MARTIANCRAFT

NOTIFICATION HANDLING ON WATCHKIT

Eric Blair
@jablair
eric@martiancraft.com

Why Focus on WatchKit?

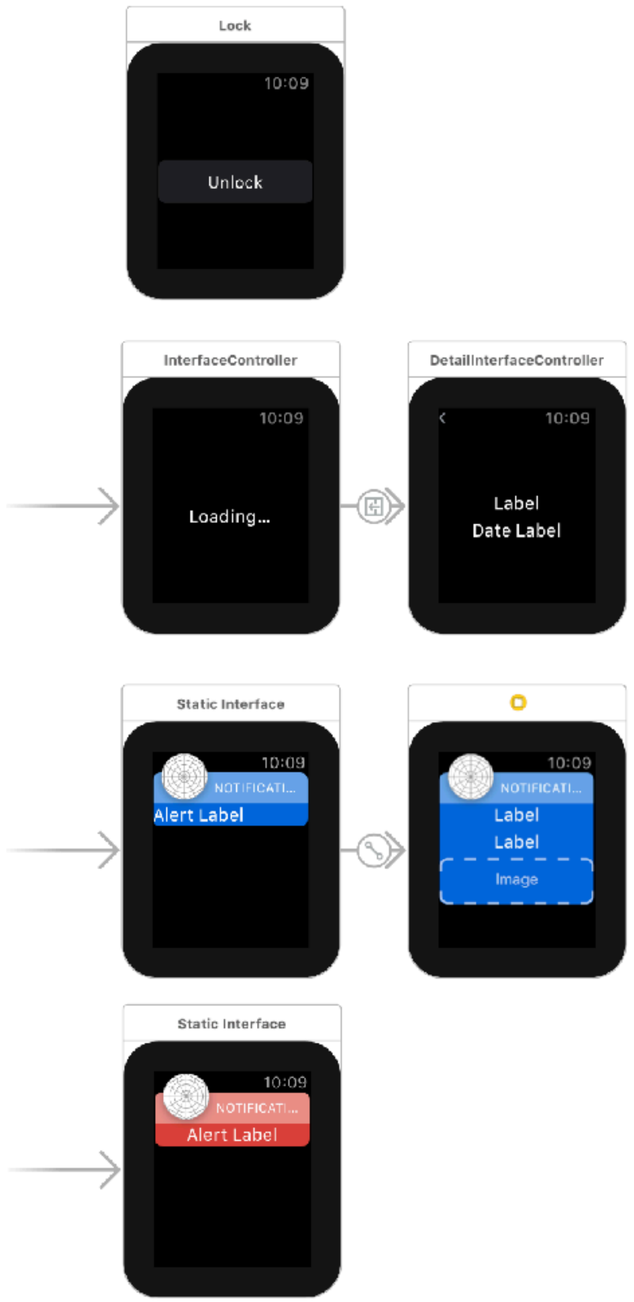
Key interaction mode on the watch

Inspired by difficulties on real world projects

Limitations of WatchKit require different approaches than UIKit

Demo App

- Mirrors real-world client projects
  - Dual-mode application
  - Specific UI, notifications, actions for each mode
- Built with iOS 10 `UserNotifications` framework
- Supports multiple notification triggers
  - Push notifications (via NWPusher)
  - Phone-triggered local notification
  - Watch-triggered local notification

