Adopting Multitasking on iPhone OS

Part 2

David Myszewski, Charles Srisuwananukorn
iPhone Performance

Introduction

- Multitasking does not mean applications run all the time
- Most applications only need fast application switching
- Some applications benefit from background execution

What You'll Learn

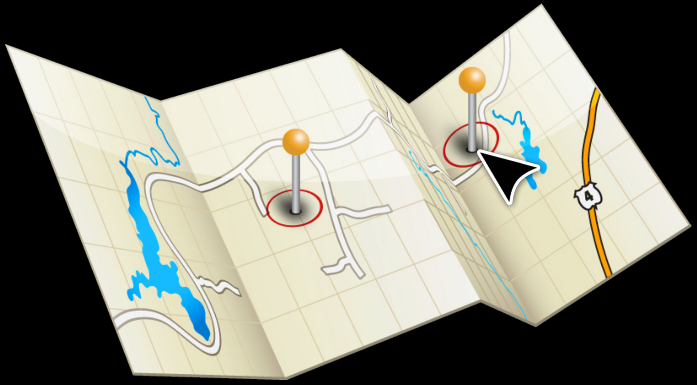
- Task completion
  - Extra time to complete a task
- Background audio
  - Play audible content to the user while in the background

What You'll Learn

- Task completion
  - Extra time to complete a task
- Background audio
  - Play audible content to the user while in the background

What You'll Learn

- Navigation
  - Keep users continuously informed of their location
- Location Tracking
  - Respond to location changes while in the background
- Voice over IP
  - Make and receive phone calls using an internet connection



What You'll Learn

- Navigation
  - Keep users continuously informed of their location
- Location Tracking
  - Respond to location changes while in the background
- Voice over IP
  - Make and receive phone calls using an internet connection

Task Completion

Task Completion
Examples

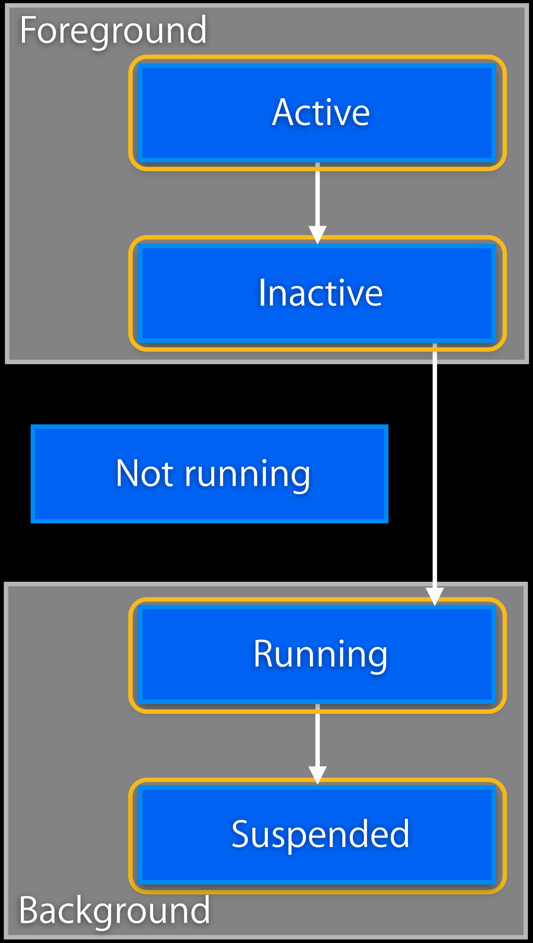
- Finishes a task in the background
  - Uploading photos or videos
  - Finishes applying an image filter
  - Finishes downloading a magazine

Task Completion

- Application can complete a task without remaining in the foreground
- User does not have to wait for the task to complete
- Task duration is limited to avoid excessive battery drain

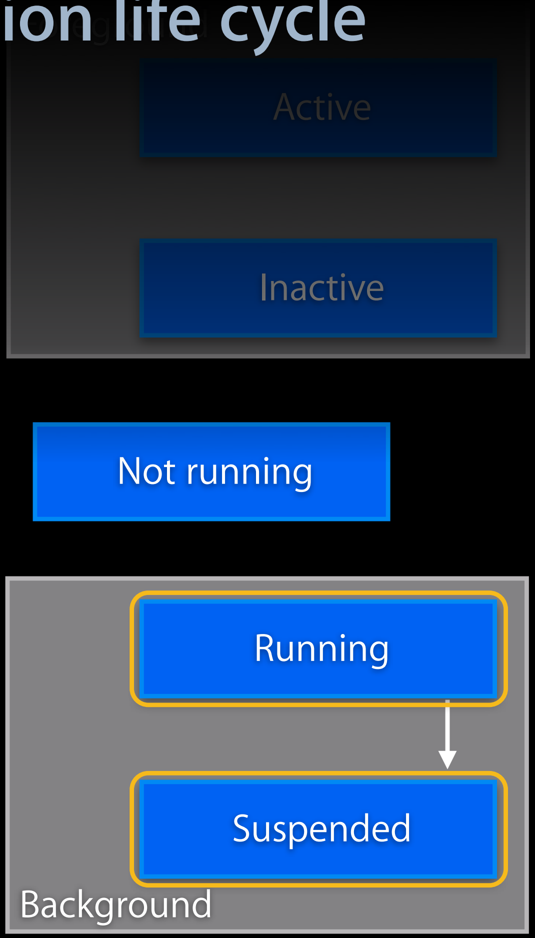
Task Completion

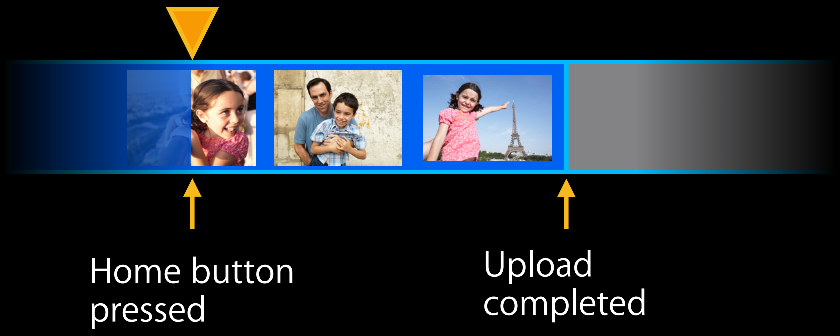
Application life cycle



Task Completion

Application life cycle





Task Completion API

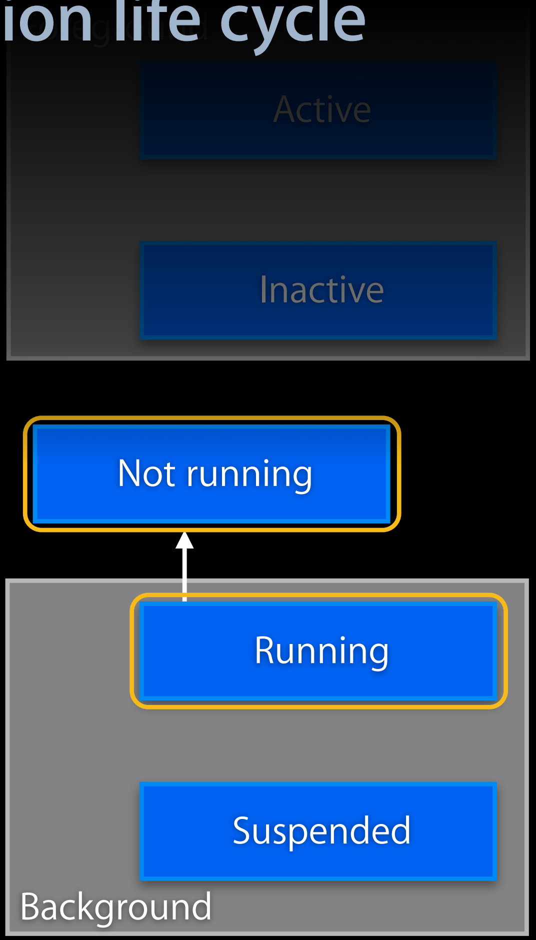
- Indicate start and end of the long running task

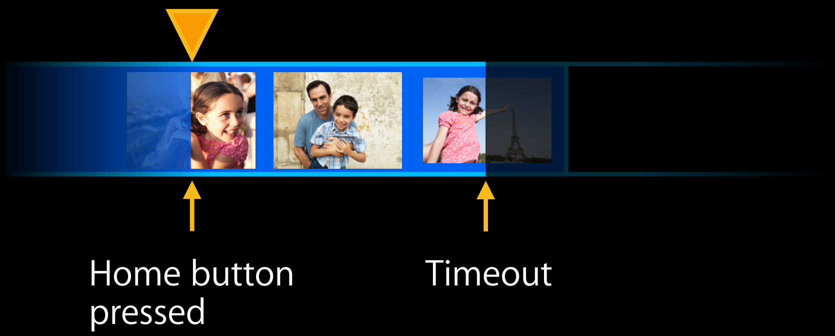
```
self.bgTask = [app beginBackgroundTaskWithExpirationHandler:^{ [self prepareForSuspend
```

```
[self uploadPhotos];
```

```
[app endBackgroundTask:self.bgTask];
```

Task Completion
Application life cycle





Task Completion API

Expiration handler

- Expiration handler called shortly before timeout
- May be called on a different thread
- Prepare for suspend
    - Save state
    - Reduce memory usage
    - Pause the long-running task
    - End the background task

