Introducing On Demand Resources
An element of App Thinning
Session 214

Steve Lewallen Frameworks Engineering
Tony Parker Cocoa Frameworks
Agenda

On Demand Resources (ODR)
Agenda

On Demand Resources (ODR)

Overview
Agenda
On Demand Resources (ODR)

Overview
Features and Benefits
Agenda

On Demand Resources (ODR)

Overview
Features and Benefits
Details
Agenda

On Demand Resources (ODR)

Overview
Features and Benefits
Details
Using On Demand Resources
Agenda

On Demand Resources (ODR)

Overview
Features and Benefits
Details
Using On Demand Resources
Best Practices
Overview

The traditional application
Traditional Application
Traditional Application

- Executable
- Base Resources
Traditional Application

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level 'n'
Traditional Application

App Store

Executable

Base Resources

Game Level 1

Game Level 2

Game Level ‘n’
Traditional Application

App Store
- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’

Device
- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’
Traditional Application
Traditional Application
Traditional Application
Overview

The On Demand Resources application
On Demand Resources Application

Space-friendly packaging
On Demand Resources Application

Space-friendly packaging

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’
On Demand Resources Application

Space-friendly packaging

- Executable
- Base Resources
  - Game Level 1
  - Game Level 2
  - Game Level ‘n’
On Demand Resources Application

App Store

Executable

Base Resources

Game Level 1

Game Level 2

Game Level ‘n’
On Demand Resources Application

App Store

Executable

Base Resources

Game Level 1

Game Level 2

Game Level ‘n’

Device

Executable

Base Resources
On Demand Resources Application

**App Store**
- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’

**Device**
- Executable
- Base Resources
- Game Level 1
On Demand Resources Application

App Store

Executable

Base Resources

Game Level 1

Game Level 2

Game Level ‘n’

Device

Executable

Base Resources

Game Level 1
On Demand Resources Application

App Store
-Executable
  -Base Resources
    -Game Level 1
    -Game Level 2
    -Game Level 'n'

Device
-Executable
  -Base Resources
    -Game Level 1
    -Game Level 2
On Demand Resources Application

App Store

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level 'n'

Device

- Executable
- Base Resources
- Game Level 1
- Game Level 2
On Demand Resources Application

App Store
- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level 'n'

Device
- Executable
- Base Resources
- Game Level 1
- Game Level 2
### On Demand Resources Application

<table>
<thead>
<tr>
<th>App Store</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Executable</strong></td>
<td><strong>Executable</strong></td>
</tr>
<tr>
<td><strong>Base Resources</strong></td>
<td><strong>Base Resources</strong></td>
</tr>
<tr>
<td><strong>Game Level 1</strong></td>
<td><strong>Game Level 1</strong></td>
</tr>
<tr>
<td><strong>Game Level 2</strong></td>
<td><strong>Game Level 2</strong></td>
</tr>
<tr>
<td><strong>Game Level ’n’</strong></td>
<td><strong>Game Level ’n’</strong></td>
</tr>
</tbody>
</table>
On Demand Resources Application

App Store

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’

Device

- Executable
- Base Resources
- Game Level 2
- Game Level ‘n’
On Demand Resources Application

App Store

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’

Device

- Executable
- Base Resources
- Game Level 2
- Game Level ‘n’
Features

Benefits for the developer
On Demand Resources

Features

Dynamically loaded content
On Demand Resources

Features

Dynamically loaded content
Hosted on the App Store
On Demand Resources

Features

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
On Demand Resources

Features

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Automated downloads before first launch
On Demand Resources

Features

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Automated downloads before first launch
Intelligent content caching
On Demand Resources

Features

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Automated downloads before first launch
Intelligent content caching
Max app size with On Demand Resources increases to 20GB
Features

Benefits for the user
On Demand Resources

User benefits
On Demand Resources

User benefits

Better install experience
On Demand Resources

User benefits

Better install experience
More apps ready to run
On Demand Resources

User benefits

Better install experience
More apps ready to run
Greater, richer app content
Details

Understanding On Demand Resources
On Demand Resources
An element of App Thinning
On Demand Resources
An element of App Thinning

Part of iOS 9's App Thinning
On Demand Resources
An element of App Thinning

Part of iOS 9's App Thinning
Integrated with App Slicing
On Demand Resources
An element of App Thinning