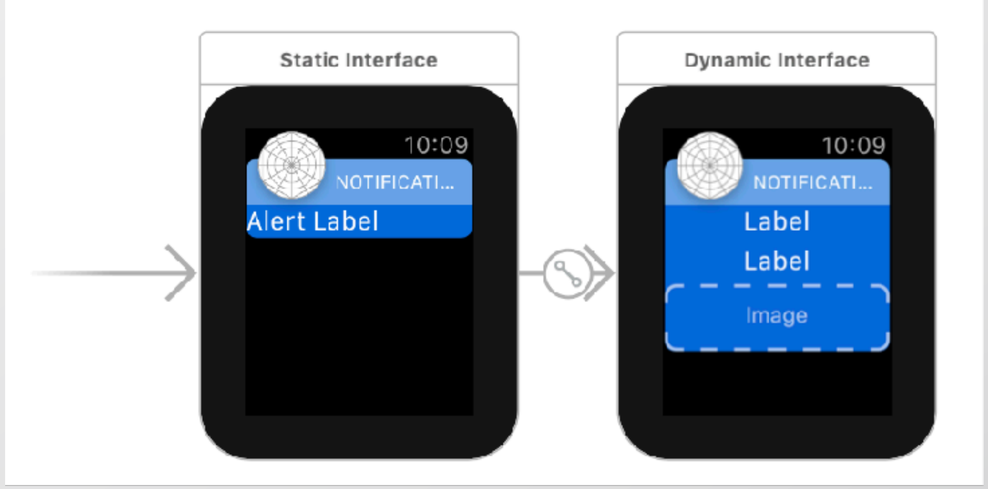


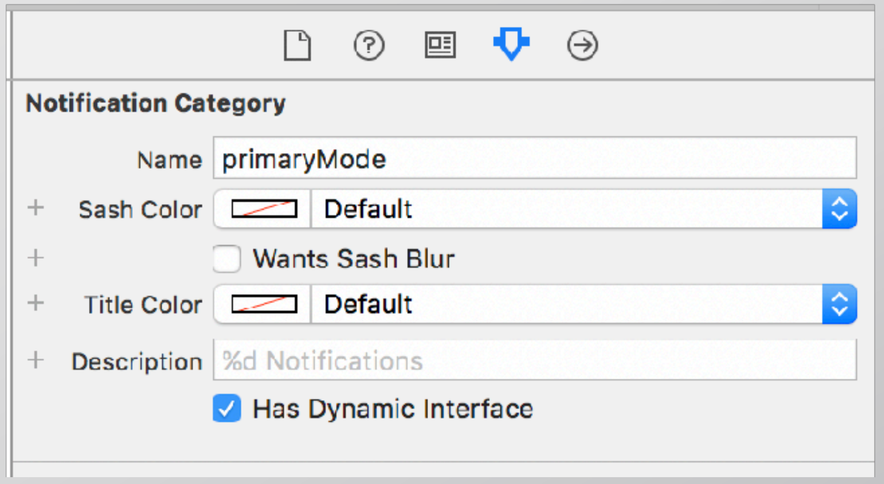
Configuring Notifications

- UNNotificationCategory
- UNNotificationAction
- Set notification categories on the current notification center

Watch Notification Display

- Generic, Static, Dynamic Display
- Determined by notification category
- `WKUserNotificationInterfaceController` subclass
- Limited time to configure





Populating Dynamic Notification

```
typealias ReceivedHandler = (WKUserNotificationInterfaceType) -> Void

override func didReceive(_ notification: UNNotification,
                         withCompletion completionHandler: @escaping ReceivedHandler)  {
    // Do stuff to populate your notifications UI
    // Set up label content, image content, etc, etc

    completionHandler(.custom)
}
```

Handling Notifications

- Notification processed by `UNUserNotificationCenterDelegate`
- Must be configured prior to completion of app launch
- Two scenarios for notification handling
  - Application opened from notification
  - Notification received while application open

Opening App from Notification

```
func userNotificationCenter(_ center: UNUserNotificationCenter,
                            didReceive response: UNNotificationResponse,
                            withCompletionHandler completionHandler: @escaping () -> Void) {
    // Check for real action identifier
    guard !response.actionIdentifier.isEmpty else { return }

    // Do stuff with the notification

    completionHandler()
}
```

Receiving Notification when App Open

- Ignore
- Process
  - In both cases, the non-watch notifications stay in the notification center
- Display Notification