Task Completion API

Expiration handler

```
self.bgTask = [app beginBackgroundTaskWithExpirationHandler:^{
    [self prepareForSuspend];
    [self pauseUpload];
}];

// returns after upload finishes or pauses
[self uploadPhotos];

[app endBackgroundTask:self.bgTask];
```

Task Completion

Best practices

- System prioritizes foreground activity
  - CPU
  - Network I/O
  - File I/O
- Some resources are off limits
  - GPU
  - Real-time threads

Task Completion

Best practices

- Minimize resource usage
  - CPU
  - Memory
- End background task as soon as possible
- Make background task resumable
- Avoid timeout by ending the background task in the expiration handler

Background Audio

Background Audio

Example application

- Plays audible content to the user
- Streams audio
- Continues playing in the background
- Integrates with remote controls

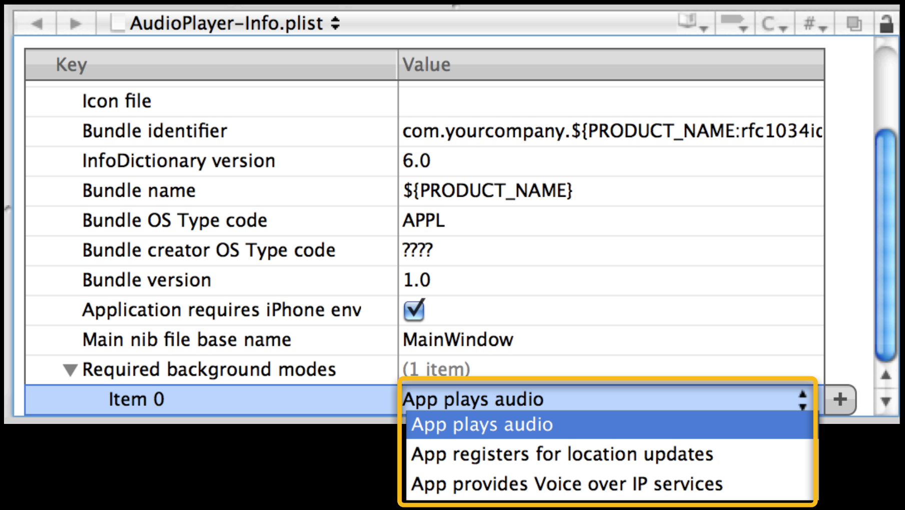
Background Audio

Audio system overview

- Audio system provides many services for both foreground and background audio
  - Prioritizing audio
  - Mixing and ducking
  - Headsets
  - External speakers
- We will focus on audio services for media player applications

Background Audio

Background modes


AudioPlayer-Info.plist

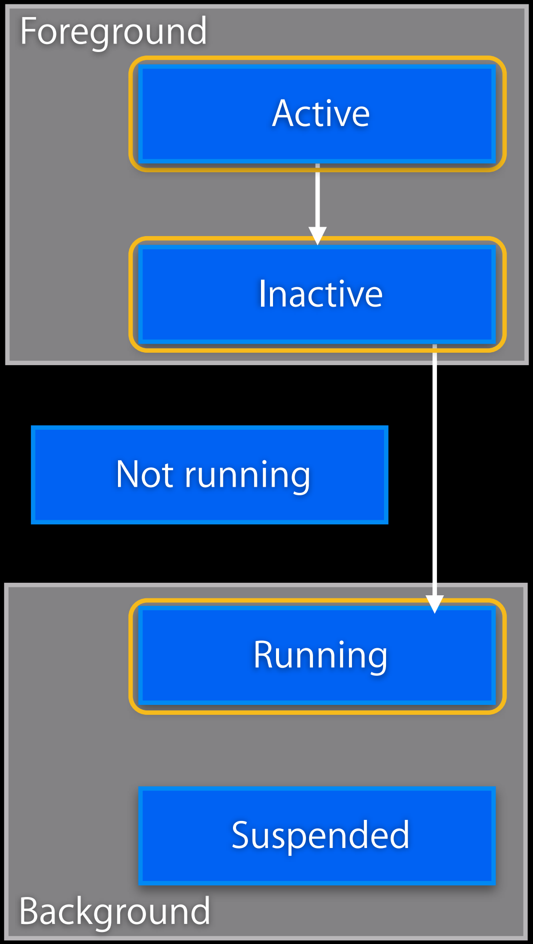
Key	Value
Icon file	
Bundle identifier	com.yourcompany.${PRODUCT_NAME:rfc1034ic
InfoDictionary version	6.0
Bundle name	${PRODUCT_NAME}
Bundle OS Type code	APPL
Bundle creator OS Type code	????
Bundle version	1.0
Application requires iPhone env	✓
Main nib file base name	MainWindow
▼ Required background modes	(1 item)
Item 0	App plays audio

App plays audio
App registers for location updates
App provides Voice over IP services

- App plays audio
  - Audio continues playing in the background

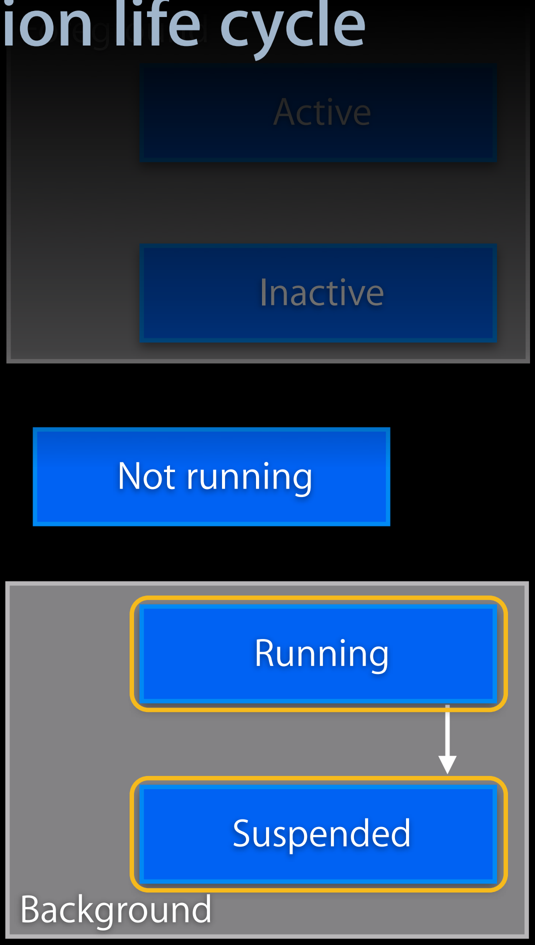
Background Audio

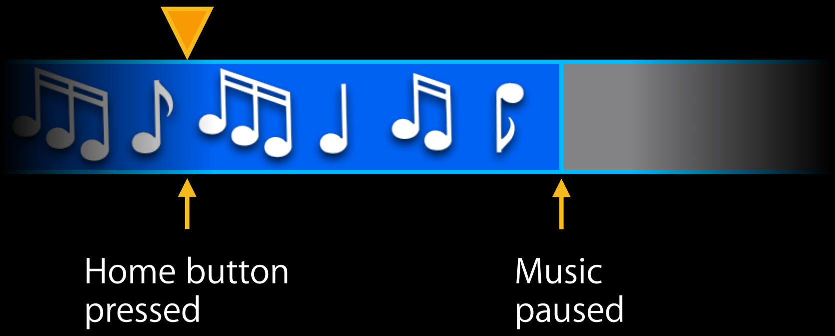
Application life cycle



Background Audio

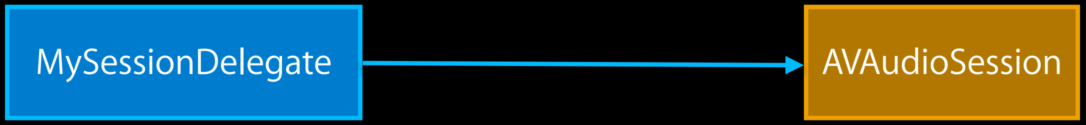
Application life cycle





Background Audio

Implementing audio behaviors



```
[session setCategory:AVAudioSessionCategoryPlayback error:&error]
```

```
-setCategory:error:
```

Background Audio

Implementing audio behaviors

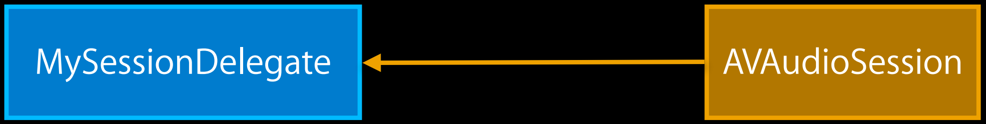
- Silences other audio
- Plays behind the lock screen
- Ignores ringer switch
- Continues playing in the background

Background Audio

Audio interruptions

- Handle audio interruptions
- During an interruption
  - Audio system silences interrupted application
  - Update UI appropriately
  - Resume after the interruption

Background Audio
Audio interruptions



`-beginInterruption`

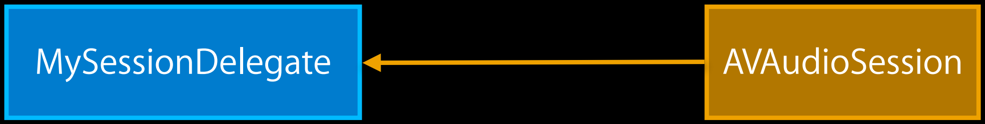
Background Audio

Audio interruptions

- In `beginInterruption`
  - Stop downloading the stream
  - Update UI
    - Play/Pause button
    - Play time
  - Stop visualizations

