Using Local and Push Notifications

On iOS and Mac OS X

Session 517

Darryl Bleau
APNs Engineering Manager

Push Notifications

Introduction and Review

Darryl Bleau
APNs Engineering Manager

Agenda

Topics we'll be covering

- Introduction to notifications
- Push architecture
- Local and push notifications on iOS
- Push notifications on Mac OS X
- Best practices

Introduction to Notifications

Questions and answers

- What is a notification?
  - User visible information reflecting some event
- Why should I use notifications?
  - Ensure time-sensitive delivery even when your app isn't running
- How does push compare to poll?
  - Pushes are server-driven and immediate, polls are app-driven and latent

Push Notifications

Overview

- Send notifications directly to target device
- Intended as a ‘tap on the shoulder’
- Should reflect server-side state

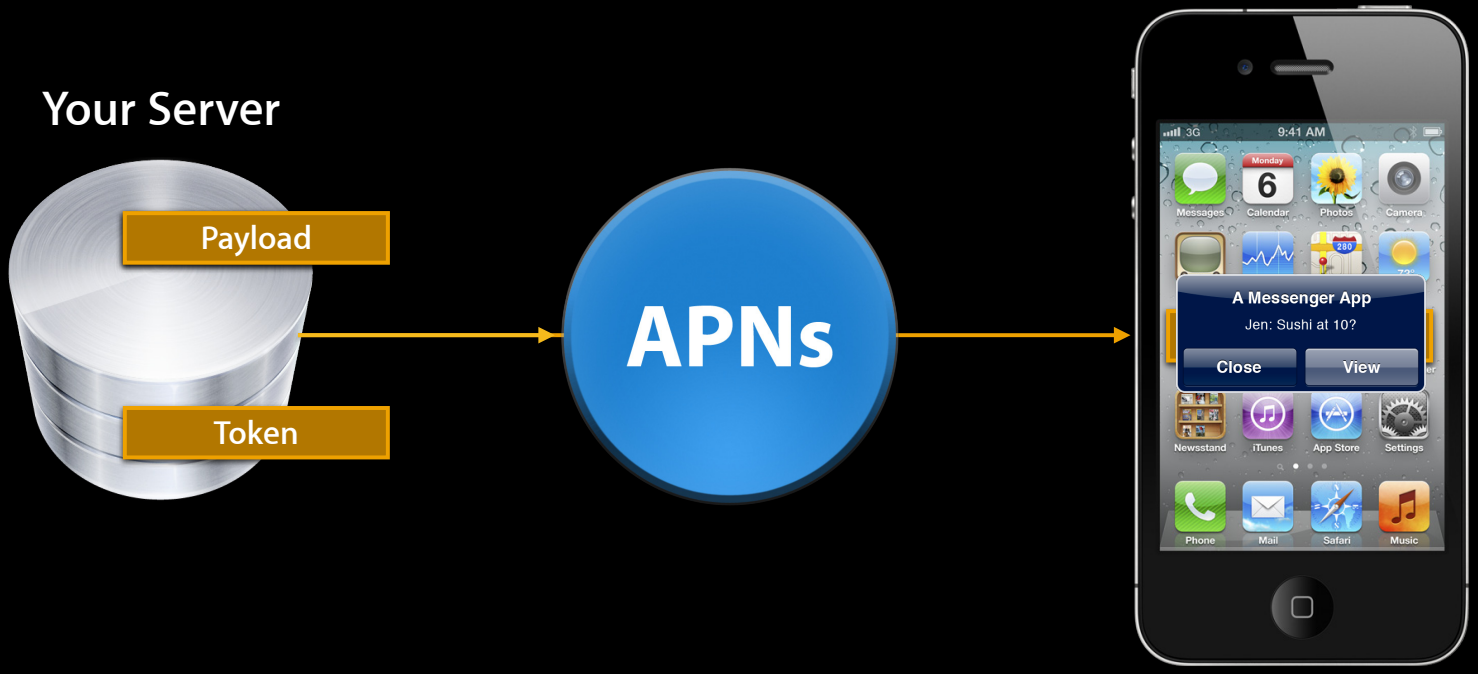
Push Notifications

Store and forward

