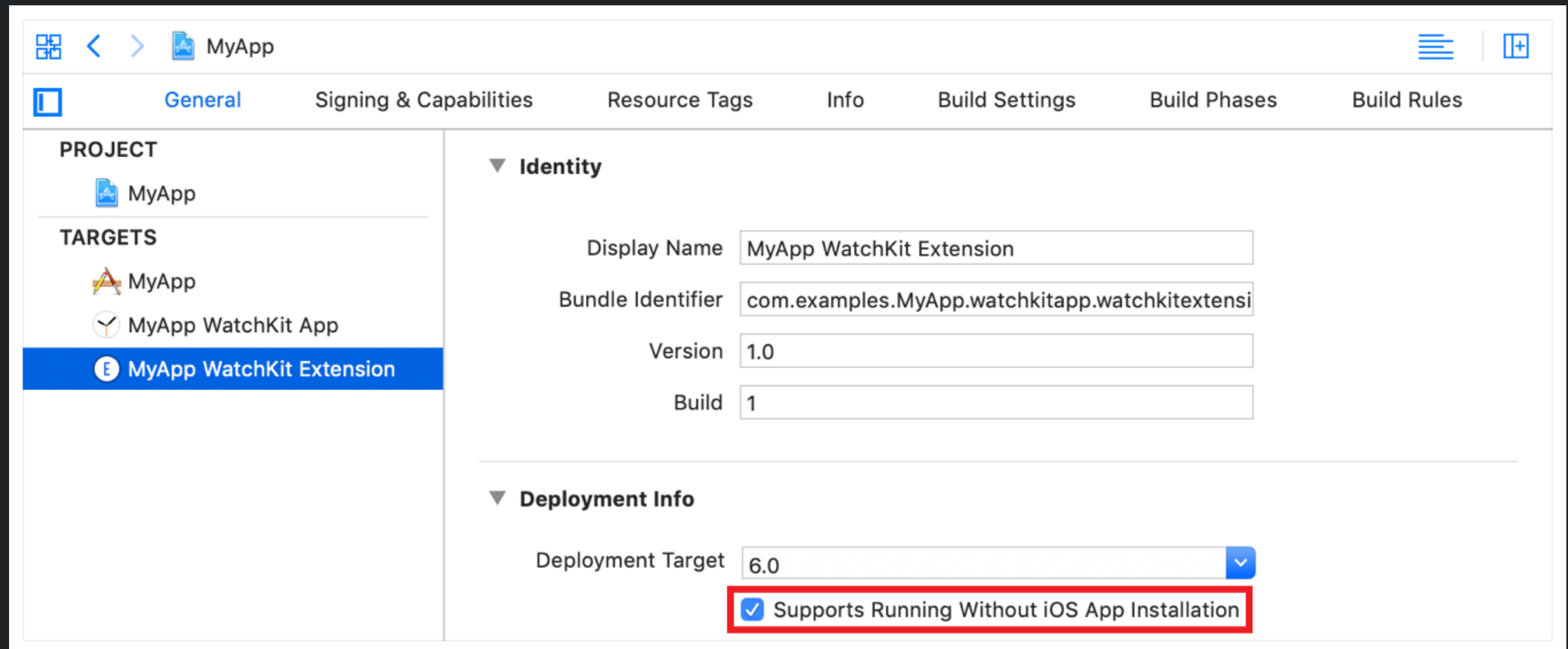
- principles**
- practices**

Vadim Drobinin | @valzevul

Independent apps

**Independent
apps. Just put
a checkmark?**

Independent apps



questions?

drobinin.com | @valzevul

**Independent
apps. Just put
~~a cheekmark?~~
Think different.**

Agenda

Agenda

— Back to the roots

Agenda

- Back to the roots
- watchOS 1..<6

Agenda

- Back to the roots
- watchOS 1.. <6
- watchOS 6

Agenda

- Back to the roots
- watchOS 1.. <6
- watchOS 6
- Principles

Agenda

- Back to the roots
- watchOS 1.. <6
- watchOS 6
- Principles
- Best practices

Agenda

- Back to the roots
- watchOS 1..<6
- watchOS 6
- Principles
- Best practices
- Watch App Store