Background Audio

Audio interruptions



```
-endInterruptionWithFlags:
```

Background Audio

Audio interruptions

- In `endInterruptionWithFlags:`
  - Resume audio if `AVAudioSessionInterruptionFlags_ShouldResume` is set
  - Audio should resume for phone calls
  - Audio should not resume if iPod interrupts

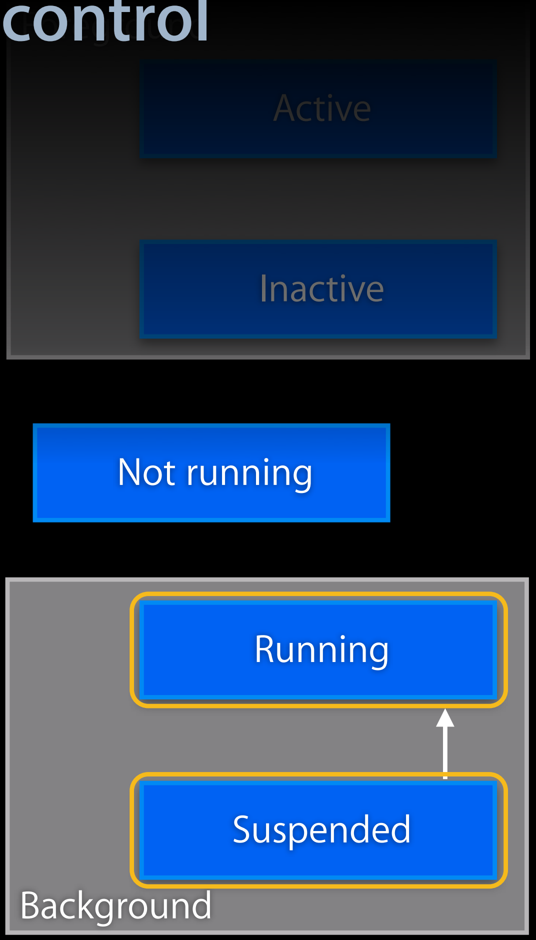
Background Audio

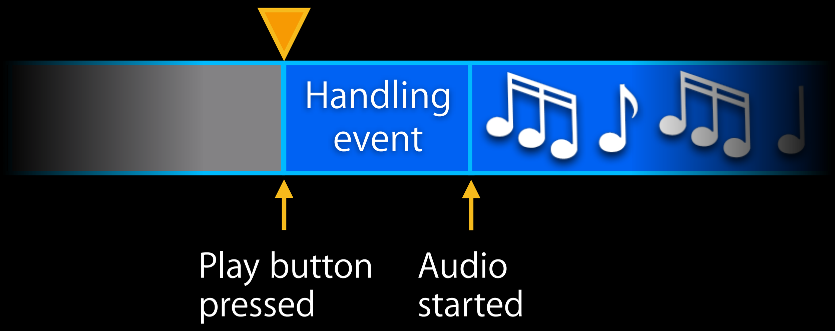
Remote control

- Users can interact with audio applications in the background through the remote controls
- The last app to play audio receives the events
- Events are routed through the responder chain
- Applications suspend until events are delivered

Background Audio

Remote control





Background Audio

Remote control

- Call `beginReceivingRemoteControlEvents` to indicate an interest in remote control events

```
[[UIApplication sharedApplication] beginReceivingRemoteControlEvents];
```

Background Audio

Remote control

```
- (void)remoteControlReceivedWithEvent:(UIEvent *)event {
    switch (event.subtype) {
        case UIEventSubtypeRemoteControlTogglePlayPause:
            [self togglePlayPause];
            break;

        case UIEventSubtypeRemoteControlNextTrack:
            [self playNextTrack];
            break;

        case UIEventSubtypeRemoteControlPreviousTrack:
            [self playPreviousTrack];
            break;

        // ...
    }
}
```

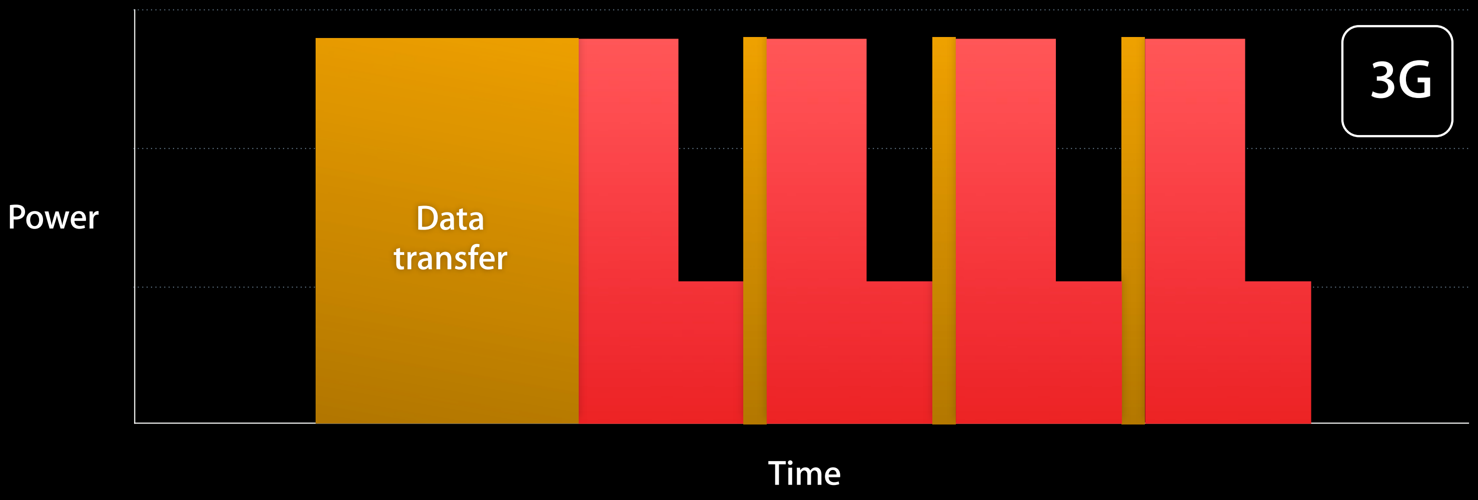
Background Audio
Best practices

- System ensures that background audio plays smoothly
  - Network I/O
  - File I/O
  - CPU
- Minimize CPU and resource usage
- Avoid real-time threads unless necessary

Background Audio

Best practices

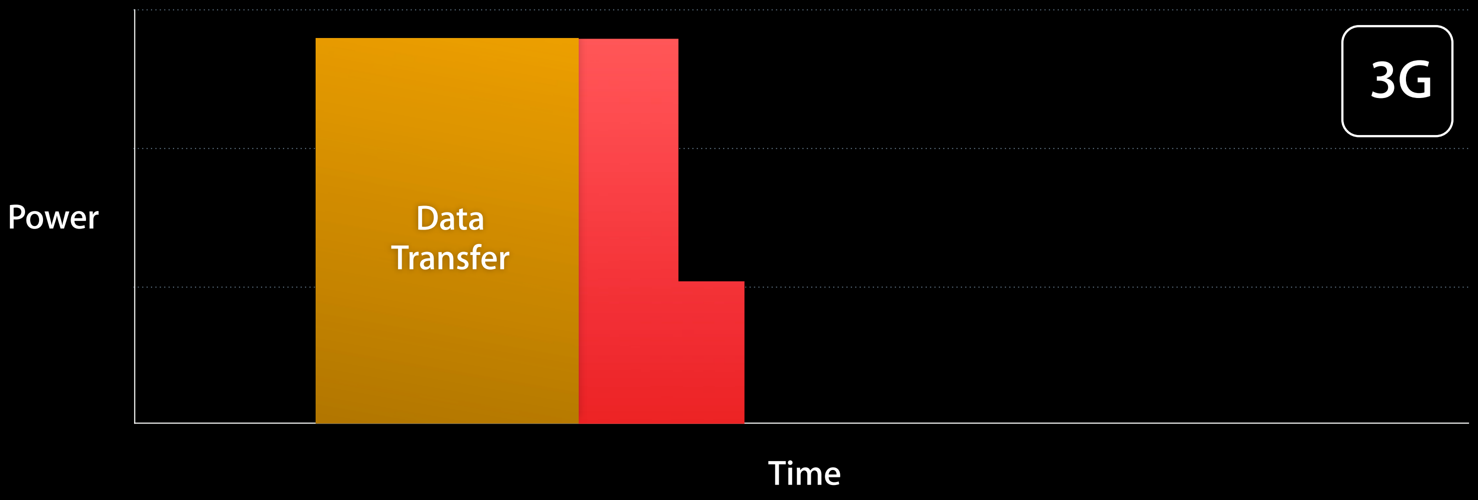
- Very expensive to send data
- 3G networks require phones stay in high-power state for a few seconds after last packet is sent or received