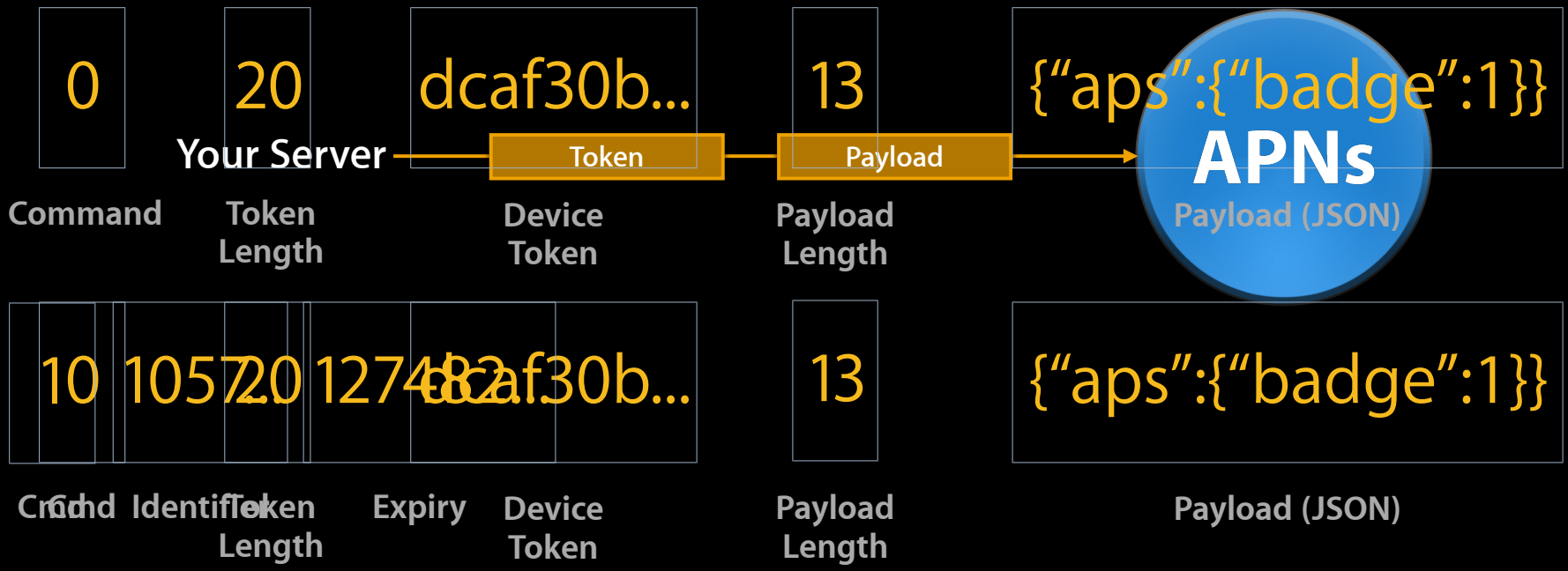
- Notifications sent to offline devices are stored for future delivery
- Reconnecting devices receive the latest notification per app
- Notifications can be sent with an expiry time

Expiry only affects storage; notifications to online devices are delivered regardless of expiry time

Push Notification Service Architecture



Binary Protocols



Sending Notifications

Message payload

- Strict RFC 4627 JSON
- Human readable
- Compact
- Easily mapped to NSDictionary
- 256 byte maximum
- Keys outside “aps” are available for your own use

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?",
        "badge" : 1,
        "sound" : "Jingle.aiff"
    },
    "acme1" : "conversation9964"
}
```

Sizing Considerations

Whitespace-stripped JSON is a good idea

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?",
        "badge" : 1,
        "sound" : "Jingle.aiff"
    },
    "acme1" : "conversation9964"
}
```