Agenda

- Back to the roots
- watchOS 1.. <6
- watchOS 6
- Principles
- Best practices
- Watch App Store
- Summary

Special Offer! TRS-80 Pocket Computer

Save
29⁹⁵

at the Cassette
Store (26-3503) at
a Charge When
you buy a TRS-80 PC-1

49⁹⁵

Gift for Students and Engineers
in Easy-to-Learn Pocket BASIC or
Ready-to-Run Cassette Software

Pocket Computer Model PC-1. It's an on-the-go
calculator's best friend—ideal for accountants,
engineers, and engineering and science students. With the
cassette interface that you get as a special bonus and a
cassette recorder, you can use our ready-to-run software. Add
to the sale-priced programs shown at right and you can
solve almost any technical or engineering problem, analyze statistics,
determine your financial status, handle real estate—or just
develop your own applications with the Pocket
Computer. PC-1 has a 1.9K memory, 10-digit
display with 2-digit exponents, 15 arithmetic functions,
LCD readout, typewriter-style keyboard with
function keys, multiple statements, strings, arrays, and
programs and data are retained when power is off. It's
small and weighs less than 6 oz. Includes carry case and
cassette recorder. Reg. \$149.95. 26-3501 149.95
Deluxe Carrying Case. Protects your PC-1 and
cassette recorder. Reg. \$14.95. 26-3508 Sale 7.49



Sale! Save Up to 50% on PC-1 Software

Personal Finance. Budget management, electronic checkbook,
annuities, metric/English conversion. Reg. \$19.95. 26-3518
Business Finance. Loans, annuities, bond prices, yields, compound
depreciation, and more. Reg. \$19.95. 26-3517
Business Statistics. Seasonal variation, norm T/F distribution,
multiple regression and more. Reg. \$19.95. 26-3516
Real Estate. Annuities, compound interest, depreciation, cash flow
residential purchase analysis. Reg. \$24.95. 26-3510
Surveying. Leveling, volume, irregular area, closure and more. Reg.
26-3512
Aviation. Altitude, airspeed, headings and more. Reg.
26-3513
Electrical Engineering I. Filter, transistor and RF coil design
analysis, more. Reg. \$24.95. 26-3520
Civil Engineering. Simple beam computations, column buckling
formulas, stress in cylinders, bolt torque. Reg.
26-3511
Games I. Fun for the whole family. Reg. \$14.95. 26-3515
Games II. More family fun. Reg. \$14.95. 26-3523
Math Drill. Teaches basic skills. Reg. \$14.95. 26-3514
Programs require cassette recorder, plus cassette interface or printer/plotter.

**Our Most Powerful
Pocket Computer**
279⁹⁵

- Fast Extended Pocket BASIC
- Upper/Lower Case Display
- Fully-Addressable 7 x 15 Dot Matrix Graphics
- 2640-Character User Memory For Programs and Data Storage