Background Audio

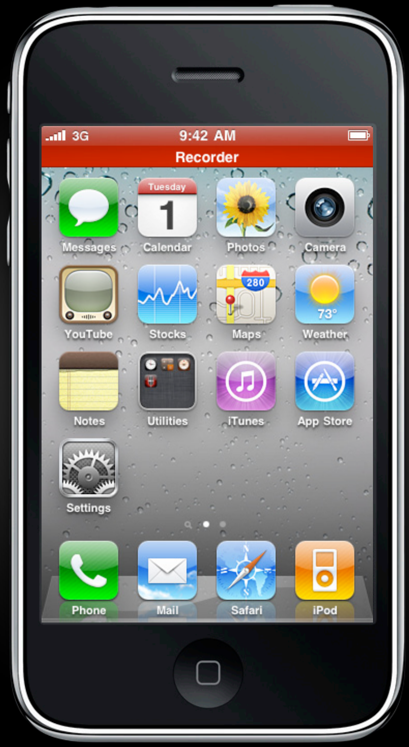
Best practices

- Coalesce data into large chunks, rather than thin stream
- Minimize amount of data transmitted



Background Audio

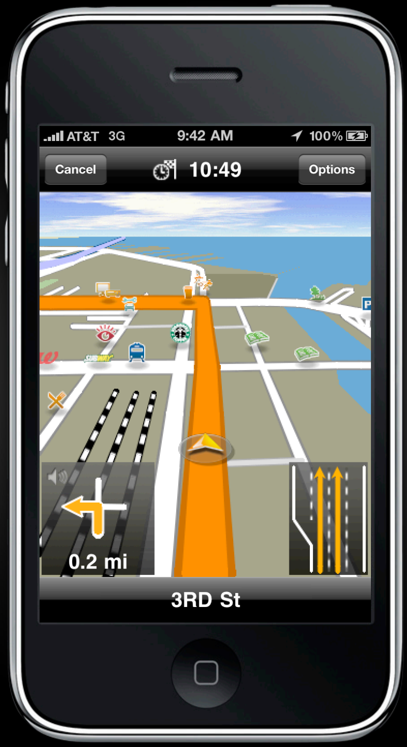
Recording audio



- The `AVAudioSessionCategoryRecord` category allows an application to record in the background
- The system creates a double-height status bar while recording in the background
  - Privacy
  - Tap to return

Demo
Background audio

Navigation



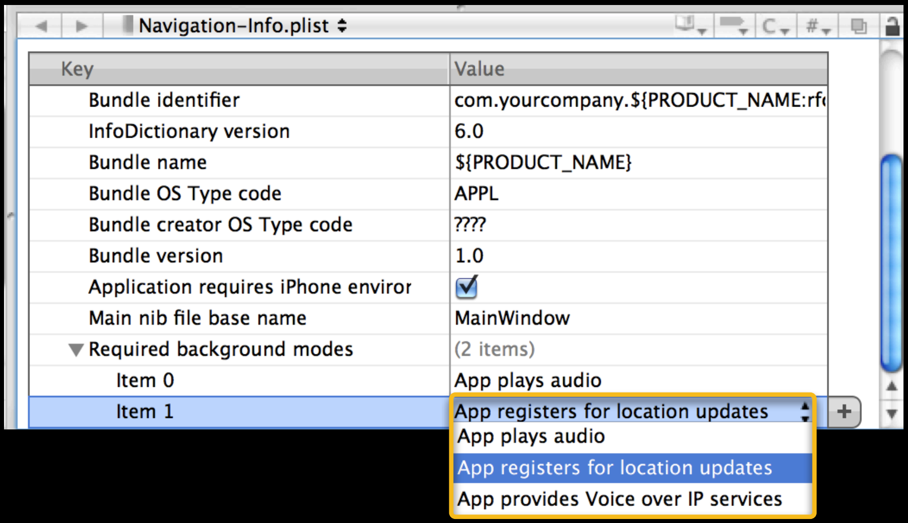
Navigation

Example application

- Keeps users informed of their location
- Gives turn-by-turn directions
- Speaks directions

Navigation

Background modes



Navigation-Info.plist

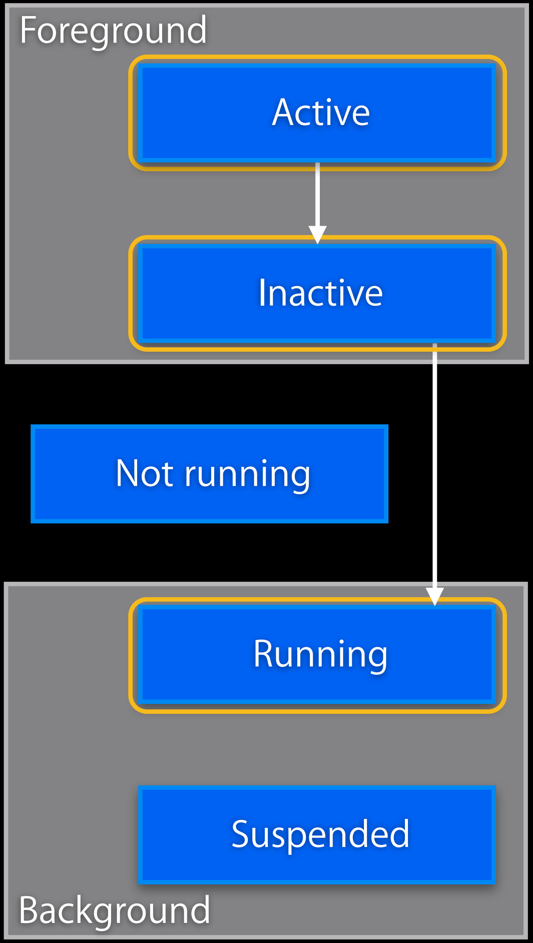
Key	Value
Bundle identifier	com.yourcompany.${PRODUCT_NAME:rf
InfoDictionary version	6.0
Bundle name	${PRODUCT_NAME}
Bundle OS Type code	APPL
Bundle creator OS Type code	????
Bundle version	1.0
Application requires iPhone enviror	✓
Main nib file base name	MainWindow
▼ Required background modes	(2 items)
Item 0	App plays audio
Item 1	App registers for location updates

App plays audio
App registers for location updates
App provides Voice over IP services

- App registers for location updates
  - Allows app to receive updates in the background with high accuracy
- App plays audio
  - Allows app to speak directions in the background

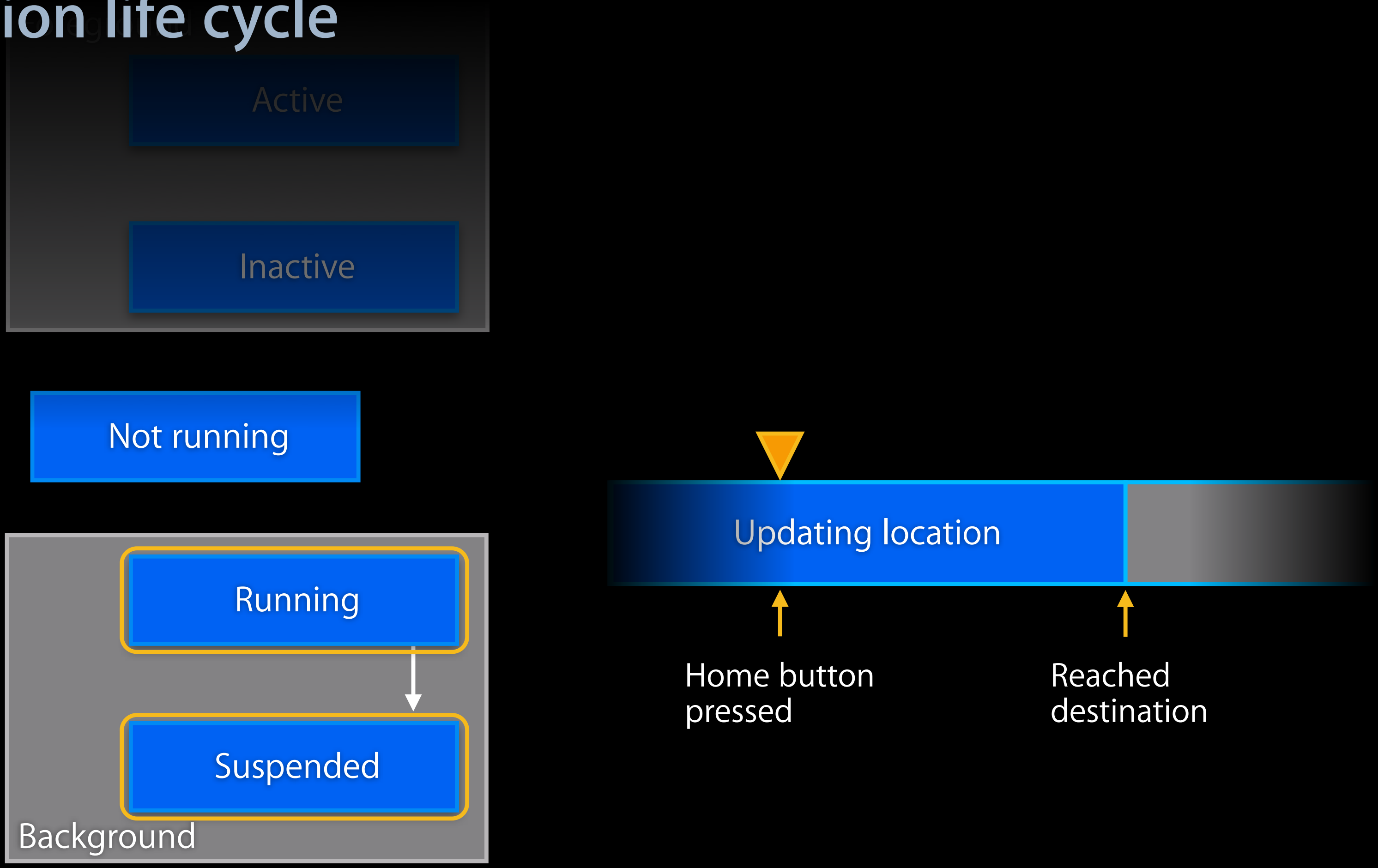
Navigation

Application life cycle



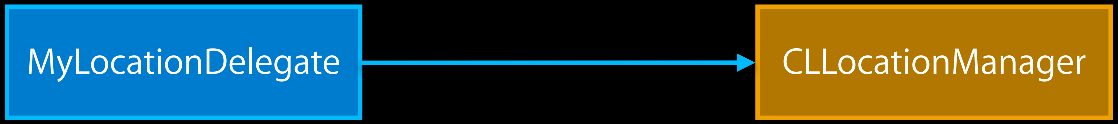
Navigation

Application life cycle



Navigation

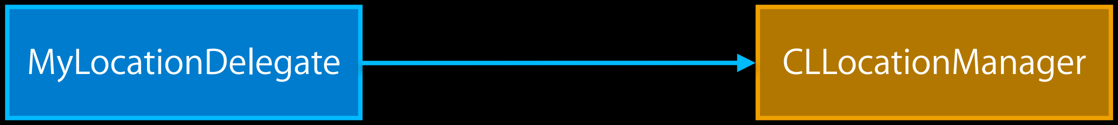
Location services



```
–[manager setDesiredAccuracy:kCLLocationAccuracyBestForNavigation]
```

