Advanced Notifications
Session 708

Michele Campeotto
iOS Notifications
Agenda

Notifications User Interface
Media Attachments
Customization
Notifications User Interface
Order 4000 coffees
Invitation From Matt
June 15, 2016 at 9:00 AM
Matt

did you order the coffees?

Slide for more
Order 4000 coffees
Invitation From Matt
June 15, 2016 at 9:00 AM...
Press for more
<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>9 AM</td>
<td>Order 4000 coffees</td>
</tr>
<tr>
<td>10 AM</td>
<td>Feed the lemurs</td>
</tr>
</tbody>
</table>

**Order 4000 coffees**
Invitation From Matt  
June 15, 2016 at 9:00 AM  
Moscone Center  
747 Howard St, San Francisco, CA 94103, United States
Matt
Did you order the coffees?
Media Attachments
Media Attachments

Push Payload
Media Attachments

```json
{
  aps: {
    alert: { ... },
    mutable-content: 1
  }
  my-attachment: "https://example.com/photo.jpg"
}
```
Media Attachments

```
{
    aps: {
        alert: { … },
        mutable-content: 1
    }
    my-attachment: "https://example.com/photo.jpg"
}
```
Media Attachments

{  
  aps: {  
    alert: { ... },  
    mutable-content: 1  
  }  
  my-attachment: "https://example.com/photo.jpg"  
}
Media Attachments

Push Payload ➔ Service Extension
Media Attachments

Push Payload → Service Extension

Hey guys
What's going on here?
// Adding an attachment to a user notification

public class NotificationService: UNNotificationServiceExtension {
    override public func didReceive(_ request: UNNotificationRequest, 
        withContentHandler contentHandler: (UNNotificationContent) -> Void) {

        let fileURL = // ...

        let attachment = UNNotificationAttachment(identifier: "image", 
            url: fileURL, 
            options: nil)

        let content = request.content.mutableCopy as! UNMutableNotificationContent
        content.attachments = [attachment]

        contentHandler(content)
    }
}
public class NotificationService: UNNotificationServiceExtension {
  override public func didReceive(_ request: UNNotificationRequest,
         withContentHandler contentHandler: (UNNotificationContent) -> Void) {

    let fileURL = // ...

    let attachment = UNNotificationAttachment(identifier: "image",
               url: fileURL,
               options: nil)

    let content = request.content.mutableCopy as! UNMutableNotificationContent
    content.attachments = [ attachment ]

    contentHandler(content)
  }
}
// Adding an attachment to a user notification

public class NotificationService: UNNotificationServiceExtension {
    override public func didReceive(_ request: UNNotificationRequest,
                                       withContentHandler contentHandler: (UNNotificationContent) -> Void) {

        let fileURL = // ...

        let attachment = UNNotificationAttachment(identifier: "image",
                                                   url: fileURL,
                                                   options: nil)

        let content = request.content.mutableCopy as! UNMutableNotificationContent
        content.attachments = [ attachment ]

        contentHandler(content)
    }
}
public class NotificationService: UNNotificationServiceExtension {
    override public func didReceive(_ request: UNNotificationRequest, withContentHandler contentHandler: (UNNotificationContent) -> Void) {

        let fileURL = // ...

        let attachment = UNNotificationAttachment(identifier: "image", url: fileURL, options: nil)

        let content = request.content.mutableCopy as! UNMutableNotificationContent
        content.attachments = [ attachment ]

        contentHandler(content)
    }
}
// Adding an attachment to a user notification

public class NotificationService: UNNotificationServiceExtension {
    override public func didReceive(_ request: UNNotificationRequest,
        withContentHandler contentHandler: (UNNotificationContent) -> Void) {

        let fileURL = // ...

        let attachment = UNNotificationAttachment(identifier: "image",
            url: fileURL,
            options: nil)

        let content = request.content.mutableCopy as! UNMutableNotificationContent
        content.attachments = [ attachment ]

        contentHandler(content)
    }
}


// Adding an attachment to a user notification

public class NotificationService: UNNotificationServiceExtension {
    override public func didReceive(_ request: UNNotificationRequest,
                                          withContentHandler contentHandler: (UNNotificationContent) -> Void) {

        let fileURL = // ...