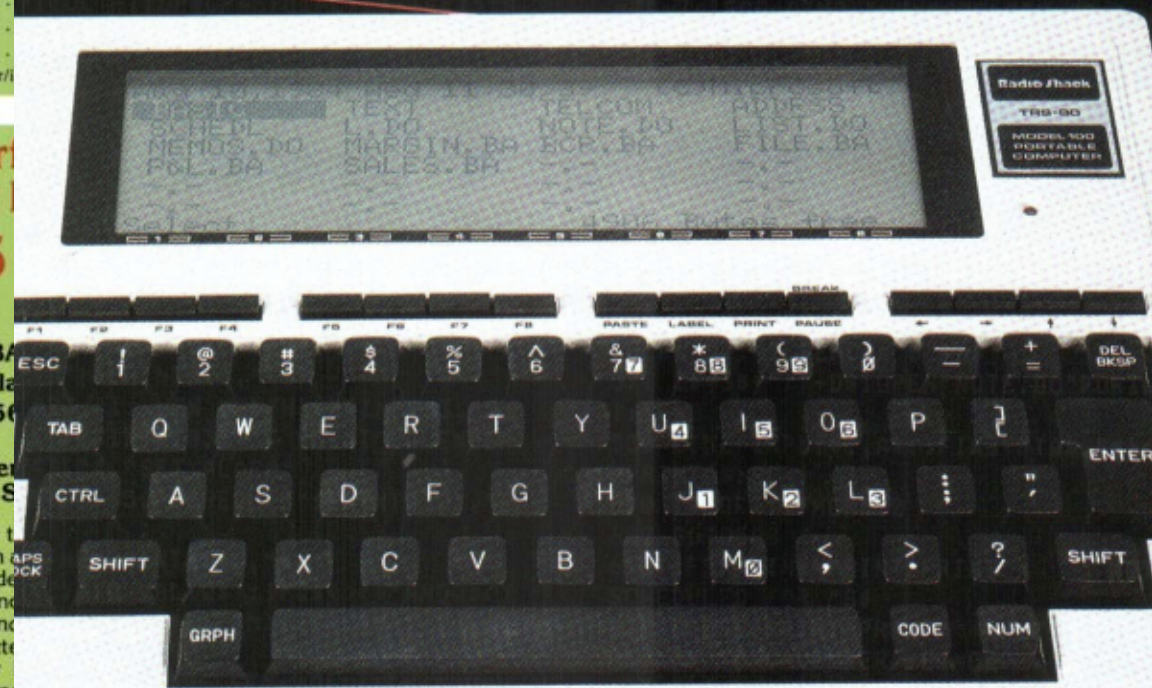
TRS-80 Pocket Computer PC-2. Get the
power and speed of many desktop computers in a
package you can hold in your hand. It's ideal for the
professional. Extended Pocket BASIC includes
statements, 36 functions and 6 command
dimension arrays, variable length character
program merge and more. Typewriter-style
with datapad includes 18 programmable
functions and 18 user-definable keys. For
expand with a 4K or 8K RAM plug-in module.
Quartz real-time clock. All memory information
is retained when power is off. Rechargeable
batteries. 26-3501



Printer/Plotter Interface. For larger, more complex
programs. Individual modules
are not expandable.
4K. 26-3506 199.95
8K. 26-3507 299.95
PC-2 Memory Modules.
For larger, more complex
programs. Individual modules
are not expandable.
4K. 26-3506 199.95
8K. 26-3507 299.95
PC-2 Printer Carrying Case. Holds printer (with
PC-2 installed) and AC
adapter/charger. Keeps
organized. 26-3508 29.95

Radio Shack
has redefined the concept
of personal computing with its
TRS-80® Model 100 Portable Computer

—Isaac Asimov
Renowned Science and
Science Fiction Author



ELECTRONICS
Australia
AUST \$1.60* NZ \$1.70
SEPTEMBER, 1980



Acoustically
coupled modem

TRS-80 pocket
computer!

in tremolo
organs

signing vented
maker enclosures

Who are the
computer criminals?

**FREE TANG
CATALOG**

When It Comes to Computers—Come to Your Nearby Radio Shack Computer Center

<https://paleotronic.com/2019/09/28/from-pocket-computers-to-palmtops-an-early-history-of-mobile-telematics/>

Back to the roots

Back to the roots

— 13th century: eyeglasses

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock
- 1975: calculator wristwatch

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock
- 1975: calculator wristwatch
- 1984: The Organiser

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock
- 1975: calculator wristwatch
- 1984: The Organiser
- 2002: bluetooth headset

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock
- 1975: calculator wristwatch
- 1984: The Organiser
- 2002: bluetooth headset
- 2006-2013: Nike+, Fitbit, Google Glass

Back to the roots

- 13th century: eyeglasses
- 16th century: wearable clocks
- 1960s: "Beat the Dealer" clock
- 1975: calculator wristwatch
- 1984: The Organiser
- 2002: bluetooth headset
- 2006-2013: Nike+, Fitbit, Google Glass
- 2014: Apple Watch