Navigation

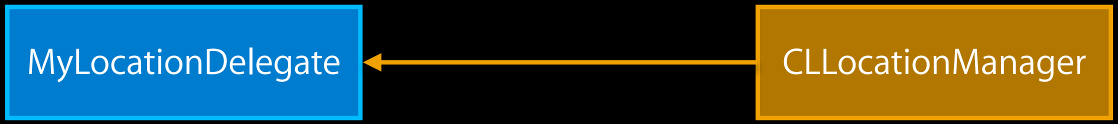
Location services



```
-startUpdatingLocation
```

Navigation

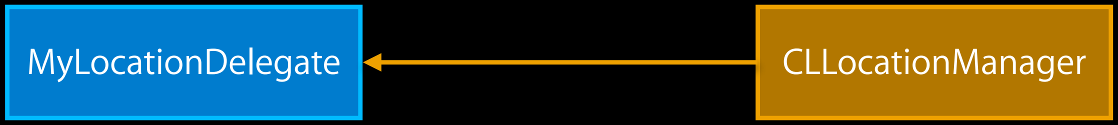
Location services



```
-locationManager:didUpdateToLocation:fromLocation:
```

Navigation

Location services



```
-locationManager:didUpdateToLocation:fromLocation:
```

Navigation

Audio categories

```
- (BOOL)setupWithError:(NSError **)error
{
    UInt32 mix = 1, duck = 1;
    OSStatus status = kAudioServicesNoError;

    [session setCategory:AVAudioSessionCategoryPlayback
                   error:error];
    if (*error) return NO;

    status = AudioSessionSetProperty (
        kAudioSessionProperty_OverrideCategoryMixWithOthers,
        sizeof(mix), &mix );
    if (status != kAudioServicesNoError) return NO;

    status = AudioSessionSetProperty (
        kAudioSessionProperty_OtherMixableAudioShouldDuck,
        sizeof(duck), &duck );
    if (status != kAudioServicesNoError) return NO;

    return YES;
}
```

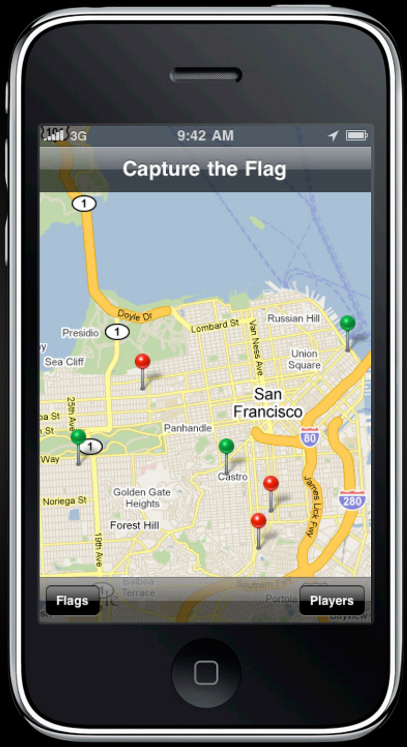
Navigation
Best practices

- Minimize CPU usage
- Turn off location updates after reaching the destination

Location Tracking

Location Tracking

Example application



- Responds to location changes while in the background
- Location-aware Capture the Flag application
- Capture other team's flag by entering their region and returning to yours
- Can display a map of all players

Location Tracking

- Significant location changes
  - Sends a notification after changing cell towers
- Region monitoring
  - Sends a notification upon entering and exiting regions of interest

Location Tracking

	Significant Location Changes	Region Monitoring
Uses less power than standard location services	✓	✓
Resumes suspended applications	✓	✓
Launches terminated applications	✓	✓
Notifications are not coalesced		✓
Supported on iPhone 4	✓	✓
Supported on iPhone 3GS	✓	

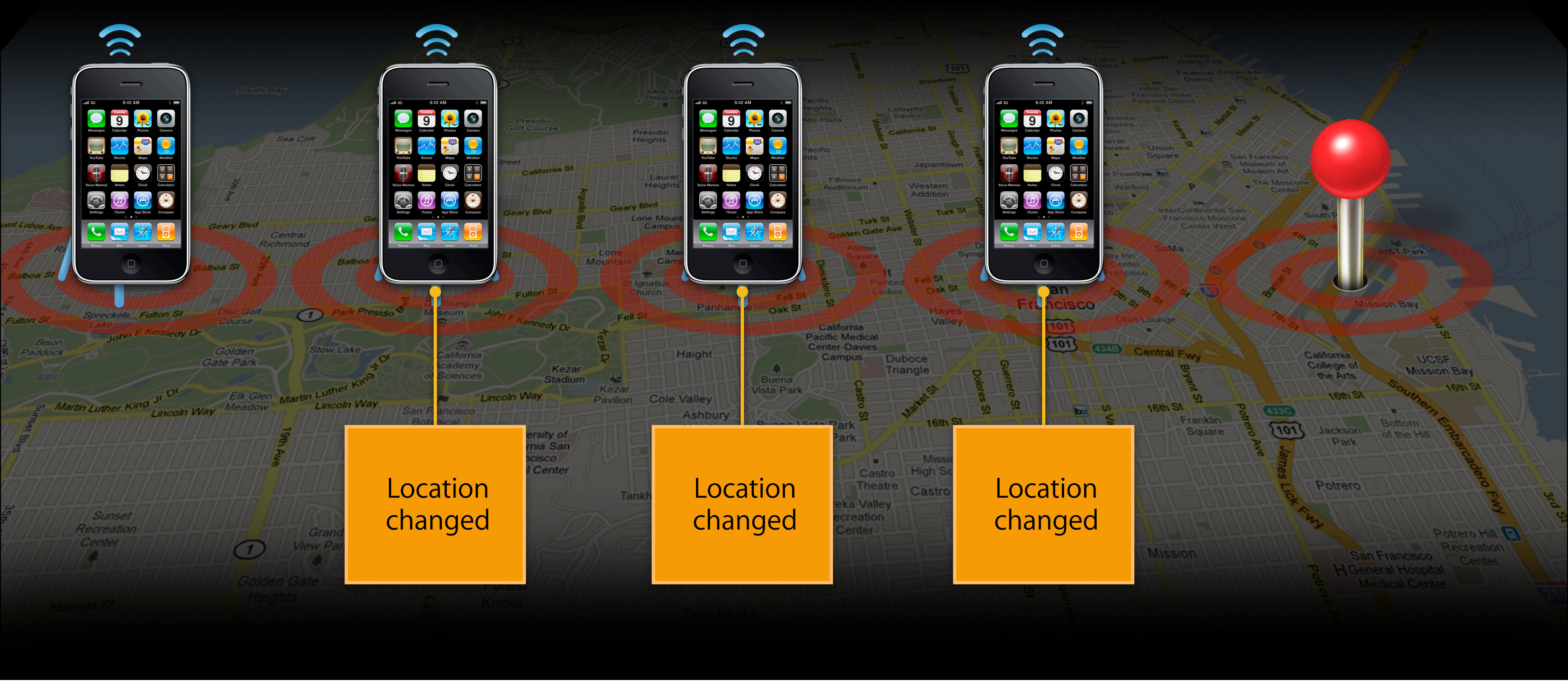
Location Tracking

Significant location changes

- Sends a notification after moving a significant distance
- Calculates position after changing cell towers
- Accuracy similar to cell positioning
- Notifications may be coalesced while device sleeps to prevent battery drain