Part of iOS 9's App Thinning
Integrated with App Slicing
• Tailoring the download to a specific device
On Demand Resources App Structure

Partitioning content
On Demand Resources App Structure

Partitioning content

Partition assets into tagged groups

<table>
<thead>
<tr>
<th>Executable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Resources</td>
</tr>
<tr>
<td>Game Level 1</td>
</tr>
<tr>
<td>Game Level 2</td>
</tr>
<tr>
<td>Game Level ’n’</td>
</tr>
</tbody>
</table>
On Demand Resources App Structure

Partitioning content

Partition assets into tagged groups

- Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’
On Demand Resources App Structure

Partitioning content

Partition assets into tagged groups

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’
On Demand Resources App Structure

Partitioning content

Partition assets into tagged groups

Executable
Base Resources
Game Level 1
Game Level 2
Game Level 'n'

.Asset Packs

.app
Partitioning content

Partition assets into tagged groups

- Tag assets using Xcode
### On Demand Resources App Structure

#### Partitioning content

Partition assets into tagged groups

- Tag assets using Xcode
- Tags are simple strings

<table>
<thead>
<tr>
<th>Executable</th>
<th>Base Resources</th>
<th>Asset Packs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Game Level 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Game Level 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Game Level ‘n’</td>
<td></td>
</tr>
</tbody>
</table>

.\app
On Demand Resources App Structure

Partitioning content

Partition assets into tagged groups

- Tag assets using Xcode
- Tags are simple strings
  - Ex. “Level 1”
Partition assets into tagged groups

- Tag assets using Xcode
- Tags are simple strings
  - Ex. “Level 1”
- May be single files or entire folders
On Demand Resources App Structure

Types of content

Tagged assets

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’
On Demand Resources App Structure

Types of content

Tagged assets

• Images

Executable

Base Resources

Game Level 1

Game Level 2

Game Level ‘n’
On Demand Resources App Structure

Types of content

Tagged assets

• Images
• Sounds

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’
On Demand Resources App Structure

Types of content

Tagged assets
• Images
• Sounds
• Data

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’
On Demand Resources App Structure

Types of content

Tagged assets
- Images
- Sounds
- Data
- Scripts

Executable
- Base Resources
- Game Level 1
- Game Level 2
- Game Level ‘n’

.app

Asset Packs
On Demand Resources App Structure

Types of content

Tagged assets
- Images
- Sounds
- Data
- Scripts
- In-app purchased content

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’
On Demand Resources App Structure

Types of content

Tagged assets

- Images
- Sounds
- Data
- Scripts
- In-app purchased content
- No executable content

Executable
Base Resources
Game Level 1
Game Level 2
Game Level ‘n’

.app
Asset Packs
Hosting On Demand Resources

Whether at deployment or during development
Hosting On Demand Resources

Whether at deployment or during development

App Store
Hosting On Demand Resources
Whether at deployment or during development

App Store
Xcode
Hosting On Demand Resources

Whether at deployment or during development

App Store
Xcode
Xcode Server
Hosting On Demand Resources

Whether at deployment or during development

App Store
Xcode
Xcode Server
TestFlight
Hosting On Demand Resources
Whether at deployment or during development

App Store
Xcode
Xcode Server
TestFlight
Enterprise Deployment
Getting Started with On Demand Resources

Identifying your assets
Getting Started with On Demand Resources
Identifying your assets
Getting Started with On Demand Resources

Identifying your assets
Getting Started with On Demand Resources

Tagging your assets

Part 1—Developer tags assets
Getting Started with On Demand Resources

Tagging your assets

Part 1—Developer tags assets

• “Level 1”
Getting Started with On Demand Resources
Tagging your assets

Part 1—Developer tags assets
• “Level 1”
• “Level 2”
Getting Started with On Demand Resources

Tagging your assets

Part 1—Developer tags assets

• “Level 1”
• “Level 2”
• “Level 1,” “Level 2”
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs

Level 1

Level 2

Level 1 + Level 2
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs

Level 1

Level 2

Level 1 + Level 2
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs
Getting Started with On Demand Resources