Back to the roots



Back to the roots

— PDAs





Back to the roots

- PDAs
- Smartphones



Back to the roots

- PDAs
- Smartphones
- Wearables

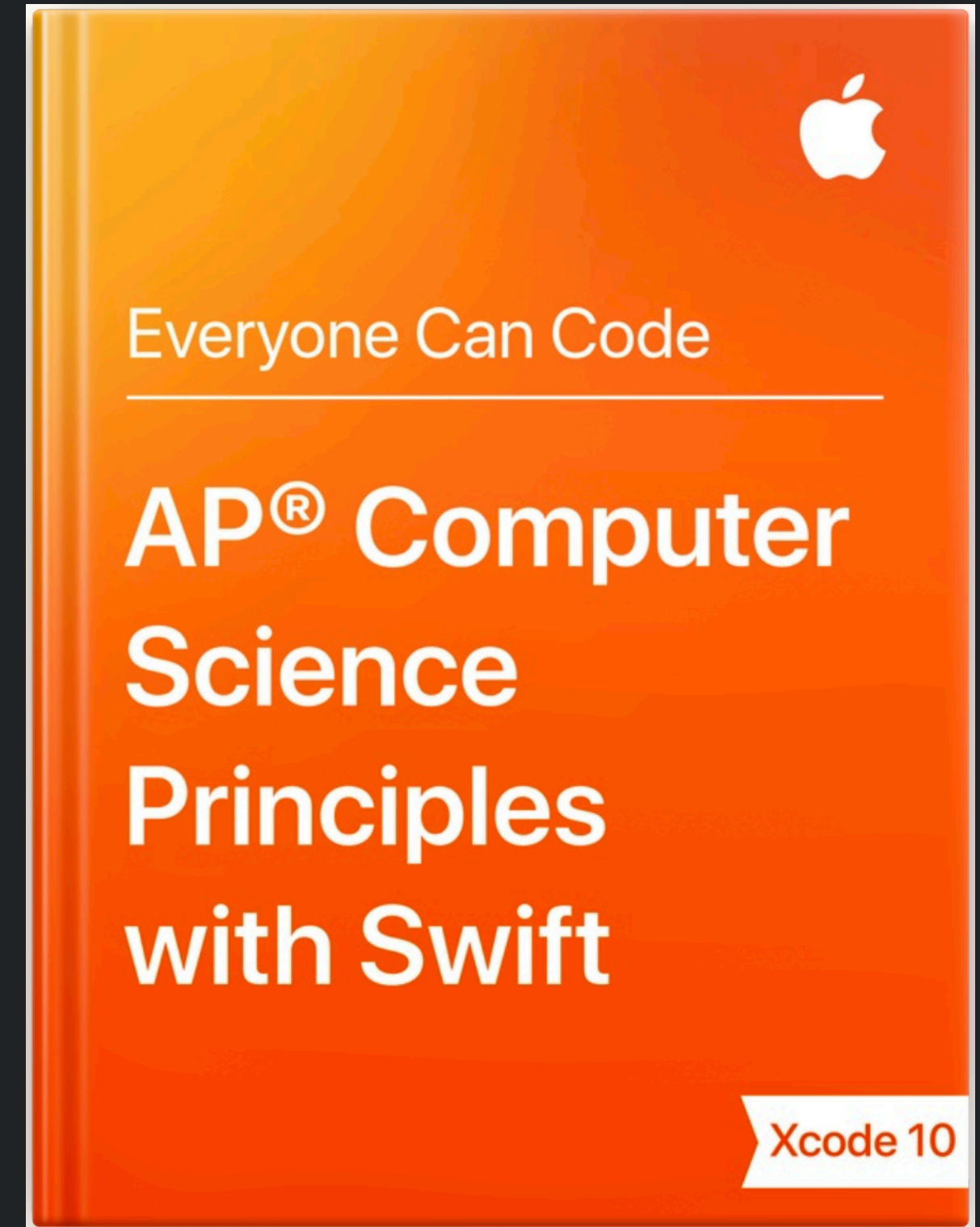
watchOS 1..<6

Name	Apple Watch
------	-------------

Purpose and/or Function	Tell time, track fitness
-------------------------	--------------------------

Beneficial Effect(s)	Promote health awareness
----------------------	--------------------------

Harmful Effect(s)	Potentially distracting
-------------------	-------------------------



watchOS 6

**More insightful.
More delightful.**



watchOS 6

watchOS 6

— Dedicated App Store

watchOS 6

- Dedicated App Store
- New APIs for developers

watchOS 6

- Dedicated App Store
- New APIs for developers
- Software Updates without iPhone

watchOS 6

- Dedicated App Store
- New APIs for developers
- Software Updates without iPhone
- New watch faces, Noise app

watchOS 6

- Dedicated App Store
- New APIs for developers
- Software Updates without iPhone
- New watch faces, Noise app
- Cycle Tracking app, Activity Trends