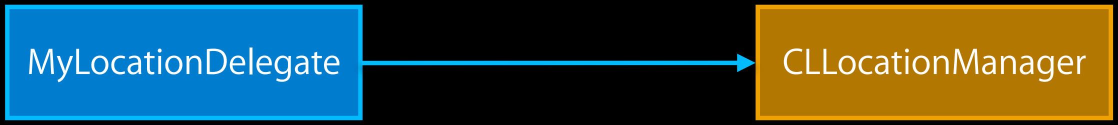
Location Tracking

Significant location changes



Location Tracking

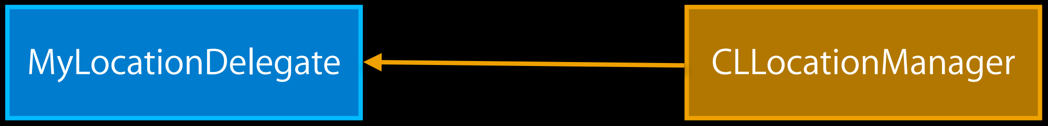
Using significant location changes



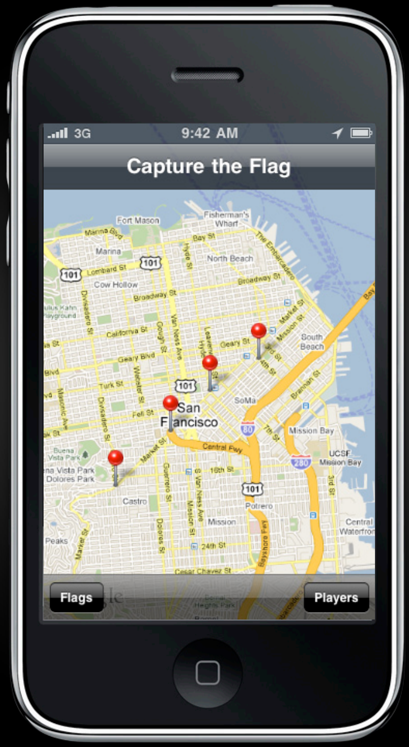
```
-startMonitoringSignificantLocationChanges
```

Location Tracking

Using significant location changes



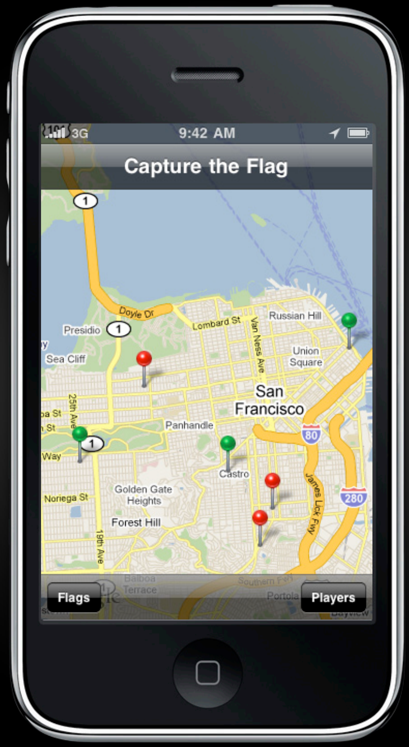
```
-locationManager:didUpdateToLocation:fromLocation:
```



Location Tracking

Using significant location changes

- On a location update
  - Use the Task Completion API to keep running
  - Upload location to the server



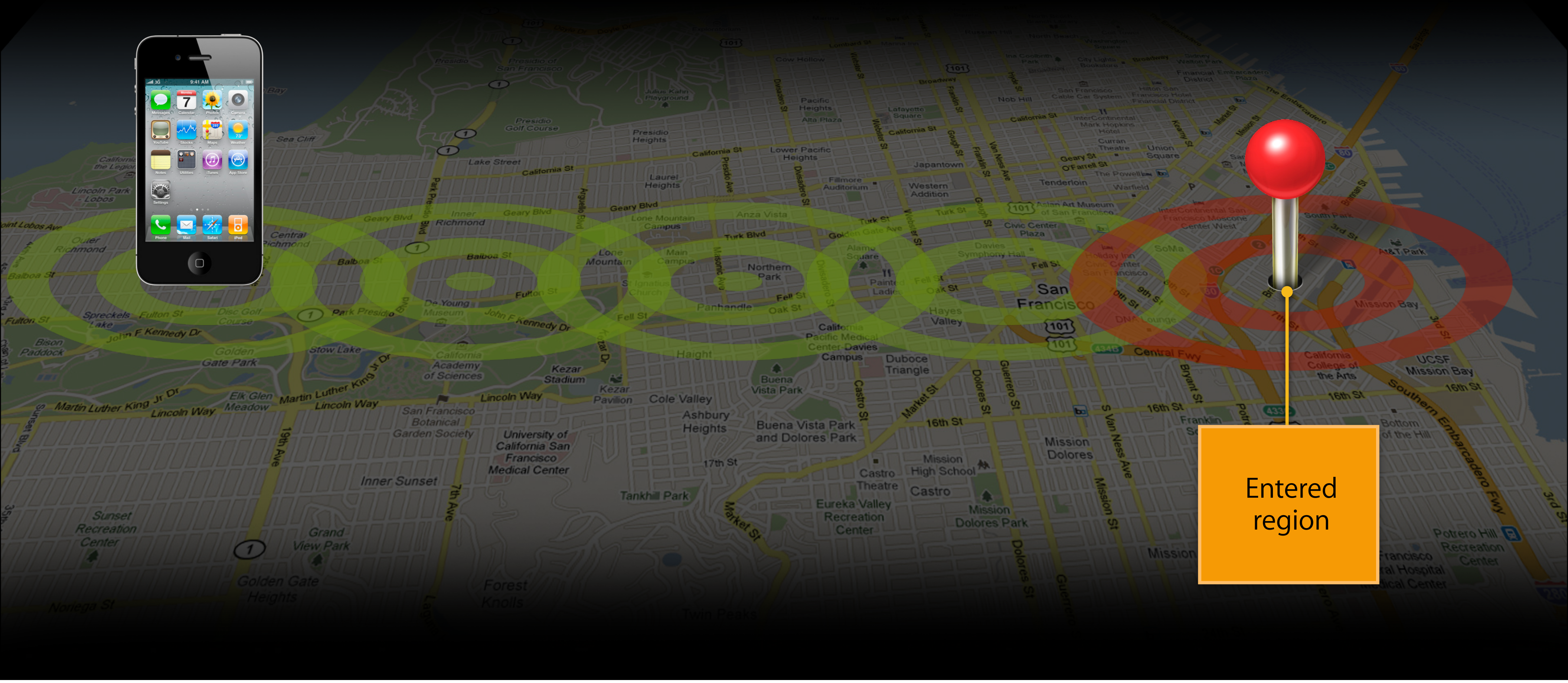
Location Tracking

Region monitoring

- Sends a notification upon entering or exiting regions of interest
- Application is suspended until entering or exiting a region
- Based on cell-positioning
- Also lower power than standard location services
- Limited number of regions
- Only supported on iPhone 4

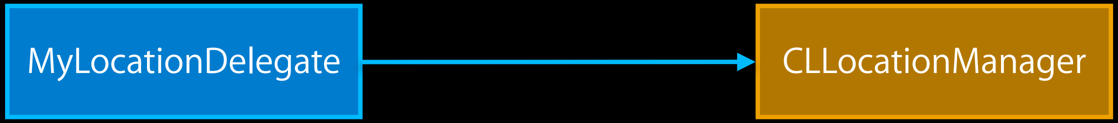
Location Tracking

Region monitoring



Location Tracking

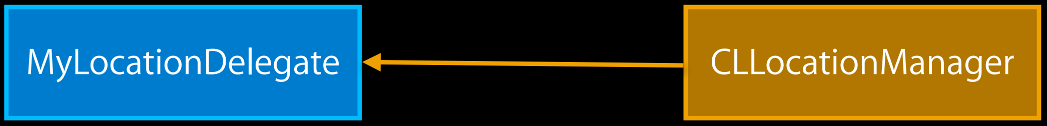
Using region monitoring



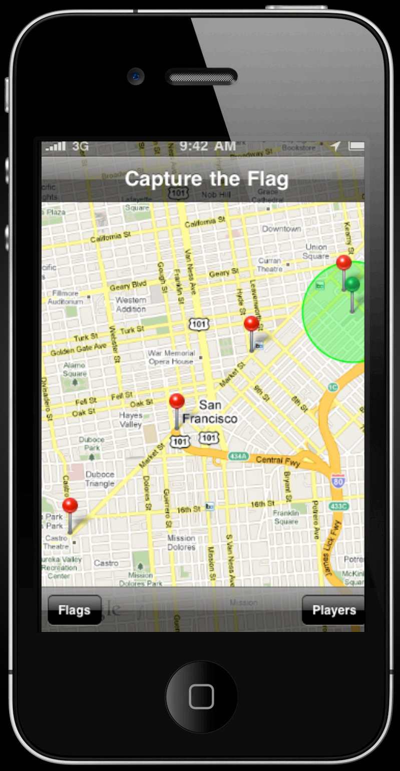
```
-startMonitoringForRegion:
```

Location Tracking

Using region monitoring

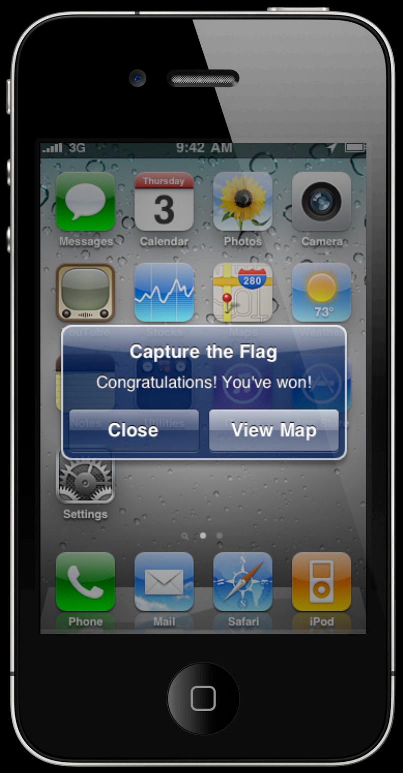


```
-locationManager:didEnterRegion:
 -locationManager:didExitRegion:
```