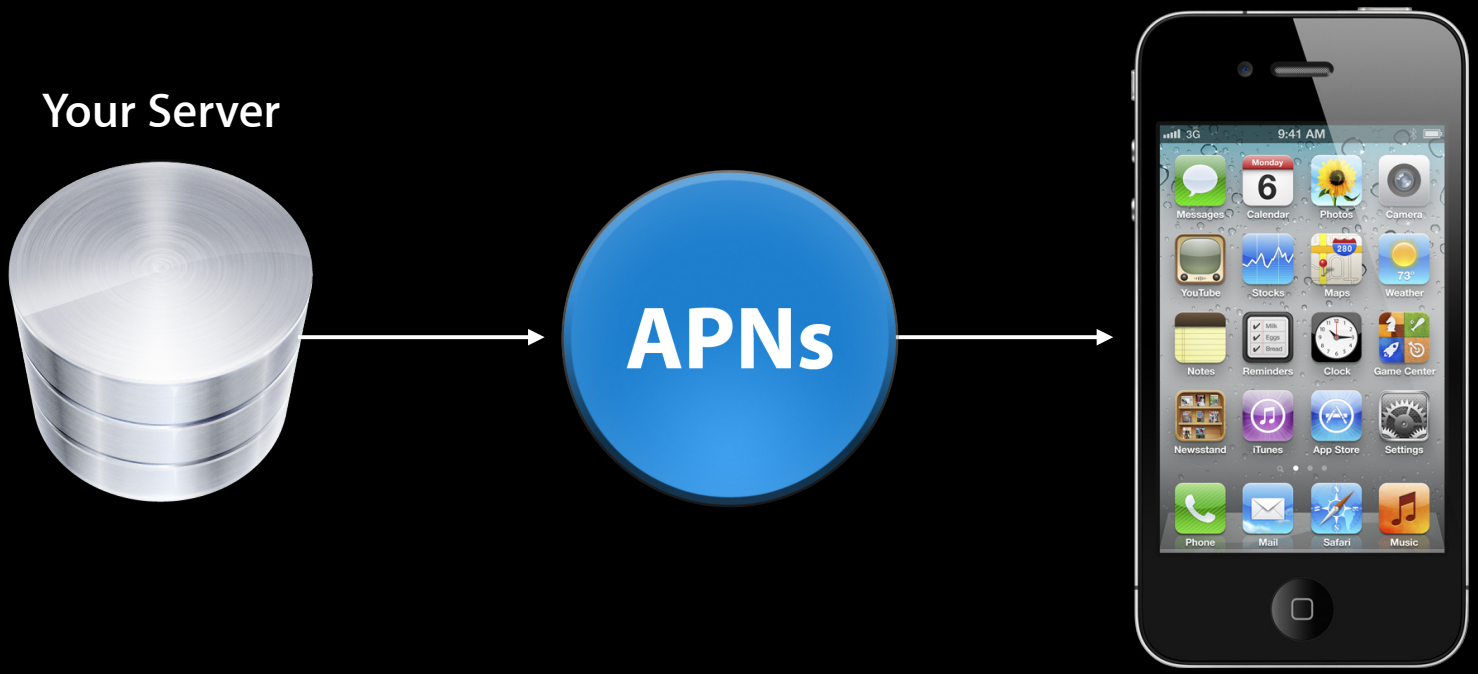
```
{"aps":{"alert":"Jen: Sushi at 10?","badge":1,
"sound":"Jingle.aiff"},"acme1":"conversation9964"}
```

150 bytes

96 bytes

Push Notifications

Over to the client side



Notifications on iOS

Push and local notifications

James Callender
iOS Software Engineer

Push Notifications

Using Push Notifications

When to use them

- Social networking
- Current events
- Games

Using Push Notifications

When not to use them

- Delivering critical application information
  - Push may not be available
  - Instead, pull data from server