Responding to Notification when App Open

4.0

```
typealias Presenter = (UNNotificationPresentationOptions) -> Void
func userNotificationCenter(_ center: UNUserNotificationCenter,
                            willPresent notification: UNNotification,
                            withCompletionHandler completionHandler: @escaping Presenter) {
    let shouldDisplay: Bool = …

    let presentationOptions: UNNotificationPresentationOptions
    if shouldDisplay {
        // Just show the notification
        presentationOptions = [.sound, .alert]
    }
    else {
        presentationOptions = []
        let message = ClearNotificationCommand(identifier: notification.request.identifier)
        self.connectivityManager?.send(message: message)
        // Do stuff with the notification
    }
    completionHandler(presentationOptions)
}
```

Responding to Notification when App Open

4.0

```
typealias Presenter = (UNNotificationPresentationOptions) -> Void
func userNotificationCenter(_ center: UNUserNotificationCenter,
                            willPresent notification: UNNotification,
                            withCompletionHandler completionHandler: @escaping Presenter) {
    let shouldDisplay: Bool = …

    let presentationOptions: UNNotificationPresentationOptions
    if shouldDisplay {
        // Just show the notification
        presentationOptions = [.sound, .alert]
    }
    else {
        presentationOptions = [.sound]
        let message = ClearNotificationCommand(identifier: notification.request.identifier)
        self.connectivityManager?.send(message: message)
        // Do stuff with the notification
    }
    completionHandler(presentationOptions)
}
```

Responding to Notification when App Open

```
func didReceive(message: [String: Any], replyHandler: MessageReplyHandler? = nil) {
    switch command {
    case .clearNotification:
        guard let clearNotification = ClearNotificationCommand(messageRepresentation: message)
            else { return }

        let id = clearNotification.identifier
        UNUserNotificationCenter.current().removeDeliveredNotifications(withIdentifiers: [id])
    }
}
```

What About `handleAction(withIdentifier:for:)`?

Notification Action Types

- Handled in the current app state
- Request a clean app state
- System URL triggers

Handle in Current State

- Apps with a flat hierarchy
- Actions that can run regardless of the state of the root interface controller

Notification Handling Protocol

```
protocol NotificationHandleable {
    func canHandle(notification: UNNotification, with response: UNNotificationResponse?) -> Bool
    func handle(notification: UNNotification, with response: UNNotificationResponse?)
}
```