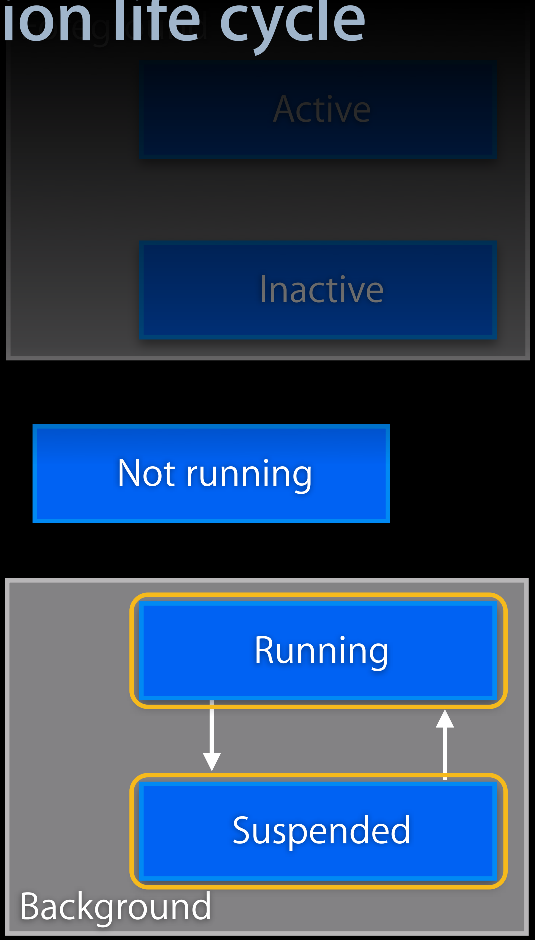
Location Tracking

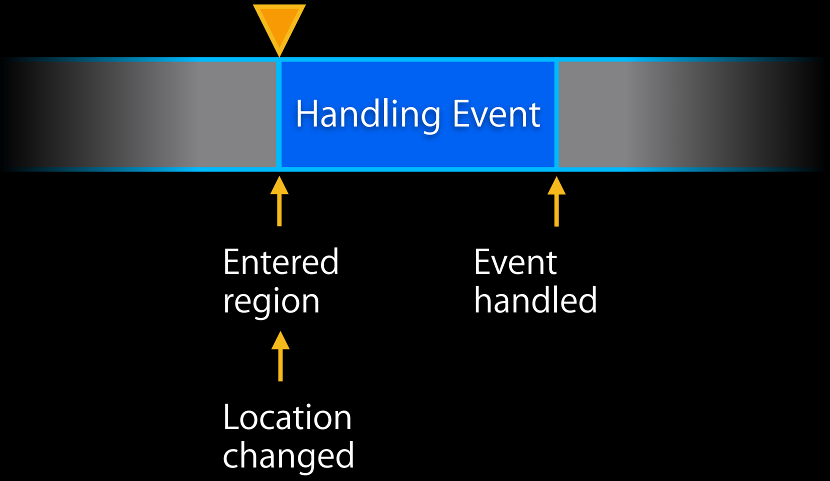
Using region monitoring


Capture the Flag

Congratulations! You've won!

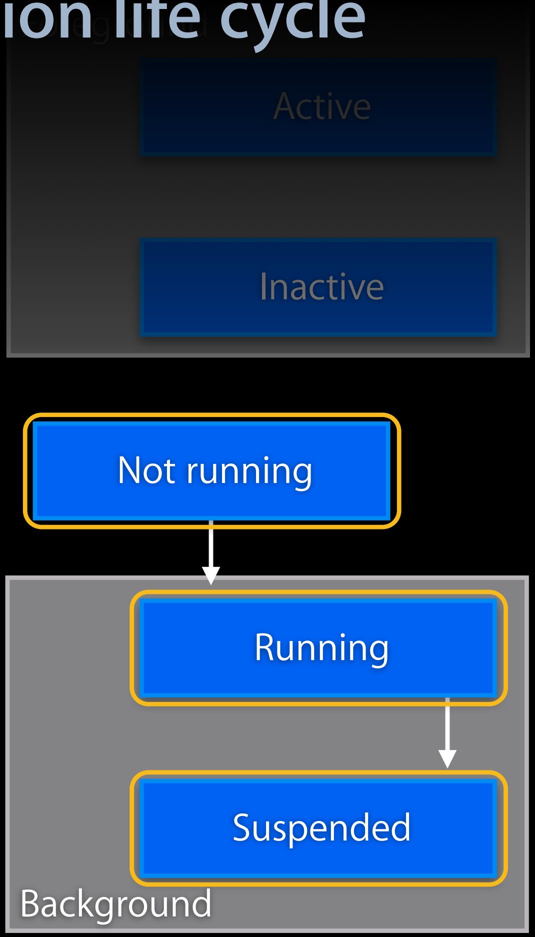
Location Tracking
Application life cycle

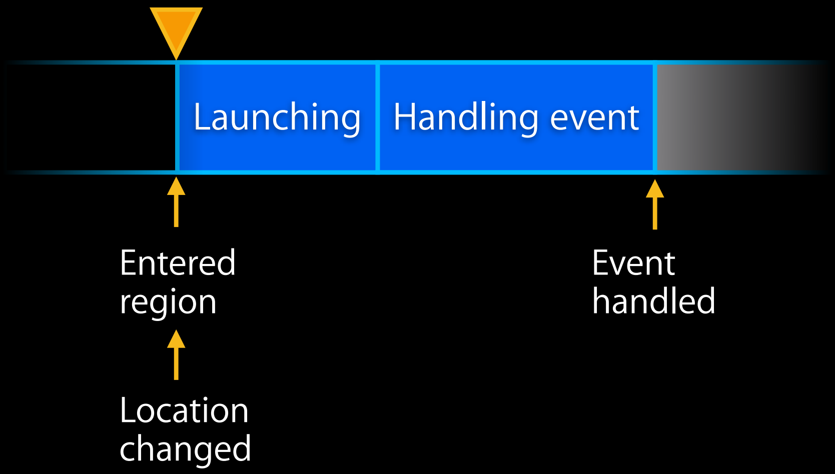




Location Tracking

Application life cycle





Location Tracking

Application life cycle

```
          - (void)application:(UIApplication *)app
didFinishLaunchingWithOptions:(NSDictionary *)options
{
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];

    BOOL launchedForLocation =
        [[options objectForKey:UIApplicationLaunchOptionsLocationKey]
            boolValue];

    if (launchedForLocation) {
        // Create and configure a CLLocationManager...
    }
}
```

Location Tracking

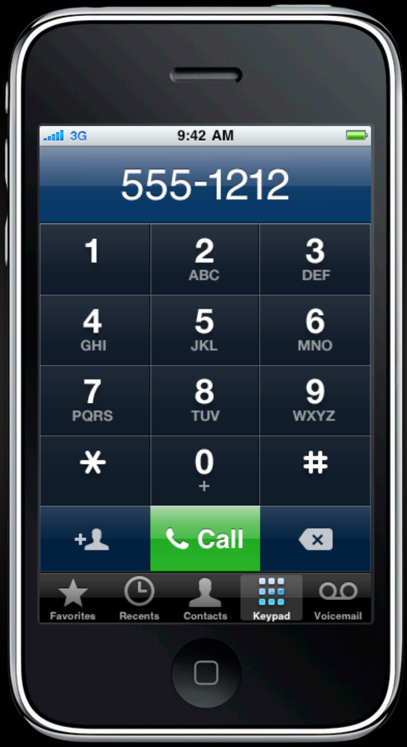
Best practices

- Use significant location changes and region monitoring if possible
  - Only use standard location services if needed
  - Can use standard location services on a location change as well
- Stop significant location updates when no longer needed
- Stop monitoring regions when no longer needed

Voice over IP

Voice over IP

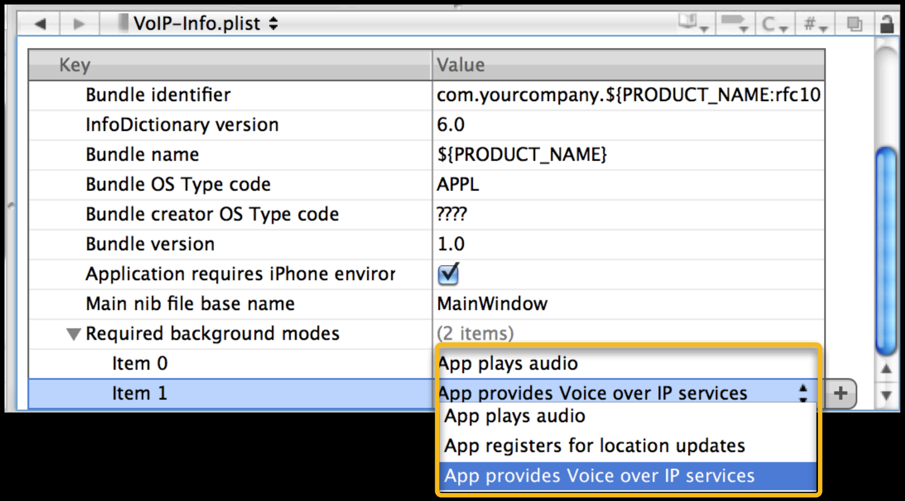
Example application



- Makes and receives phone calls using an Internet connection
- Notifies users of incoming calls
- Receives calls in the background
- Stays on a call when entering the background

