CS E-76: Building Mobile Applications

TF Seminar: Code Signing, Installing Your Apps on Hardware & Push Notifications

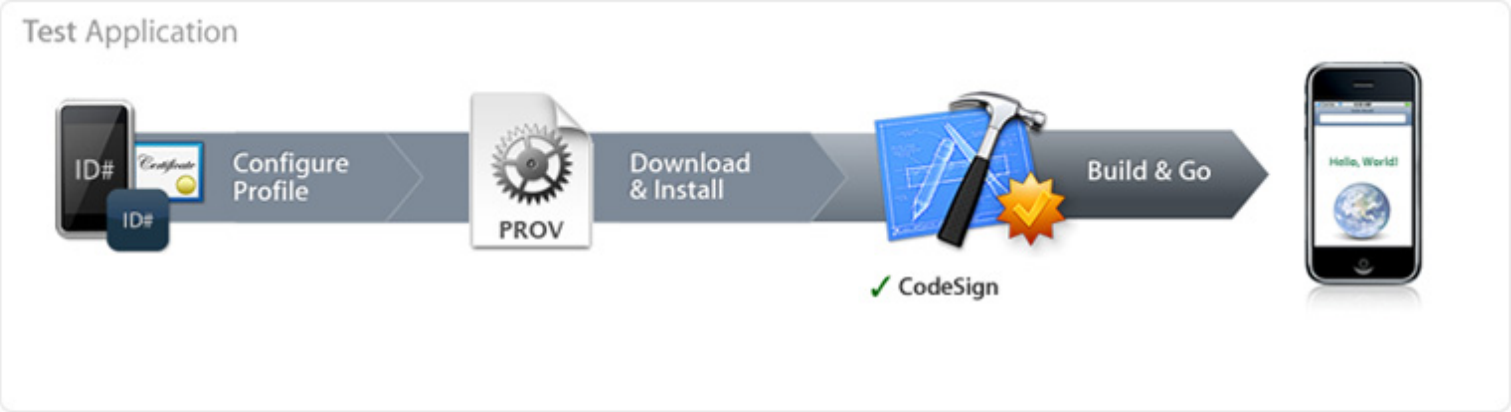
Bob Kinney

Agenda

- Code Signing
- Installing Applications on Hardware
  - With XCode for development
  - Distribution methods
- Notifications
  - Local Notifications
  - Push (Remote) Notifications
- Wrap up

Code Signing

- Used to verify "valid" code is running on device
  - Apple is the ultimate arbiter of what is considered valid (walled garden effect)
- Signature verification requires three main components



https://developer.apple.com/ios/manage/provisioningprofiles/howto.action

Code Signing (cont.)

- Developer Identity
  - Developer creates key/certificate request, uploads to Apple provisioning portal
  - Apple generates certificate
- Device ID
  - Each iOS device has a Universal Device ID (UDID)
  - UDIDs are added to Apple provisioning portal
- Application ID
  - Set in XCode/added to Apple provisioning portal
    - ID stored in portal may be a wildcard (put not for push)

Provisioning Profiles

- Provisioning Profiles are a combination of the 3 main components
  - Created through provisioning portal
  - Multiple UDIDs and Certicates can be in profile, only one App ID (or wildcard)
- Developers/testers install profile on device
- Code will run if
  - UDID of device is included in profile
  - App ID matches ID (or wildcard) included in profile
  - Developer signed with certificate included in profile

Provisioning Profiles (cont.)

2 types of profiles

1. Development profile
   - Used by developers during testing
2. Distribution profile
   - Used for Ad-Hoc distribution and submission to Apple

Development profiles can be created by any admin, distribution profiles can only be created by *program coordinator*.