New APIs for developers

New APIs for developers

— Independent apps

New APIs for developers

- Independent apps
- Extended Runtime

New APIs for developers

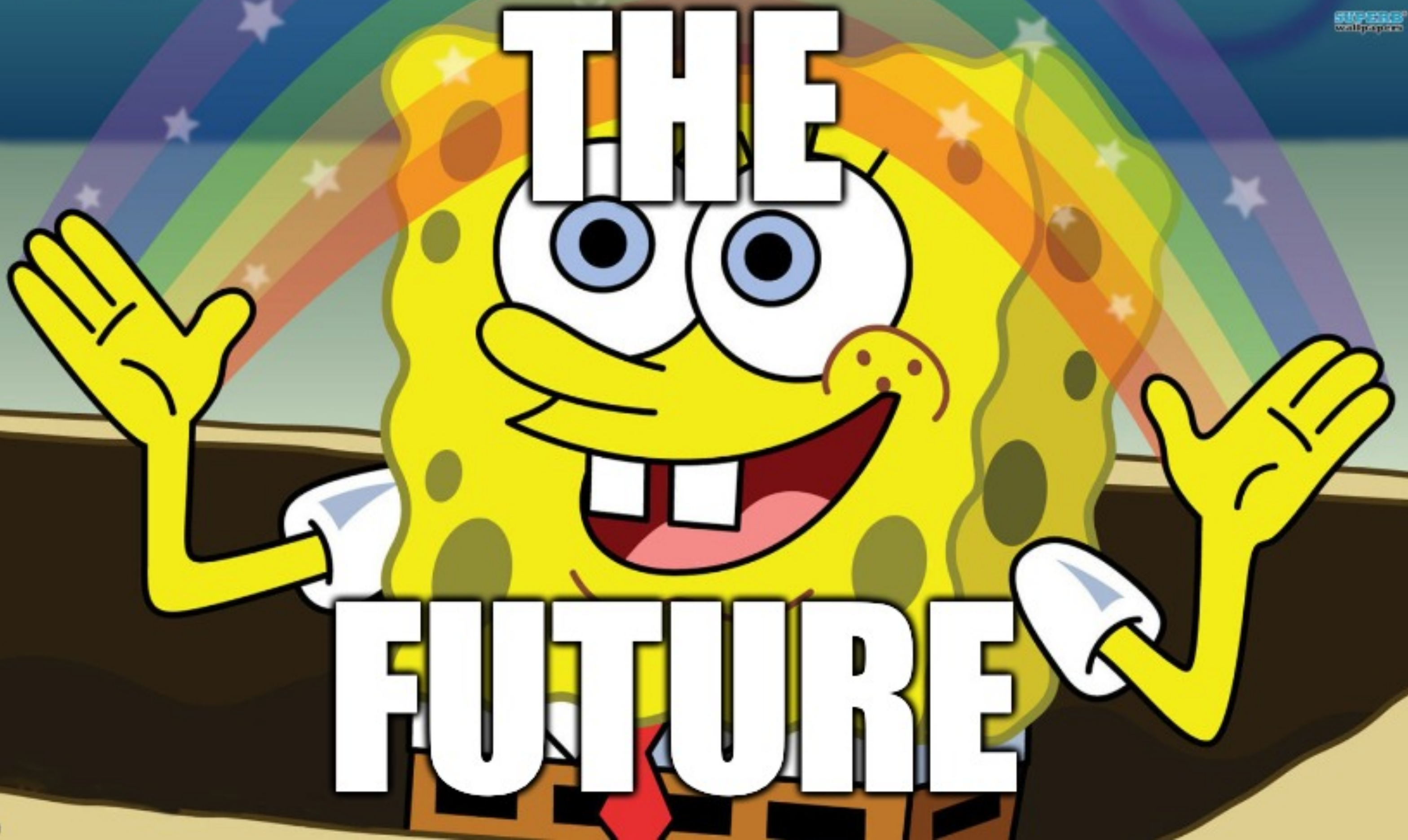
- Independent apps
- Extended Runtime
- Streaming audio API