Registering for Notifications

Application launch

- UIApplication to register
  - Pass the types you want to receive

```
-(void)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)options {
   UIRemoteNotificationType myTypes = UIRemoteNotificationTypeSound |
                                      UIRemoteNotificationTypeBadge;
   [application registerForRemoteNotificationTypes:myTypes];
}
```

Registering for Notifications

Delegate callbacks

- Successful registration

```
- (void)application:(UIApplication *)application
    didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)token
```

  - Phone home with your token

- Failure to register

```
- (void)application:(UIApplication *)application
    didFailToRegisterForRemoteNotificationsWithError:(NSError *)error
```

  - Check your provisioning profile for entitlements
  - Push not supported in the simulator

Registering for Notifications

The device token

```
96385da767191121a851963983fdac9bbdf74dcf6219ae14ed8d082288aaebaf
```

- Uniquely identifies the device
  - Separate from the UDID
- May change
  - Call registration API on every launch
  - Don’t depend on a cached copy

Creating Push Notifications

Message payload

- aps dictionary reserved for the sound, badge, or alert keys
  - All keys optional

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?",
        "badge" : 1,
        "sound" : "Jingle.aiff"
    },
    "acme1" : "conversation9964"
}
```

- Rest of payload is for your app

Creating Push Notifications

Badges

- Specified using a positive integer

```
{
    "aps" : {
        "badge" : 1
    }
}
```

- Omitting will clear the badge

```
{
    "aps" : { }
}
```

Creating Push Notifications

Sounds

- Filename in application bundle

```
{
    "aps" : {
        "sound" : "Jingle.aiff"
    }
}
```

- Or "default" for the default notification noise

```
{
    "aps" : {
        "sound" : "default"
    }
}
```

- Vibration is automatic

Creating Push Notifications

Alerts

- Simplest form is just a string value

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?"
    }
}
```

Creating Push Notifications

Customizing alerts

- Replace the alert string with a dictionary

```
{
    "aps" : {
        "alert" : {
            "loc-key" : "MSG_POSTED",
            "loc-args" : [ "Jen", "iPhone" ],
            “action-loc-key” : “VIEW”
        }
    }
}
```


```
Spanish.lproj/Localizable.strings
“MSG_POSTED” = “%@ acaba de fijar un mensaje para usted sobre %@”;
“VIEW” = “Ver”;
```

Push Notification Delivery

Getting the payload

- If your app is running, you'll only get

```
- (void)application:(UIApplication *)application
    didReceiveRemoteNotification:(NSDictionary *)userInfo
```

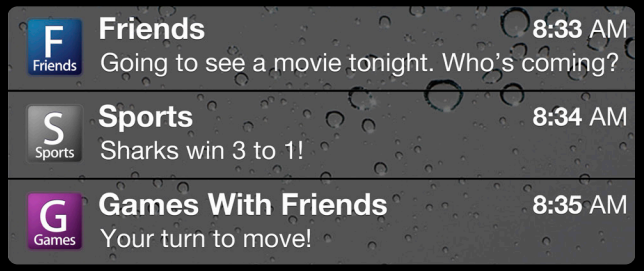
- If not, tapping View will launch your app with:

```
- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    NSDictionary *userInfo = [launchOptions objectForKey:
        UIApplicationLaunchOptionsRemoteNotificationKey];
    ...
}
```

Local Notifications

Local Notifications vs. Push Notifications

How they are similar



Appearance: badges, alerts, and sounds

iOS acts on behalf of your app

Push Notifications vs. Local Notifications

How they are different

Push Notifications	Local Notifications
Originate from server	Originate from your app
Single shot	Scheduled, repeatable