Xcode creates asset packs from tags

Part 2—Xcode creates asset packs
Getting Started with On Demand Resources

Developer requests assets

Part 3—Developer requests assets by tag

• Request “Level 1”
Part 3—Developer requests assets by tag

- Request “Level 1”
Getting Started with On Demand Resources

Developer requests assets

Part 3—Developer requests assets by tag

• Request “Level 1”

• Request “Level 2”
On Demand Resources API

Tony Parker Cocoa Frameworks
NSBundleResourceRequest
NSBundleResourceRequest

Foundation-level command object

- Set up with tags and other options
- Tell it to begin a request
NSBundleResourceRequest

Foundation-level command object

- Set up with tags and other options
- Tell it to begin a request

Create as many as you need—the system reference counts tags
NSBundleResourceRequest

Foundation-level command object

- Set up with tags and other options
- Tell it to begin a request

Create as many as you need—the system reference counts tags

Request decoupled from use of resources
NSBundleResourceRequest

State machine
NSBundleResourceRequest
State machine

init → requested
NSBundleResourceRequest
State machine
NSBundleResourceRequest
State machine

init → requested → available → ended

→ error
NSBundleResourceRequest

Basic methods
NSBundleResourceRequest

Basic methods

Initialize with set of tags

convenience init(tags: Set<String>)
NSBundleResourceRequest

Basic methods

Initialize with set of tags

```swift
convenience init(tags: Set<String>)
```

Begin a request

```swift
func beginAccessingResourcesWithCompletionHandler((NSError?) -> Void)
```
NSBundleResourceRequest

Basic methods

Initialize with set of tags

```
convenience init(tags: Set<String>)
```

Begin a request

```
func beginAccessingResourcesWithCompletionHandler((NSError?) -> Void)
```

Tell the system you’re finished

```
func endAccessingResources()
```
Demo

On Demand Resources
Progress Reporting

Information on progress of request

var progress: NSProgress
Progress Reporting

Information on progress of request

```swift
var progress: NSProgress
```

Cancel a request

```swift
request.progress.cancel()
request.progress.pause()
request.progress.resume()
```
Progress Reporting

Information on progress of request

```swift
var progress: NSProgress
```

Cancel a request

```swift
request.progress.cancel()
request.progress.pause()
request.progress.resume()
```

---

Best Practices for Progress Reporting

Pacific Heights

Friday 1:30PM
Conditional Requests
Conditional Requests

Access resources, but only if they are already downloaded

func conditionallyBeginAccessingResourcesWithCompletionHandler((Bool) -> Void)
Conditional Requests

Access resources, but only if they are already downloaded

```swift
func conditionallyBeginAccessingResourcesWithCompletionHandler((Bool) -> Void)
```
Conditional Requests

Access resources, but only if they are already downloaded

```swift
func conditionallyBeginAccessingResourcesWithCompletionHandler((Bool) -> Void)
```
Conditional Requests

Access resources, but only if they are already downloaded

```
func conditionallyBeginAccessingResourcesWithCompletionHandler((Bool) -> Void)
```
Conditional Requests

Access resources, but only if they are already downloaded

```swift
func conditionallyBeginAccessingResourcesWithCompletionHandler((Bool) -> Void)
```
var loadingPriority: Double
Loading Priority

var loadingPriority: Double
Provides ordering for outstanding requests in your app
Loading Priority

var loadingPriority: Double
Provides ordering for outstanding requests in your app
Value ranges from zero to one
Loading Priority

var loadingPriority: Double
Provides ordering for outstanding requests in your app
Value ranges from zero to one
For urgent requests, use NSBundleResourceRequestLoadingPriorityUrgent
extension NSBundle {
    func setPreservationPriority(Double, forTags: Set<String>)
}
extension NSBundle {
    func setPreservationPriority(Double, forTags: Set<String>)
}

Provides ordering of purging for unused tags in your app
extension NSBundle {
    func setPreservationPriority(Double, forTags: Set<String>)
}

Provides ordering of purging for unused tags in your app

Value ranges from zero to one
Best Practices
Building optimal On Demand Resources applications
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Random Access
Consider Your App’s Behavior
Understand app behavior to tag assets appropriately

Random Access
• Entirely unpredictable

Use many tags
• Tag small groups of assets for progressive download and consumption
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Random Access
• Entirely unpredictable

Use many tags
• Tag small groups of assets for progressive download and consumption
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Random Access