New APIs for developers

- Independent apps
- Extended Runtime
- Streaming audio API
- Sign-in with Apple

New APIs for developers

- Independent apps
- Extended Runtime
- Streaming audio API
- Sign-in with Apple
- Core ML and the Neural Engine



Extended Runtime

Extended Runtime

- Provide runtime to complete targeted tasks

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care
 - Mindfulness

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care
 - Mindfulness
 - Physical Therapy

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care
 - Mindfulness
 - Physical Therapy
 - Health Monitoring

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care
 - Mindfulness
 - Physical Therapy
 - Health Monitoring
 - Alarm

Extended Runtime

- Provide runtime to complete targeted tasks
 - Self Care
 - Mindfulness
 - Physical Therapy
 - Health Monitoring
 - Alarm
- Preserve battery

Extended Runtime

Extended Runtime

— For non-workout activities

Extended Runtime

- For non-workout activities
- Move and Exercise rings not affected

Extended Runtime

- For non-workout activities
- Move and Exercise rings not affected
- Heart rate sensor off by default

Extended Runtime

```
let runtimeSession = WKExtendedRuntimeSession()
```

```
// Must be called while app is active  
runtimeSession.delegate = self  
runtimeSession.start()
```

```
runtimeSession.invalidate()
```

Extended Runtime

```
let runtimeSession = WKExtendedRuntimeSession()
```

```
// Must be called while app is active  
runtimeSession.delegate = self  
runtimeSession.start()
```

```
runtimeSession.invalidate()
```

Extended Runtime

```
let runtimeSession = WKExtendedRuntimeSession()
```

```
// Must be called while app is active  
runtimeSession.delegate = self  
runtimeSession.start()
```

```
runtimeSession.invalidate()
```

Streaming audio API

Streaming audio API

— HLS (via AVQueuePlayer)

Streaming audio API

- HLS (via AVQueuePlayer)
- Custom audio protocols

Streaming audio API

- HLS (via AVQueuePlayer)
- Custom audio protocols
- Network.framework

Streaming audio API

- HLS (via AVQueuePlayer)
- Custom audio protocols
- Network.framework
- URLSessionStreamingTask from URLSession

Streaming audio API

- HLS (via AVQueuePlayer)
- Custom audio protocols
- Network.framework
- URLSessionStreamingTask from URLSession
- URLSessionWebSocketTask (new in iOS 13 and watchOS 6)

Streaming audio API

- HLS (via AVQueuePlayer)
- Custom audio protocols
- Network.framework
- URLSessionStreamingTask from URLSession
- URLSessionWebSocketTask (new in iOS 13 and watchOS 6)
- AVPlayer and AVQueuePlayer

Sign-in with Apple

Sign-in with Apple

— Use Sign-in with Apple:

Sign-in with Apple

- Use Sign-in with Apple:
 - WKInterfaceAuthorizationAppleIDButton

Sign-in with Apple

- Use Sign-in with Apple:
 - WKInterfaceAuthorizationAppleIDButton
 - AuthenticationServices.framework

Sign-in with Apple

- Use Sign-in with Apple:
 - WKInterfaceAuthorizationAppleIDButton
 - AuthenticationServices.framework
- Create an app without user accounts

Sign-in with Apple

- Use Sign-in with Apple:
 - WKInterfaceAuthorizationAppleIDButton
 - AuthenticationServices.framework
- Create an app without user accounts
- Create custom form

Core ML and the Neural Engine

Core ML and the Neural Engine

- SoundAnalysis.framework
"Use the SoundAnalysis framework to analyze audio and recognize it as a particular type, such as laughter or applause"

Core ML and the Neural Engine

- SoundAnalysis.framework
"Use the SoundAnalysis framework to analyze audio and recognize it as a particular type, such as laughter or applause"
- MLUpdateTask et al
"A task that updates a model with additional training data"



THE FUTURE

Principles



Lightweight interactions



Holistic design



Personal communication

Glanceable

Glanceable

— readily available information

Glanceable

- readily available information
- no distractions

Glanceable

- readily available information
- no distractions
- up-to-date snapshots

Glanceable

- readily available information
- no distractions
- up-to-date snapshots
- provides a complication

Glanceable

- readily available information
- no distractions
- up-to-date snapshots
- provides a complication
- a custom notification interface