        let attachment = UNNotificationAttachment(identifier: "image",
                                                    url: fileURL,
                                                    options: nil)

        let content = request.content.mutableCopy as! UNMutableNotificationContent
        content.attachments = [ attachment ]

        contentHandler(content)
    }
}
Media Attachments

Local and remote notifications
Media Attachments

Local and remote notifications

Image, audio, video
Media Attachments

Local and remote notifications
Image, audio, video
Download in the service extension
Media Attachments

Local and remote notifications
Image, audio, video
Download in the service extension
• Limited processing time and size
Media Attachments

Local and remote notifications
Image, audio, video
Download in the service extension
• Limited processing time and size
Add attachment to notification
Media Attachments

Local and remote notifications
Image, audio, video
Download in the service extension
• Limited processing time and size
Add attachment to notification
• File is moved and managed by the system
GIFs work
Did U say GIF?
Custom User Interface
Custom User Interface
Custom User Interface

Notification content extension
Custom User Interface

Notification content extension

Custom views
Custom User Interface

Notification content extension
Custom views
No interaction
Custom User Interface

Notification content extension
Custom views
No interaction
Respond to notification actions
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM...
Press for more
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM...
Press for more
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM
Moscone Center
747 Howard St, San Francisco, CA 94103, United States

Accept
Maybe
Decline
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM
Moscone Center
747 Howard St, San Francisco, CA 94103, United States

Accept
Maybe
Decline
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM
Moscone Center
747 Howard St, San Francisco, CA 94103, United States

Accept
Maybe
Decline
Order 4000 coffees
Invitation From Michele Campeotto
June 15, 2016 at 9:00 AM
Moscone Center
747 Howard St, San Francisco, CA 94103, United States

Accept
Maybe
Decline
Notification Content Extension
Choose a template for your new target:

iOS
- Application
- Framework & Library
- Application Extension
- Test

watchOS
- Application
- Framework & Library
- Application Extension
- Test

tvOS
- Application
- Framework & Library
- Application Extension
- Test

OS X
- Application
- Framework & Library
- Application Extension
- Test

Notification Content Extension
This template builds a Notification Content application extension.
Choose a template for your new target:

**iOS**
- Application
- Framework & Library
- Application Extension
- Test

**watchOS**
- Application
- Framework & Library

**tvOS**
- Application
- Framework & Library
- Application Extension
- Test

**OS X**
- Application
- Framework & Library

**Notification Content Extension**
This template builds a Notification Content application extension.
// NotificationViewController.swift
// EventNotification
//
// Created by Michele Campeotto on 6/1/16.
// Copyright © 2016 Apple. All rights reserved.
//
import UIKit
import UserNotifications
import UserNotificationsUI

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var label: UILabel?

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any required interface initialization here.
    }

    func didReceive(_ notification: UNNotification) {
        self.label?.text = notification.request.content.body
    }
}
// Minimal Content Extension

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var label: UILabel?

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any required interface initialization here.
    }

    func didReceive(_ notification: UNNotification) {
        label?.text = notification.request.content.body
    }
}

class NotificationViewController: UIViewController, UNNotificationContentExtension {

  @IBOutlet var label: UILabel?

  override func viewDidLoad() {
    super.viewDidLoad()
    // Do any required interface initialization here.
  }

  func didReceive(_ notification: UNNotification) {
    label?.text = notification.request.content.body
  }
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var label: UILabel?

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any required interface initialization here.
    }

    func didReceive(_ notification: UNNotification) {
        label?.text = notification.request.content.body
    }
}
### Extension Info.plist

Here is the content of the `Info.plist` file:

```xml
<plist version="1.0">
  <dict>
    <!-- ... -->
    <key>NSExtension</key>
    <dict>
      <key>NSExtensionAttributes</key>
      <dict>
        <key>UNNotificationExtensionCategory</key>
        <string>event-invite</string>
      </dict>
    </dict>
  </dict>
</plist>
```