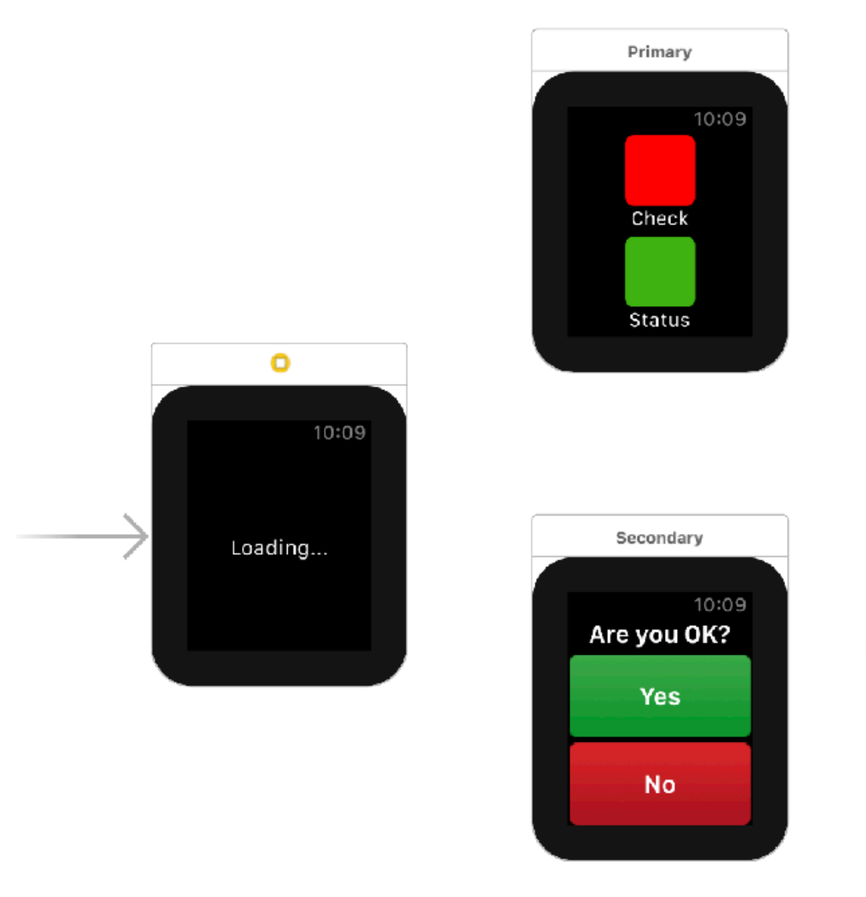
Main Interface Implementation

```
func handle(notification: UNNotification, with response: UNNotificationResponse? = nil) {
    let rawCategory = notification.request.content.categoryIdentifier
    guard let category = UserNotificationCategory(rawValue: rawCategory) else { return }

    let action = response.flatMap { UserNotificationAction(rawValue:  $0.actionIdentifier) }

    switch self.mode {
    case .primary where category == .primaryMode && action == .modal:
        self.dismiss() // Dismisses a currently displayed modal
        self.presentController(withName: DetailInterfaceController.identifer, context: notification)
    case .secondary where category == .secondaryMode:
        self.notificationReceivedLabel.setHidden(false)
        self.notificationReceivedDateLabel.setHidden(false)
        let notificationData = InterfaceController.dateFormatter.string(from: notification.date)
        self.notificationReceivedDateLabel.setText(notificationDate)
    default:
        print ("No handle-able notifications for combination of mode = \(self.mode), category =
                \(rawCategory), action = \(action?.rawValue ?? "none")")
    }
}
```

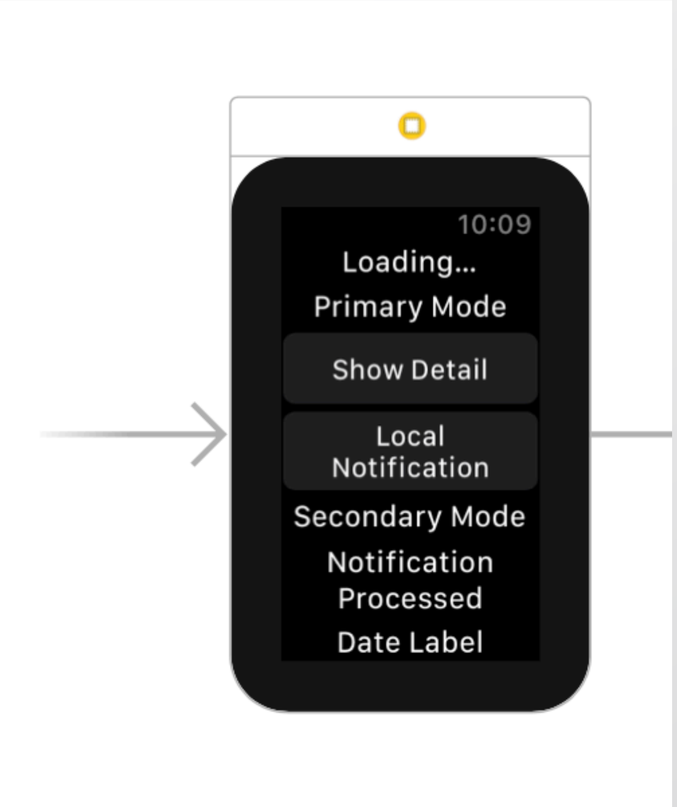
Request a Clean State

Reload the root interface controller and pass in the notification info as part of the content

LAUNCH INTERFACE



DEMO LAUNCH INTERFACE




8
30
RUNNING TODAY?