Actionable

Actionable

— mindful of the information

Actionable

- mindful of the information
- what's onscreen is always current and relevant

Actionable

- mindful of the information
- what's onscreen is always current and relevant
- refreshes in background

Actionable

- mindful of the information
- what's onscreen is always current and relevant
- refreshes in background
- custom actions in notifications

Responsive

Responsive

— interactions are quick

Responsive

- interactions are quick
- minimizes the launch time

Responsive

- interactions are quick
- minimizes the launch time
- responds with immediate feedback

Best practices

Best practices

Best practices

— Create account / Sign in

Best practices

- Create account / Sign in
- Request proper permissions (i.e HealthKit)

Best practices

- Create account / Sign in
- Request proper permissions (i.e HealthKit)
- Download data directly to the watch

Best practices

- Create account / Sign in
- Request proper permissions (i.e HealthKit)
- Download data directly to the watch
- Send Push Notifications directly to the watch

Best practices

The independent app can't use WatchConnectivity as its main source of data, and must be capable of accessing information on its own ^a

^a Ensure Your App Runs Independently, developer.apple.com

Extended Runtime (self-care)

Extended Runtime (self-care)

— Focused on health or emotional well-being

Extended Runtime (self-care)

- Focused on health or emotional well-being
- Sessions run up to 10 minutes

Extended Runtime (self-care)

- Focused on health or emotional well-being
- Sessions run up to 10 minutes
- Exiting app ends session

Extended Runtime (mindfulness)

Extended Runtime (mindfulness)

— Silent or near-silent meditation

Extended Runtime (mindfulness)

- Silent or near-silent meditation
- Sessions run up to 1 hour

Extended Runtime (mindfulness)

- Silent or near-silent meditation
- Sessions run up to 1 hour
- Exiting app ends session

Extended Runtime (physical therapy)

Extended Runtime (physical therapy)

- Stretching, strengthening, or range-of-motion exercises

Extended Runtime (physical therapy)

- Stretching, strengthening, or range-of-motion exercises
- Continues to run in the background

Extended Runtime (physical therapy)

- Stretching, strengthening, or range-of-motion exercises
- Continues to run in the background
- Sessions run up to 1 hour

Extended Runtime (physical therapy)

- Stretching, strengthening, or range-of-motion exercises
- Continues to run in the background
- Sessions run up to 1 hour
- Move and Exercise rings not affected

Extended Runtime (health monitoring)

Extended Runtime (health monitoring)

- Requires entitlement

Extended Runtime (health monitoring)

- Requires entitlement
- Use respectfully

Extended Runtime (health monitoring)

- Requires entitlement
- Use respectfully
- Continues in the background

Extended Runtime (alarm)

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm
- Can be scheduled up to 36 hours in advance

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm
- Can be scheduled up to 36 hours in advance
- Sessions run up to 30 minutes

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm
- Can be scheduled up to 36 hours in advance
- Sessions run up to 30 minutes
- Continues to run in the background

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm
- Can be scheduled up to 36 hours in advance
- Sessions run up to 30 minutes
- Continues to run in the background
- Must call `WKInterfaceDevice.play()`

Extended Runtime (alarm)

- Monitor for the optimal time to play an alarm
- Can be scheduled up to 36 hours in advance
- Sessions run up to 30 minutes
- Continues to run in the background
- Must call `WKInterfaceDevice.play()`
- Supports haptics

Streaming audio API

Streaming audio API

— WKInterfaceDevice.current().supportsAudioStreaming

Streaming audio API

- WKInterfaceDevice.current().supportsAudioStreaming
- Caching

Streaming audio API

- WKInterfaceDevice.current().supportsAudioStreaming
- Caching
- Reachability

Streaming audio API

- WKInterfaceDevice.current().supportsAudioStreaming
- Caching
- Reachability
- Network transitions

Streaming audio API

- WKInterfaceDevice.current().supportsAudioStreaming
- Caching
- Reachability
- Network transitions
- Audio Quality

Sign-in

Sign-in

— Support Password Autofill:

Sign-in

- Support Password Autofill:
 - WKTextContentType (username, password, newPassword, oneTimeCode)

Sign-in

- Support Password Autofill:
 - WKTextContentType (username, password, newPassword, oneTimeCode)
- Set up your app's associated domains