Installing Your Apps on Hardware

What you will need:

- A valid iOS device
  - David will be sending instructions on how to have your UDID to the Harvard program
- A valid developer identity certificate
  - Instructions will include how to get this
- A provisioning profile that includes the above
  - Instructions will include how to get this

DEMO

Notifcations

Two types of notifications

1. Local
   - Generated by application locally
   - Can be timed or initiated during background task
2. Remote (Push)
   - Generated by *Push Notification Provider*
   - Routed through Apple, require a network connection

Reasons to Use Notification

- Local Notification
  - Timed alerts (calendar, alarm)
- Push Notification
  - Exchange of information (chat, mail, etc)
  - Alert user to presence of new information
  - Alert user to new version of software
  - Advertisements (yes, they can be used for evil)

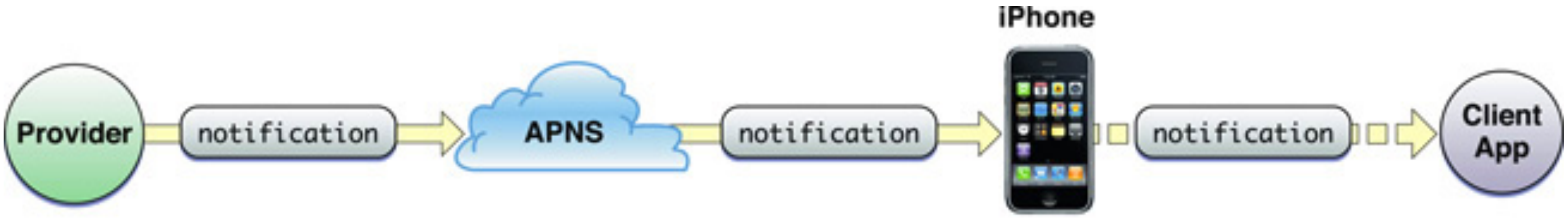
Local Notifications

- Create `UILocalNotification` object
- Can include *payload*
  - Badge, message, and/or sound
    - non-default sounds must be included in App
  - Can choose to include an *action* button
    - More on handling notifications in a bit

DEMO

Remote (Push) Notifications

- Generated by a Provider
  - API is well documented
  - Servers exist in a number of languages
    - http://code.google.com/p/ruby-apns-daemon/
    - http://code.google.com/p/php-apns/
  - Requires an additional certificate installed on provider
    - Tied to AppID



http://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/ApplePushService/ApplePushService.html

Push Notification Workflow

- App calls `registerForRemoteNotificationTypes:` to register device/app combination with APNS
- AppDelegate needs to handle two cases
  - `application:didRegisterForRemoteNotificationsWithDeviceToken:`
    - Registration succeeded. Token must be passed to provider
  - `application:didFailToRegisterForRemoteNotificationsWithError:`
    - Registration failed. Error should be handled
- Provider sends notifications to APNS with token and *payload*

```
"aps" : {
        "alert" : "You got your emails.",
        "badge" : 9,
        "sound" : "bingbong.aiff"
    },
    "acme1" : "bar",
    "acme2" : 42
}
```

DEMO

Handling Notifications

There are 3 contexts in which a notifications will be received:

1. Device is locked
2. Device is unlocked but application is not loaded
3. Device is unlocked and application is running.

Application is Not Running

If the push notification causes the user to load the application push data is handled via `application:didFinishLaunchingWithOptions:`

Application can then handle notification payload

- Does it need to load a particular mail message?
- Does it need to fetch information from a server?

Application is Running

- `UIApplicationDelegate` has 2 methods to handle notifations
  - `application:didReceiveRemoteNotification:`
  - `application:didReceiveLocalNotification:`
- No UIAlertView is generated by default
  - For local notification, full object is available
  - For remote notifications, only dictionary of additional data is available

DEMO

Don’t want to do it yourself?

- http://urbanairship.com/
  - Push notification provider, in-app purchase
- http://testflightapp.com
  - Make distributing your apps to testers easier

Further reading

- Apple documentation for notifications:
  http://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/

- Documentation for UILocalNotification
  http://developer.apple.com/library/ios/#documentation/iPhone/Reference/UILocalNotification_Class/Reference/Reference.html#//apple_ref/c/data/UILocalNotificationDefaultSoundName

- Google is your friend

CS E-76: Building Mobile Applications

Thanks for watching!