### Key-Value Table

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localization native development region</td>
<td>String</td>
<td>en</td>
</tr>
<tr>
<td>Bundle display name</td>
<td>String</td>
<td>EventNotification</td>
</tr>
<tr>
<td>Executable file</td>
<td>String</td>
<td>$(EXECUTABLE_NAME)</td>
</tr>
<tr>
<td>Bundle identifier</td>
<td>String</td>
<td>$(PRODUCT_BUNDLE_IDENTIFIER)</td>
</tr>
<tr>
<td>InfoDictionary version</td>
<td>String</td>
<td>6.0</td>
</tr>
<tr>
<td>Bundle name</td>
<td>String</td>
<td>$(PRODUCT_NAME)</td>
</tr>
<tr>
<td>Bundle OS Type code</td>
<td>String</td>
<td>XPC!</td>
</tr>
<tr>
<td>Bundle versions string, short</td>
<td>String</td>
<td>1.0</td>
</tr>
<tr>
<td>Bundle creator OS Type code</td>
<td>String</td>
<td>?? ??</td>
</tr>
<tr>
<td>Bundle version</td>
<td>String</td>
<td>1</td>
</tr>
</tbody>
</table>

### NSExtension

- **NSExtensionAttributes**
  - **UNNotificationExtensionCategory** String: event-invite
  - **NSExtensionMainStoryboard** String: MainInterface
  - **NSExtensionPointIdentifier** String: com.apple.usernotifications.content-extension
```
<plist version="1.0">
<dict>
  ...
  <key>NSExtension</key>
  <dict>
    <key>NSExtensionAttributes</key>
    <dict>
      <key>UNNotificationExtensionCategory</key>
      <string>event-invite</string>
    </dict>
    ...
  </dict>
</dict>
</plist>
```
```xml
<plist version="1.0">
  <dict>
    <key>NSExtension</key>
    <dict>
      <key>NSExtensionAttributes</key>
      <dict>
        <key>UNNotificationExtensionCategory</key>
        <array>
          <string>event-invite</string>
          <string>event-changed</string>
        </array>
      </dict>
    </dict>
  </dict>
</plist>
```
Custom Views
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventTitle: UILabel!
    @IBOutlet var eventDate: UILabel!
    @IBOutlet var eventLocation: UILabel!
    @IBOutlet var eventMessage: UILabel!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content
        eventTitle.text = content.title
        eventDate.text = content.subtitle
        eventMessage.text = content.body

        if let location = content.userInfo["location"] as? String {
            eventLocation.text = location
        }
    }
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventTitle: UILabel!
    @IBOutlet var eventDate: UILabel!
    @IBOutlet var eventLocation: UILabel!
    @IBOutlet var eventMessage: UILabel!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content

        eventTitle.text = content.title
        eventDate.text = content.subtitle
        eventMessage.text = content.body

        if let location = content.userInfo["location"] as? String {
            eventLocation.text = location
        }
    }
}
// Notification Content Extension

class NotificationViewController: UIViewController, UNNotificationContentExtension {

@IBOutlet var eventTitle: UILabel!
@IBOutlet var eventDate: UILabel!
@IBOutlet var eventLocation: UILabel!
@IBOutlet var eventMessage: UILabel!

func didReceive(_ notification: UNNotification) {
    let content = notification.request.content
    eventTitle.text = content.title
    eventDate.text = content.subtitle
    eventMessage.text = content.body
    if let location = content.userInfo["location"] as? String {
        eventLocation.text = location
    }
}
}
// Notification Content Extension

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventTitle: UILabel!
    @IBOutlet var eventDate: UILabel!
    @IBOutlet var eventLocation: UILabel!
    @IBOutlet var eventMessage: UILabel!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content

        eventTitle.text = content.title
        eventDate.text = content.subtitle
        eventMessage.text = content.body

        if let location = content.userInfo["location"] as? String {
            eventLocation.text = location
        }
    }
}
// Notification Content Extension

class NotificationViewController: UIViewController, UNNotificationContentExtension {

@IBOutlet var eventTitle: UILabel!
@IBOutlet var eventDate: UILabel!
@IBOutlet var eventLocation: UILabel!
@IBOutlet var eventMessage: UILabel!