Sign-in

- Support Password Autofill:
 - WKTextContentType (username, password, newPassword, oneTimeCode)
- Set up your app's associated domains
- Add placeholders

Sign-in

Sign-in

— WKAlertAction for Terms & Conditions

Sign-in

- WKAlertAction for Terms & Conditions
- Embed TextField

Sign-in

- WKAlertAction for Terms & Conditions
- Embed TextField
- Use `textContentType`

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
        }
    }
}
```

Push Notifications

```
import WatchKit
import UserNotifications

class ExtensionDelegate: NSObject, WKExtensionDelegate {
    func applicationDidFinishLaunching() {
        let center = UNUserNotificationCenter.current()
        center.requestAuthorization(options: [.alert, .sound]) {
            (granted, error) in
                if granted {
                    WKExtension.shared().registerForRemoteNotifications()
                } else { ... }
            }
        }
    }
}
```

Push Notifications

```
func didRegisterForRemoteNotifications(withDeviceToken deviceToken: Data) {  
    /* Forward the token to your provider, using a custom method. */  
}  
  
func didFailToRegisterForRemoteNotificationsWithError(_ error: Error) {  
    /* Disable remote notification features */  
}
```

Push Notifications

```
func didRegisterForRemoteNotifications(withDeviceToken deviceToken: Data) {  
    /* Forward the token to your provider, using a custom method. */  
}
```

```
func didFailToRegisterForRemoteNotificationsWithError(_ error: Error) {  
    /* Disable remote notification features */  
}
```

Push Notifications

```
func didRegisterForRemoteNotifications(withDeviceToken deviceToken: Data) {  
    /* Forward the token to your provider, using a custom method. */  
}
```

```
func didFailToRegisterForRemoteNotificationsWithError(_ error: Error) {  
    /* Disable remote notification features */  
}
```

Push Notifications

```
func didReceiveRemoteNotification(_ userInfo: [AnyHashable : Any],
    fetchCompletionHandler: @escaping
        (WKBackgroundFetchResult) -> Void) {
    // handle it
    // ...
    fetchCompletionHandler(.newData)
}
```

Push Notifications

```
func didReceiveRemoteNotification(_ userInfo: [AnyHashable : Any],  
    fetchCompletionHandler: @escaping  
        (WKBackgroundFetchResult) -> Void) {  
    // handle it  
    // ...  
    fetchCompletionHandler(.newData)  
}
```

Push Notifications

```
func didReceiveRemoteNotification(_ userInfo: [AnyHashable : Any],
    fetchCompletionHandler: @escaping
        (WKBackgroundFetchResult) -> Void) {
    // handle it
    // ...
    fetchCompletionHandler(.newData)
}
```

Push Notifications

```
func didReceiveRemoteNotification(_ userInfo: [AnyHashable : Any],  
    fetchCompletionHandler: @escaping  
        (WKBackgroundFetchResult) -> Void) {  
    // handle it  
    // ...  
    fetchCompletionHandler(.newData)  
}
```

Push Notifications

```
func didReceiveRemoteNotification(_ userInfo: [AnyHashable : Any],  
    fetchCompletionHandler: @escaping  
        (WKBackgroundFetchResult) -> Void) {  
    // handle it  
    // ...  
    fetchCompletionHandler(.newData)  
}
```

Watch App Store



Text

Optim�ization



Graphics Optimization



Rating and Reviews



Getting

Featured

Summary

Summary

— Apple Watch ≠ iPhone's extension

Summary

- Apple Watch ≠ iPhone's extension
- Check out the new APIs

Summary

- Apple Watch ≠ iPhone's extension
- Check out the new APIs
- Extended Runtime is very powerful

Summary

- Apple Watch ≠ iPhone's extension
- Check out the new APIs
- Extended Runtime is very powerful
- Think about ASO

Not enough?

Not enough?

— Streaming Audio on watchOS 6, session 716

Not enough?

- Streaming Audio on watchOS 6, session 716
- Extended Runtime for watchOS Apps, session 251

Not enough?

- [Streaming Audio on watchOS 6, session 716](#)
- [Extended Runtime for watchOS Apps, session 251](#)
- [Creating Independent Watch Apps, session 208](#)

questions?

drobinin.com | @valzevul