Notification Info

```
struct NotificationInfo {
   let notification: UNNotification
   let response: UNNotificationResponse?
}
```

Capturing the Notification

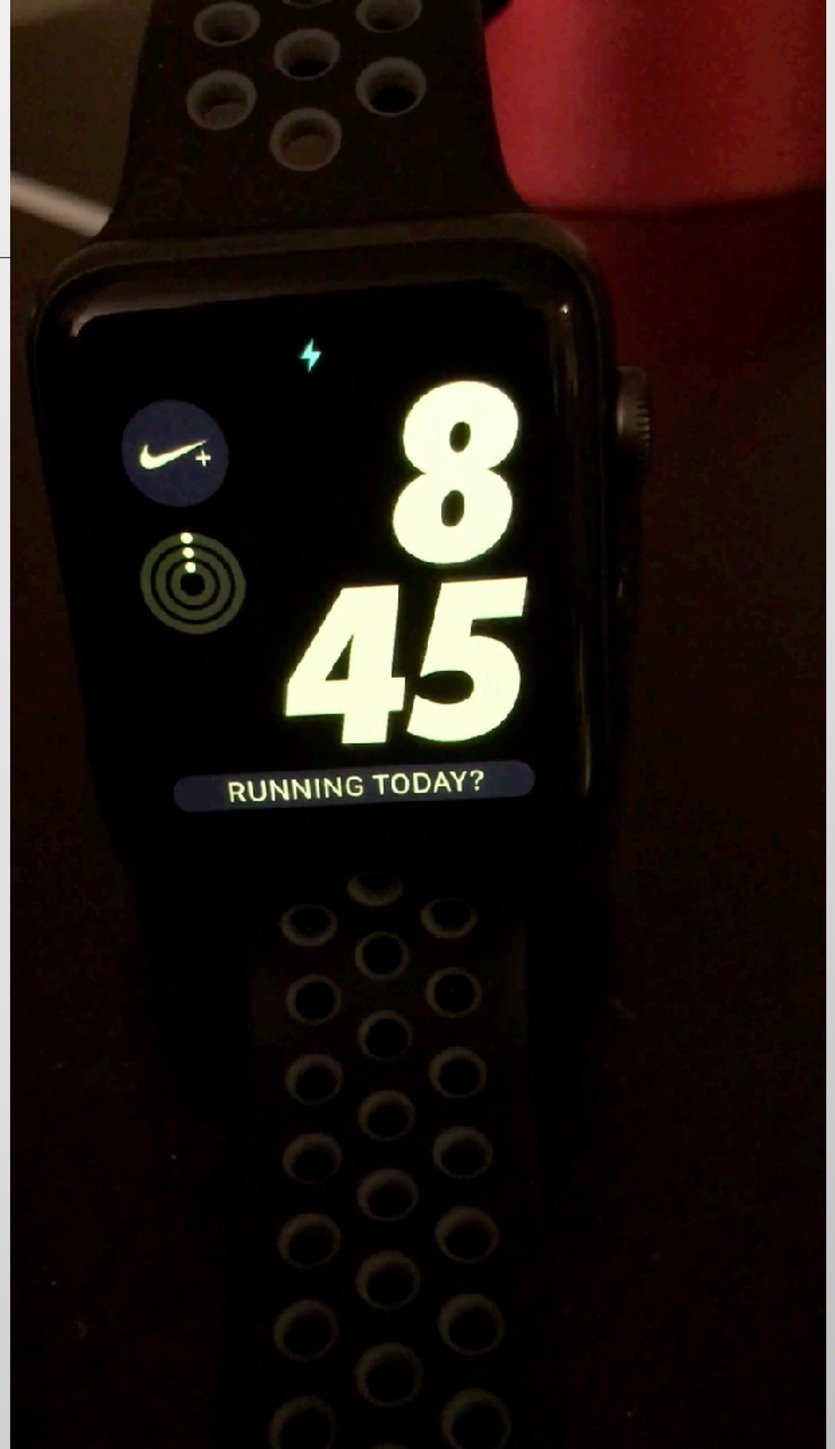
```
override func awake(withContext context: Any?) {
    super.awake(withContext: context)

    self.pendingNotificationInfo = context as? NotificationInfo
}
```