func didReceive(_ notification: UNNotification) {
    let content = notification.request.content
    eventTitle.text = content.title
    eventDate.text = content.subtitle
    eventMessage.text = content.body

    if let location = content.userInfo?["location"] as? String {
        eventLocation.text = location
    }
}
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
```xml
<key>NSExtensionAttributes</key>
<dict>
  <key>UNNotificationExtensionCategory</key>
  <string>event-invite</string>
  <key>UNNotificationExtensionDefaultContentHidden</key>
  <true/>
</dict>
```
```
<key>NSExtensionAttributes</key>
<dict>
  <key>UNNotificationExtensionCategory</key>
  <string>event-invite</string>
  <key>UNNotificationExtensionDefaultContentHidden</key>
  <true/>
</dict>
```
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    override func viewDidLoad() {
        super.viewDidLoad()

        let size = view.bounds.size
        preferredContentSize = CGSize(width: size.width, height: size.width / 2)
    }

    func didReceive(_ notification: UNNotification) {
        // ...
    }
}

// Notification Content Extension
// Notification Content Extension

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    override func viewDidLoad() {
        super.viewDidLoad()
        let size = view.bounds.size
        preferredContentSize = CGSize(width: size.width, height: size.width / 2)
    }

    func didReceive(_ notification: UNNotification) {
        // ...
    }
}

The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSExtensionAttributes</td>
<td>Dictionary</td>
<td>(3 items)</td>
</tr>
<tr>
<td>UNNotificationExtensionInitialContentSizeRatio</td>
<td>Number</td>
<td>0.5</td>
</tr>
<tr>
<td>UNNotificationExtensionDefaultContentHidden</td>
<td>Boolean</td>
<td>YES</td>
</tr>
<tr>
<td>UNNotificationExtensionCategory</td>
<td>String</td>
<td>event-invite</td>
</tr>
<tr>
<td>NSExtensionMainStoryboard</td>
<td>String</td>
<td>MainInterface</td>
</tr>
<tr>
<td>NSExtensionPointIdentifier</td>
<td>String</td>
<td>com.apple.usernotifications.content-extension</td>
</tr>
<tr>
<td>Key</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td><strong>Information Property List</strong></td>
<td>Dictionary</td>
<td>(11 items)</td>
</tr>
<tr>
<td>Localization native development region</td>
<td>String</td>
<td>en</td>
</tr>
<tr>
<td>Bundle display name</td>
<td>String</td>
<td>EventNotification</td>
</tr>
<tr>
<td>Executable file</td>
<td>String</td>
<td>${EXECUTABLE_NAME}</td>
</tr>
<tr>
<td>Bundle identifier</td>
<td>String</td>
<td>${PRODUCT_BUNDLE_IDENTIFIER}</td>
</tr>
<tr>
<td>InfoDictionary version</td>
<td>String</td>
<td>6.0</td>
</tr>
<tr>
<td>Bundle name</td>
<td>String</td>
<td>${PRODUCT_NAME}</td>
</tr>
<tr>
<td>Bundle OS Type code</td>
<td>String</td>
<td>XPC!</td>
</tr>
<tr>
<td>Bundle versions string, short</td>
<td>String</td>
<td>1.0</td>
</tr>
<tr>
<td>Bundle creator OS Type code</td>
<td>String</td>
<td>?? ??</td>
</tr>
<tr>
<td>Bundle version</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td><strong>NSExtension</strong></td>
<td>Dictionary</td>
<td>(3 items)</td>
</tr>
<tr>
<td><strong>NSExtensionAttributes</strong></td>
<td>Dictionary</td>
<td>(3 items)</td>
</tr>
<tr>
<td>UNNotificationExtensionInitialContentSizeRatio</td>
<td>Number</td>
<td>0.5</td>
</tr>
<tr>
<td>UNNotificationExtensionDefaultContentHidden</td>
<td>Boolean</td>
<td>YES</td>
</tr>
<tr>
<td>UNNotificationExtensionCategory</td>
<td>String</td>
<td>event-invite</td>
</tr>
<tr>
<td>NSEXTensionMainStoryboard</td>
<td>String</td>
<td>MainInterface</td>
</tr>
<tr>
<td>NSEXTensionPointIdentifier</td>
<td>String</td>
<td>com.apple.usernotifications.content-extension</td>
</tr>
</tbody>
</table>

Extension Attributes:
- **UNNotificationExtensionInitialContentSizeRatio**: 0.5
- **UNNotificationExtensionDefaultContentHidden**: YES
- **UNNotificationExtensionCategory**: event-invite
The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
Custom Notification UI

The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can't believe who's playing this year!

The Bash
Thursday, June 16, 7pm

Have you heard the rumors? I can't believe who's playing this year!