Voice over IP

Background modes



VoIP-Info.plist

Key	Value
Bundle identifier	com.yourcompany.${PRODUCT_NAME:rfc10
InfoDictionary version	6.0
Bundle name	${PRODUCT_NAME}
Bundle OS Type code	APPL
Bundle creator OS Type code	????
Bundle version	1.0
Application requires iPhone enviror	✓
Main nib file base name	MainWindow
▼ Required background modes	(2 items)
Item 0	App plays audio
Item 1	App provides Voice over IP services

App plays audio
App registers for location updates
App provides Voice over IP services

- App provides Voice over IP services
  - Enables VoIP API
- App plays audio
  - Enables a call in the background

Voice over IP

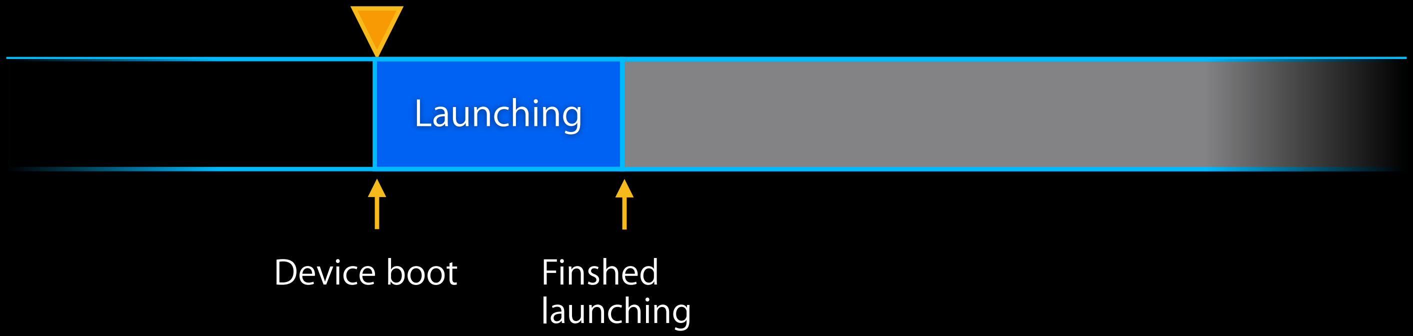
- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP

- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP
Application life cycle

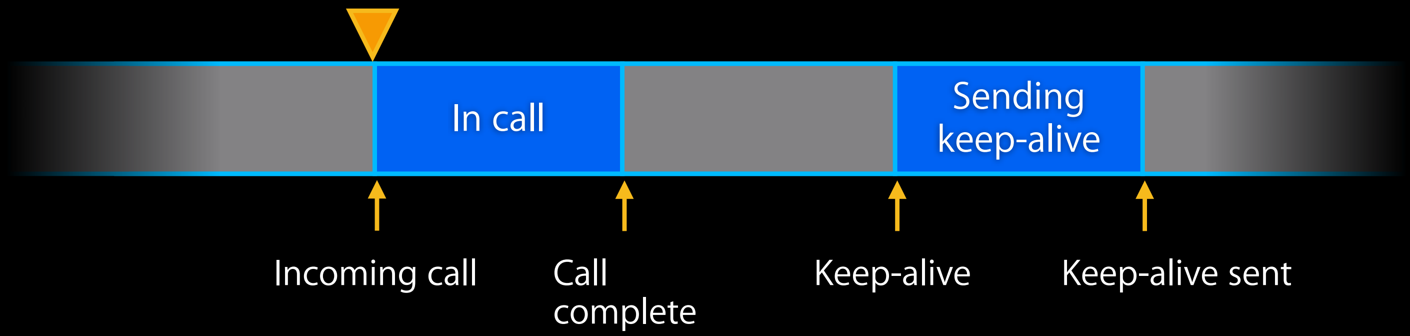
- Applications are launched on boot
- Relaunched if terminated
- Suspended until needed



Voice over IP

Application life cycle

- Resumed for incoming calls and sending keep-alives
- Keep-alives used to maintain a network connection



Voice over IP

- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP

Signaling connection

```
CFReadStreamRef readStream;
CFWriteStreamRef writeStream;

CFStreamCreatePairWithSocketToHost(NULL, host, port,
                                   &readStream, &writeStream);

NSInputStream *inputStream = (NSInputStream *)readStream;
NSOutputStream *outputStream = (NSOutputStream *)writeStream;

success = [self prepareStream:inputStream];
if (!success)
    return NO;

success = [self prepareStream:outputStream];
if (!success)
    return NO;
```

Voice over IP

Signaling connection

```
- (BOOL)prepareStream:(NSStream *)stream {
    BOOL success = NO;

    success = [stream setProperty:NSStreamNetworkServiceTypeVoIP
                           forKey:NSStreamNetworkServiceType];
    if (!success)
        return NO;

    [stream setDelegate:self];

    [stream scheduleInRunLoop:[NSRunLoop currentRunLoop]
                      forMode:NSDefaultRunLoopMode];

    [stream open];

    return YES;
}
```

Voice over IP

Signaling connection

- Can wrap POSIX/BSD sockets
- TCP streams only
  - Only needed on the signaling channel
  - Call's audio prevents suspend

Voice over IP

Keep-alive handlers

- Signaling channel can timeout
  - NAT
  - Protocol
- Set a keep-alive handler with `setKeepAliveTimeout:handler:` on `UIApplication`
- System calls keep-alive handler periodically to send keep-alive packets
- Minimum interval is 10 minutes

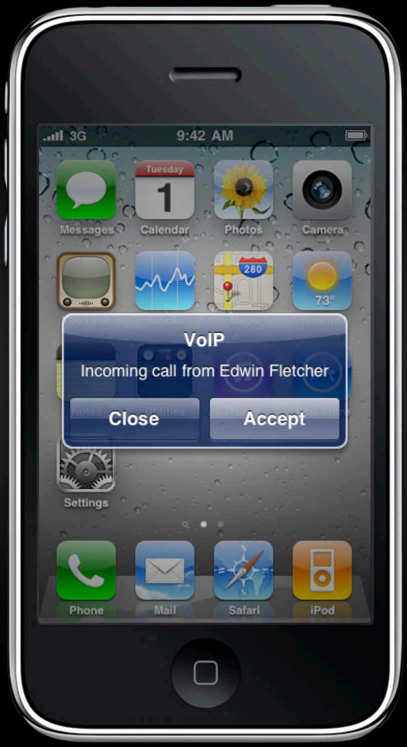
Voice over IP

- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP

Incoming call notification

- Notify the user of incoming calls with Local Notifications
- Call `presentLocalNotificationNow:` on `UIApplication`
- Can dismiss local notifications to avoid stacking



Voice over IP

- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP

Implementing audio behaviors

```
- (BOOL)setupWithError:(NSError **)error
{
    [session setCategory:AVAudioSessionCategoryPlayAndRecord
                   error:error];
    if (error)
        return NO;

    return YES;
}
```

Voice over IP

Implementing audio behaviors

- Allows simultaneous access to input and output
- Silences other audio
- Enables output to both receiver and speaker

Voice over IP

Implementing audio behaviors

```
- (void)myCallDidFinishWithError:(NSError **)error
{
    int flags = AVAudioSessionSetActiveFlags_NotifyOthersOnDeactivation;

    // Call is done
    [session setActive:NO withFlags:flags error:&errRet];

    // update UI, ...
}
```

Voice over IP

Implementing audio behaviors

- Tells other applications to resume their audio
- Sets `AVAudioSessionInterruptionFlags_ShouldResume` flag

Voice over IP

- Respond to incoming calls quickly
- Maintain a signaling connection
- Notify the user on an incoming call
- Implement the appropriate audio behaviors
- Put VoIP calls on hold during a cellular call

Voice over IP

Putting VoIP Calls on Hold

- New CoreTelephony framework
- Register a call event handler with `setCallEventHandler:` on `CTCallCenter`
- Notifies applications when the user
  - Receives an incoming cellular call
  - Ends the current cellular call
- Put VoIP calls on hold while on cellular calls

Voice over IP

Best practices

- VoIP apps are also audio applications
  - Minimize CPU
  - Avoid using large amounts of memory
- Use a long keep-alive interval
  - Maximizes battery life
  - 29 minutes is a good tradeoff

Summary

- Some applications benefit from executing in the background
- For those applications we provide some new services
  - Task completion
  - Background audio
  - Navigation
  - Location tracking
  - VoIP

Related Sessions

Adopting Multitasking on iPhone OS, Part 1	Marina Friday, 9:00AM
Audio Development for iPhone OS, Part 1	Mission Wednesday 9:00AM
Audio Development for iPhone OS, Part 2	Mission Wednesday 11:30AM
Using Core Location in iOS 4	Presidio Wednesday 10:15AM
Implementing Local and Push Notifications	Mission Thursday 2:00PM
Introducing Blocks and Grand Central Dispatch on iPhone	Russian Hill Wednesday 11:30AM

Labs

Multitasking Lab	Application Frameworks Lab D Tuesday 4:30PM-6:30PM
Multitasking Lab	Application Frameworks Lab A Wednesday 2:00PM-4:15PM
Multitasking Lab	Application Frameworks Lab A Friday 11:30AM-1:00PM

More Information

Michael Jurewitz
Developer Tools and Performance Evangelist
jurewitz@apple.com

Documentation
iPhone Application Programming Guide
http://developer.apple.com/iphone

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

WWDC10