Using Local Notifications

When to use them

- Alarm Clock
- Reminder
- Location

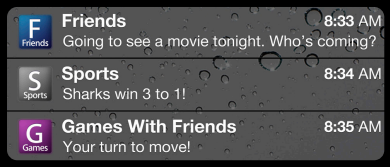
Using Local Notifications

When not to use them

- Alerts and errors
  - UIAlertView
- Calendar events
  - EventKit if possible

Creating Local Notifications

UILocalNotification Overview



- Badge

```
NSInteger applicationIconBadgeNumber
```

- Alerts

```
NSString *alertBody
BOOL hasAction
NSString *alertAction
NSString *alertLaunchImage
```

- Sound

```
NSString *soundName
```

- Scheduling

```
NSDate *fireDate
NSTimeZone *timeZone
```

- Repeating

```
NSCalendarUnit repeatInterval
NSCalendar *repeatCalendar
```

- Metadata

```
NSDictionary *userInfo
```

Creating Local Notifications

Badges

- Use the applicationIconBadgeNumber property

```
UILocalNotification *note =
  [[UILocalNotification alloc] init];
note.applicationIconBadgeNumber = 3;
```

- Setting to zero will **not** clear the badge
  - Instead use UIApplication:

```
UIApplication *application =
  [UIApplication sharedApplication];
application.applicationIconBadgeNumber = 0;
```

Creating Local Notifications

Alerts

- Set the alertBody property

```
UILocalNotification *note =
  [[UILocalNotification alloc] init];
note.alertBody = @"Baseball game starting now";
```

- Optionally set the alertAction property

```
note.hasAction = YES;
note.alertAction = @"Watch";
```

Creating Local Notifications

Localizing your alerts

- Setting the localized key

```
UILocalNotification *note =
    [[UILocalNotification alloc] init];
note.alertBody = @"MSG_POSTED";
note.hasAction = YES;
note.alertAction = @"SHOW_KEY";

French.lproj/Localizable.strings
"MSG_POSTED" = "Le match de baseball commence dans 5 minutes";
"SHOW_KEY" = "Regarder";
```

Creating Local Notifications

Alert Launch Images

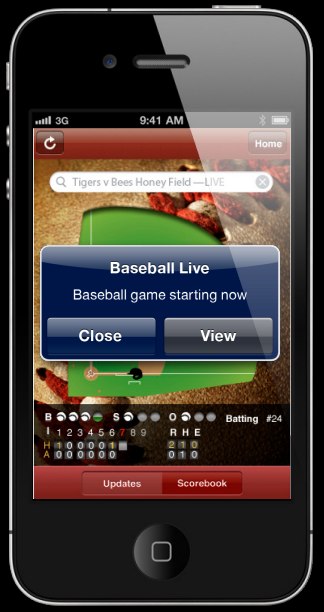
- `alertLaunchImage` will change your app's launch image

Creating Local Notifications

Alert Launch Images

- `alertLaunchImage` will change your app's launch image

```
note.alertLaunchImage = @"Default-watch.png";
```



Creating Local Notifications

Sounds

- Use the soundName property

```
UILocalNotification *note =
  [[UILocalNotification alloc] init];
note.soundName = @"Squawk.aiff";
```

- UILocalNotificationDefaultSoundName plays a default sound

```
UILocalNotification *note =
  [[UILocalNotification alloc] init];
note.soundName = UILocalNotificationDefaultSoundName;
```

Creating Local Notifications

UILocalNotification Overview

- Badge

```
NSInteger applicationIconBadgeNumber
```

- Alerts

```
NSString *alertBody
BOOL hasAction
NSString *alertAction
NSString *alertLaunchImage
```

- Sound

```
NSString *soundName
```

- Scheduling

```
NSDate *fireDate
NSTimeZone *timeZone
```

- Repeating

```
NSCalendarUnit repeatInterval
NSCalendar *repeatCalendar
```

- Metadata

```
NSDictionary *userInfo
```

A Word About Dates

Universal Time vs. Wall Time

- "Universal" time
  - Lonely NSDate
  - Conference call
  - Stock market close
- "Wall" time
  - NSDate + NSTimeZone
  - 9:00AM alarm
  - Television show (Thursdays at 8:00PM EST)