Accept
Decline
Custom Notification UI

Presentation size

UNNotificationExtensionInitialContentSizeRatio
Custom Notification UI

Presentation size

UNNotificationExtensionInitialContentSizeRatio

Default content

UNNotificationExtensionDefaultContentHidden
Custom Notification UI

Presentation size

UNNotificationExtensionInitialContentSizeRatio

Default content

UNNotificationExtensionDefaultContentHidden
Media Attachments
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventImage: UIImageView!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content

        if let attachment = content.attachments.first {
            if attachment.url.startAccessingSecurityScopedResource() {
                eventImage.image = UIImage(contentsOfFile: attachment.url.path!)
            }
            attachment.url.stopAccessingSecurityScopedResource()
        }
    }
}
// Notification Content Extension Attachments

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventImage: UIImageView!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content
        if let attachment = content.attachments.first {
            if attachment.url.startAccessingSecurityScopedResource() {
                eventImage.image = UIImage(contentsOfFile: attachment.url.path!)
                attachment.url.stopAccessingSecurityScopedResource()
            }
        }
    }
}
// Notification Content Extension Attachments

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventImage: UIImageView!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content

        if let attachment = content.attachments.first {
            if attachment.url.startAccessingSecurityScopedResource() {
                eventImage.image = UIImage(contentsOfFile: attachment.url.path!)
                attachment.url.stopAccessingSecurityScopedResource()
            }
        }
    }
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    @IBOutlet var eventImage: UIImageView!

    func didReceive(_ notification: UNNotification) {
        let content = notification.request.content

        if let attachment = content.attachments.first {
            if.attachment.url.startAccessingSecurityScopedResource() {
                eventImage.image = UIImage(contentsOfFile: attachment.url.path!)
                attachment.url.stopAccessingSecurityScopedResource()
            }
        }
    }
}
The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
Actions
Default Action Handling
Default Action Handling

Delivered to the app

Notification gets dismissed immediately
Interceptoring Action Response
Intercepting Action Response

Delivered to the extension

Can delay dismissal
// Intercepting notification action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {
            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }
            done(.dismiss)
        }
    }
}
// Intercepting notification action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {
            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }

            done(.dismiss)
        }
    }
}
// Intercepting notification action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse,
                 completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {

            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }

            done(.dismiss)
        }
    }

}
// Intercepting notification action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse,
    completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {

            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }

            done(.dismiss)
        }
    }
}

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse,
    completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {
            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }

            done(.dismiss)
        }
    }
}
// Intercepting notification action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse,
        completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        server.postEventResponse(response.actionIdentifier) {
            if response.actionIdentifier == "accept" {
                eventResponse.text = "Going!"
                eventResponse.textColor = UIColor.green()
            } else if response.actionIdentifier == "decline" {
                eventResponse.text = "Not going :(
                eventResponse.textColor = UIColor.red()
            }

            done(.dismissAndForwardAction)
        }
    }
}
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
Text Input Action
private func makeEventExtensionCategory() -> UNNotificationCategory {

    let commentAction = UNTextInputNotificationAction(
        identifier: "comment",
        title: "Comment",
        options: [],
        textInputButtonTitle: "Send",
        textInputPlaceholder: "Type here…"
    )

    return UNNotificationCategory(identifier: "event-invite",
                                   actions: [ acceptAction, declineAction, commentAction ],
                                   minimalActions: [ acceptAction, declineAction ],
                                   intentIdentifiers: [],
                                   options: [])
}

private func makeEventExtensionCategory() -> UNNotificationCategory {

    let commentAction = UNTextInputNotificationAction(
        identifier: "comment",
        title: "Comment",
        options: [],
        textInputButtonTitle: "Send",
        textInputPlaceholder: "Type here...")

    return UNNotificationCategory(identifier: "event-invite",
                                   actions: [acceptAction, declineAction, commentAction],
                                   minimalActions: [acceptAction, declineAction],
                                   intentIdentifiers: [],
                                   options: [])
}
private func makeEventExtensionCategory() -> UNNotificationCategory {

    let commentAction = UNTextInputNotificationAction(
        identifier: "comment",
        title: "Comment",
        options: [],
        textInputButtonTitle: "Send",
        textInputPlaceholder: "Type here..."
    )

    return UNNotificationCategory(identifier: "event-invite",
                                   actions: [ acceptAction, declineAction, commentAction ],
                                   minimalActions: [ acceptAction, declineAction ],
                                   intentIdentifiers: [],
                                   options: [])
}
private func makeEventExtensionCategory() -> UNNotificationCategory {

    let commentAction = UNTextInputNotificationAction(
        identifier: "comment",
        title: "Comment",
        options: [],
        textInputButtonTitle: "Send",
        textInputPlaceholder: "Type here...")