Reacting to the Notification

```
override func didAppear() {
    super.didAppear()

    guard
        let notificationInfo = self.pendingNotificationInfo,
        let rawCategory = notificationInfo.notification.request.content.categoryIdentifier,
        let category = UserNotificationCategory(rawValue: rawCategory),
        self.mode != .undefined else { return }

    switch mode {
    case .primary where category == .primaryMode:
        if notificationInfo.action == .nil {
            self.pushController(withName: DetailInterfaceController.identifer, context: notificationInfo.notification)
        }

        default:
            assertionFailure("Unsupported mode / category combination = \(mode) / \(category)")
    }
    self.pendingNotificationInfo = nil
}
```

System URL Triggers

- Opening a tel: or sms: URL
- `WKExtension.openSystemURL(_:)`
- Seems like something that could be handled regardless of the app state




THBBFT!

System URL Triggers

`WKExtension.openSystemURL(_:)`

does not play well with others

```
let rawCategory = notification.request.content.categoryIdentifier
let category = UserNotificationCategory(rawValue: rawCategory)

if category == .primaryMode && action == .call {
    DispatchQueue.main.asyncAfter(deadline: DispatchTime.now() + .seconds(1)) {
        if let phoneNumberURL = URL(string: "tel:8675309") {
            WKExtension.shared().openSystemURL(phoneNumberURL)
        }
    }
}
```

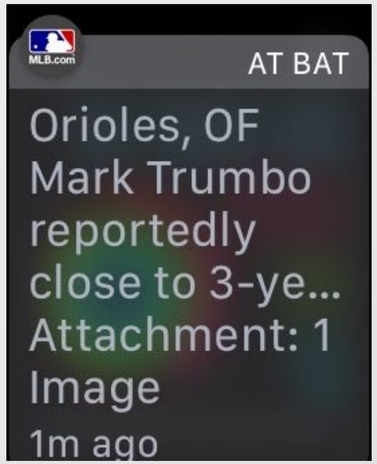
System URL Triggers

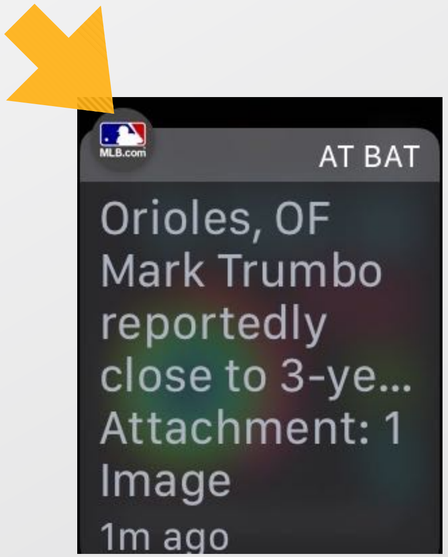
`WKExtension.openSystemURL(_:)`

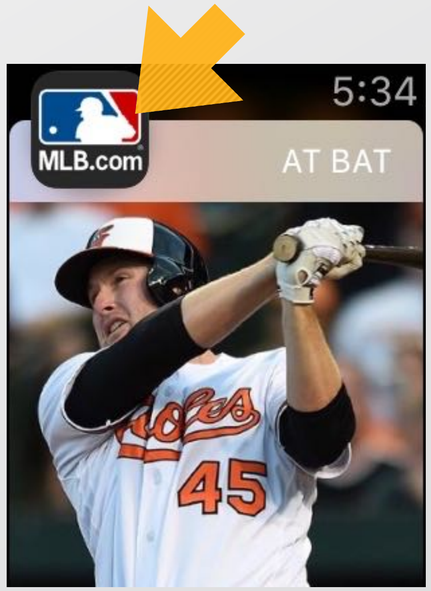
does not play well with others

```
let rawCategory = notification.request.content.categoryIdentifier
let category = UserNotificationCategory(rawValue: rawCategory)

if category == .primaryMode && action == .call {
    DispatchQueue.main.asyncAfter(deadline: DispatchTime.now() + .seconds(1)) {
        if let phoneNumberURL = URL(string: "tel:8675309") {
            WKExtension.shared().openSystemURL(phoneNumberURL)
        }
    }
}
```