Creating Local Notifications

Scheduling

- Use the `fireDate` and `timeZone` properties

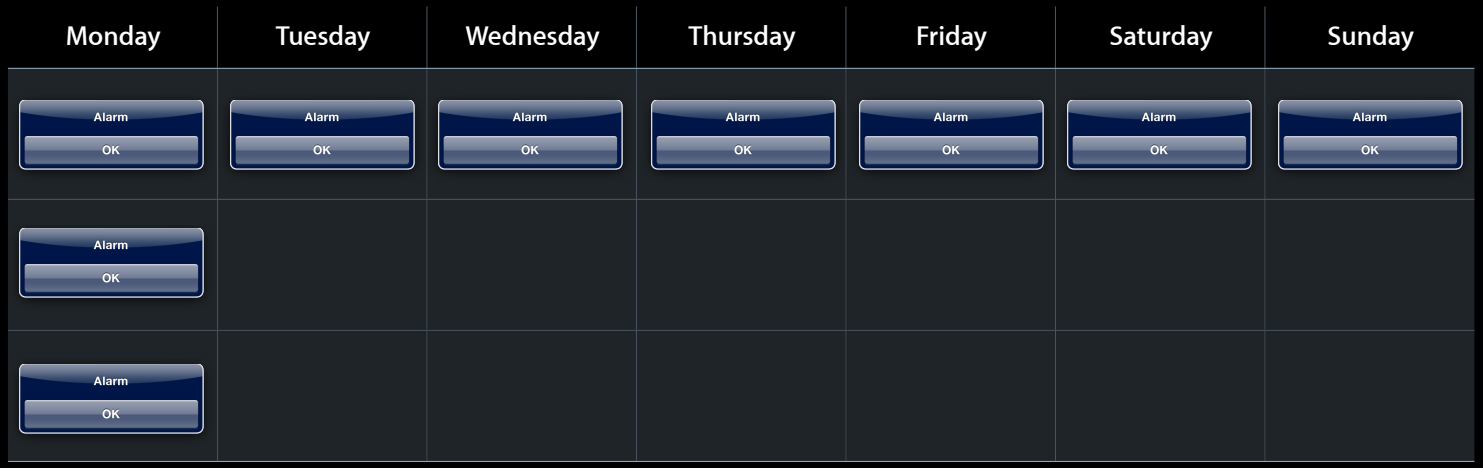
```
UILocalNotification *note =
  [[UILocalNotification alloc] init];
NSCalendar *calendar = [NSCalendar currentCalendar];
NSDateComponents *dateComps =
  [[NSDateComponents alloc] init];
[dateComps setDay:10];
[dateComps setMonth:6];
[dateComps setYear:2010];
[dateComps setHour:14];
note.fireDate = [calendar dateFromComponents:dateComps];
note.timeZone = [calendar timeZone];
```

Creating Local Notifications

Scheduling repeating notifications

- Use repeatInterval and repeatCalendar

```
note.repeatInterval = NSWeekCalendarUnit;
note.repeatCalendar = [NSCalendar currentCalendar];
```



Creating Local Notifications

UILocalNotification Overview

- Badge

```
NSInteger applicationIconBadgeNumber
```

- Alerts

```
NSString *alertBody
BOOL hasAction
NSString *alertAction
NSString *alertLaunchImage
```

- Sound

```
NSString *soundName
```

- Scheduling

```
NSDate *fireDate
NSTimeZone *timeZone
```

- Repeating

```
NSCalendarUnit repeatInterval
NSCalendar *repeatCalendar
```

- Metadata

```
NSDictionary *userInfo
```

Creating Local Notifications

Scheduling

- Schedule and canceling with UIApplication

```
- (void)scheduleLocalNotification:
    (UILocalNotification *)notification;
- (void)cancelLocalNotification:
    (UILocalNotification *)notification;
```