    return UNNotificationCategory(identifier: "event-invite",
                                   actions: [ acceptAction, declineAction, commentAction ],
                                   minimalActions: [ acceptAction, declineAction ],
                                   intentIdentifiers: [],
                                   options: [])
}

The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        if let textResponse = response as? UNTextInputNotificationResponse {
            server.send(textResponse.userText) {
                done(.dismiss)
            }
        }
    }
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        if let textResponse = response as? UNTextInputNotificationResponse {
            server.send(textResponse.userText) {
                done(.dismiss)
            }
        }
    }
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {
        if let textResponse = response as? UNTextInputNotificationResponse {
            server.send(textResponse.userText) {
                done(.dismiss)
            }
        }
    }
}
// Text input action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        if let textResponse = response as? UNTextInputNotificationResponse {
            server.send(textResponse.userText) {
                done(.dismiss)
            }
        }
    }
}
/ Text input action response

class NotificationViewController: UIViewController, UNNotificationContentExtension {

    func didReceive(_ response: UNNotificationResponse,
    completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {

        if let textResponse = response as? UNTextInputNotificationResponse {
            server.send(textResponse.userText) {
                done(.dismiss)
            }
        }
    }
}
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
The Bash
Thursday, June 16, 7pm
Bill Graham Civic

Have you heard the rumors? I can’t believe who’s playing this year!
class NotificationViewController: UIViewController, UNNotificationContentExtension {
    override func canBecomeFirstResponder() -> Bool {
        return true
    }

    override var inputAccessoryView: UIView { get {
        return inputView
    }
}

func didReceive(_ response: UNNotificationResponse,
completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {
    if response.actionIdentifier == "comment" {
        becomeFirstResponder()
        textField.becomeFirstResponder()
    }
}
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

override func canBecomeFirstResponder() -> Bool {
    return true
}

override var inputAccessoryView: UIView { get {
    return inputView
}
}

func didReceive(_ response: UNNotificationResponse,
completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {
    if response.actionIdentifier == "comment" {
        becomeFirstResponder()
        textField.becomeFirstResponder()
    }
}
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

override func canBecomeFirstResponder() -> Bool {
    return true
}

override var inputAccessoryView: UIView { get {
    return inputView
}
}

func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {
    if response.actionIdentifier == "comment" {
        becomeFirstResponder()
        textField.becomeFirstResponder()
    }
}
}
class NotificationViewController: UIViewController, UNNotificationContentExtension {

    override func canBecomeFirstResponder() -> Bool {
        return true
    }

    override var inputAccessoryView: UIView {
        return inputView
    }

    func didReceive(_ response: UNNotificationResponse, completionHandler done: (UNNotificationContentExtensionResponseOption) -> Void) {
        if response.actionIdentifier == "comment" {
            becomeFirstResponder()
            textField.becomeFirstResponder()
        }
    }
}
The Bash
Thursday, June 16, 7pm
Have you heard the rumors? I can’t believe who’s playing this year!
Summary

Attachments and custom UI
Attachments with service extension
Custom UI with content extension
• Media attachments
• User interaction
More Information

https://developer.apple.com/wwdc16/708
<table>
<thead>
<tr>
<th>Related Sessions</th>
<th>Pacific Heights</th>
<th>Wednesday 9:00AM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to Notifications</td>
<td>Pacific Heights</td>
<td>Wednesday 9:00AM</td>
</tr>
<tr>
<td>Lab Name</td>
<td>Location</td>
<td>Time</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-----------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Notifications Lab</td>
<td>Frameworks Lab C</td>
<td>Wednesday 11:00AM</td>
</tr>
<tr>
<td>Notifications Lab</td>
<td>Graphics, Games, and Media Lab B</td>
<td>Friday 9:00AM</td>
</tr>
</tbody>
</table>