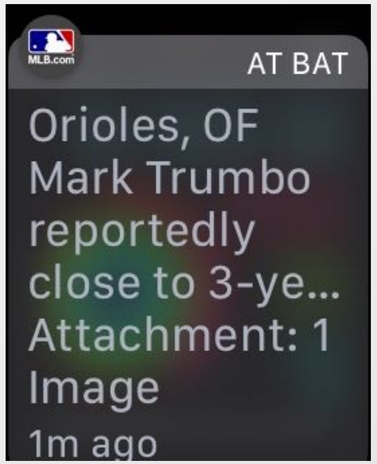
Notification Attachments


MLB.com
AT BAT
Orioles, OF Mark Trumbo reportedly close to 3-ye...
Attachment: 1 Image
1m ago

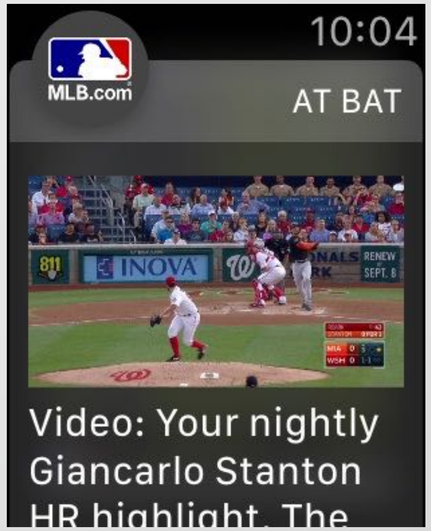

MLB.com
AT BAT
Orioles, OF Mark Trumbo reportedly close to 3-ye...
Attachment: 1 Image
1m ago


5:34
MLB.com
AT BAT
45

4.0


MLB.com
AT BAT
Orioles, OF Mark Trumbo reportedly close to 3-ye...
Attachment: 1 Image
1m ago


5:34
MLB.com
AT BAT
45


10:04
MLB.com
AT BAT
INOVA
Video: Your nightly Giancarlo Stanton HR highlight. The

Dynamic Notification Attachments

```
override func didReceive(_ notification: UNNotification, withCompletion completionHandler: @escaping (WKUserNotificationInterfaceType) -> Swift.Void)
{
    if #available(watchOS 4, *), !notification.request.content.attachments.isEmpty {
       completionHandler(.default)
    }

    if let notificationIdentifier = notification.request.content.userInfo[UserNotificationInfoKey.notificationAttachmentIdentifier.rawValue] {
        var attachmentURLComponents = URLComponents()
        attachmentURLComponents.scheme = "https"
        attachmentURLComponents.host = “host.com"
        attachmentURLComponents.path = "/\(notificationIdentifier)-Watch.jpg"

        guard let imageURL = attachmentURLComponents.url else {
           completionHandler(.custom)
           return
        }

        let attachmentDownloadSession = URLSession.shared.dataTask(with: imageURL) { (data: Data?, _, error: Error?) in
            defer { completionHandler(.custom) }

            guard let data = data, image = UIImage(data: data) else { return }

            self.notificationImage.setImage(image)
        }
        attachmentDownloadSession.resume()
    }
}
```

Sample Code

https://github.com/jablair/360iDev-WatchKitNotifications

THANK YOU

martiancraft.com

Eric Blair @jablair eric@martiancraft.com