- Scheduling for background apps

```
- (void)presentLocalNotificationNow:
    (UILocalNotification *)notification;
```

Local Notification Delivery

Getting the notification

- If your app is running, you'll only get:

```
- (void)application:(UIApplication *)application
    didReceiveLocalNotification:(UILocalNotification *)notification
```

- If not, tapping View will launch your app with:

```
- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    UILocalNotification *note = [launchOptions objectForKey:
        UIApplicationLaunchOptionsLocalNotificationKey];
    ...
}
```

Push Notifications on Mac OS X

Jason Thorpe
Mac OS X Software Engineering Manager

Push Notifications on Mac OS X

How they are the same

- AppKit API is the same as UIKit API
- Notifications are delivered to running applications via your NSApplication delegate
- Icons are badged in Dock and LaunchPad for non-running applications

Push Notifications on Mac OS X

How they are different

- Badges only; no alerts or sounds
- Application cannot be launched in response to a Push notification
- No systemwide preferences for Push Notifications
- No iOS-style local notifications

Push Notifications on Mac OS X

Server-side impact

- Different Push provider certificates for iOS and Mac versions of an application
  - Clients must identify themselves!

Push Notifications on Mac OS X

Server-side impact

- Same Push payload can be sent to iOS and Mac applications
  - You only have to construct it once!

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?",
        "badge" : 1,
        "sound" : "Jingle.aiff"
    },
    "acme1" : "conversation9964"
}
```

Push Notifications on Mac OS X

Developer considerations

- Applications require a provisioning profile from the Apple Developer Certificate Utility

Push Notifications on Mac OS X

Developer considerations

- Only applications distributed through the Mac App Store may use Push notifications

Registering for Notifications

Application launch

- NSApplication to register
  - Pass the types you want to receive

```
-(void)applicationDidFinishLaunching:(NSNotification *)aNotification {
    NSApplication *application = [aNotification object];
    NSRemoteNotificationType myTypes = NSRemoteNotificationTypeBadges;
    [application registerForRemoteNotificationTypes:myTypes];
}
```

Registering for Notifications

Delegate callbacks

- Successful registration

```
- (void)application:(NSApplication *)application
    didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)token
```

  - Phone home with your token
    - Make sure to distinguish between iOS and Mac versions of your app!
- Failure to register

```
- (void)application:(NSApplication *)application
    didFailToRegisterForRemoteNotificationsWithError:(NSError *)error
```

  - Check your provisioning profile for entitlements

Push Notification Delivery

Getting the payload

- Implement this in your NSApplication delegate

```
- (void)application:(NSApplication *)application
    didReceiveRemoteNotification:(NSDictionary *)userInfo
```

- Payload is contained in `userInfo`

```
{
    "aps" : {
        "alert" : "Jen: Sushi at 10?",
        "badge" : 1,
        "sound" : "Jingle.aiff"
    },
    "acme1" : "conversation9964"
}
```

Demo

Darrin Jewell
Mac OS X Software Engineer

Push Notifications

Best Practices

Darryl Bleau
APNs Engineering Manager

Best Practices

- Pull data while your app is running
  - Push service may be turned off
  - Apps should be fully functional without Push
- Upload the device token to your server often
  - Users can sync and restore your application data without your server's knowledge
- Don't annoy your users with alerts
  - Send notifications your user cares about
  - If you don't need them, don't use them

Badges

We do need them

- Primary notification type
- For actionable information
- Keep a server-side count
- Test your logic
  - Count can change while inactive
  - Notifications can arrive while active
  - Ensure accurate information

More Information

Bill Dudney
Application Frameworks Evangelist
Bill Dudney <dudney@apple.com

Apple Developer Forums
http://devforums.apple.com

Sample Application: PushyMac
http://developer.apple.com/library/mac/#samplecode/PushyMac/Introduction/Intro.html

Labs

Local and Push Notifications Lab	Internet and Web Lab A Friday 9:00–11:15AM


WWDC2011