• Entirely unpredictable

Use many tags

• Tag small groups of assets for progressive download and consumption
Consider Your App’s Behavior
Understand app behavior to tag assets appropriately

Random Access

• Entirely unpredictable

Use many tags

• Tag small groups of assets for progressive download and consumption
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Limited Prediction
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Limited Prediction
- Semi-predictable
- Many possibilities will not be used

Use many tags
- Load subset of possibles
- Quickly end accessing on unused asset requests
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Limited Prediction
- Semi-predictable
- Many possibilities will not be used

Use many tags
- Load subset of possibles
- Quickly end accessing on unused asset requests
Consider Your App’s Behavior
Understand app behavior to tag assets appropriately

Limited Prediction
• Semi-predictable
• Many possibilities will not be used

Use many tags
• Load subset of possibles
• Quickly end accessing on unused asset requests
**Consider Your App’s Behavior**

Understand app behavior to tag assets appropriately

**Limited Prediction**
- Semi-predictable
- Many possibilities will not be used

**Use many tags**
- Load subset of possibles
- Quickly end accessing on unused asset requests
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Limited Prediction
- Semi-predictable
- Many possibilities will not be used

Use many tags
- Load subset of possibles
- Quickly end accessing on unused asset requests
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Limited Prediction
- Semi-predictable
- Many possibilities will not be used

Use many tags
- Load subset of possibles
- Quickly end accessing on unused asset requests
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear
Consider Your App’s Behavior
Understand app behavior to tag assets appropriately

Linear

- Majority of assets will be used

Download well in advance

End accessing assets when done using
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear

- Majority of assets will be used

Download well in advance

End accessing assets when done using
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear

- Majority of assets will be used

Download well in advance

End accessing assets when done using
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear

- Majority of assets will be used

Download well in advance

End accessing assets when done using
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear

• Majority of assets will be used

Download well in advance

End accessing assets when done using
Consider Your App’s Behavior

Understand app behavior to tag assets appropriately

Linear

• Majority of assets will be used

Download well in advance

End accessing assets when done using
The On Demand Resources Timeline
Scheduling resource loading
The On Demand Resources Timeline
Scheduling resource loading

App Launch

App Exit
The On Demand Resources Timeline

Scheduling resource loading

App Launch

Anticipate Need

App Exit
The On Demand Resources Timeline

Scheduling resource loading

App Launch

beginAccessing...

App Exit
The On Demand Resources Timeline
Scheduling resource loading

beginAccessing...

App Launch

Need Resources

App Exit
The On Demand Resources Timeline

Scheduling resource loading

beginAccessing...

“Loading…”

Need Resources

App Launch

→

App Exit
The On Demand Resources Timeline

Scheduling resource loading

- App Launch
- beginAccessing...
- Need Resources
- Resources Ready
- App Exit
The On Demand Resources Timeline
Scheduling resource loading

App Launch

beginAccessing...

Need Resources

Use Resources

Resources Ready

App Exit
The On Demand Resources Timeline
Scheduling resource loading

- Begin Accessing...
- Need Resources
- Resources Ready
- End Accessing...
- App Launch
- App Exit
Optimize App Installation

Include initially required resources with app install
Optimize App Installation

Include initially required resources with app install

Consider what is required immediately on first launch

• Ex. Level 1 of game
Optimize App Installation

Include initially required resources with app install

Consider what is required immediately on first launch

• Ex. Level 1 of game
• Add tags to Xcode’s “Initial Install Tags” prefetch priority

<table>
<thead>
<tr>
<th>Resource</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Install Tags</td>
<td>140MB</td>
</tr>
<tr>
<td>Common Textures</td>
<td>50MB</td>
</tr>
<tr>
<td>Common Sounds</td>
<td>30MB</td>
</tr>
<tr>
<td>Level 1</td>
<td>60MB</td>
</tr>
</tbody>
</table>
Optimize App Installation
Include initially required resources with app install

Consider what is required immediately on first launch

• Ex. Level 1 of game
• Add tags to Xcode’s “Initial Install Tags” prefetch priority
  - Included in total size of app in App Store

<table>
<thead>
<tr>
<th>Resource</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Install Tags</td>
<td>140MB</td>
</tr>
<tr>
<td>Common Textures</td>
<td>50MB</td>
</tr>
<tr>
<td>Common Sounds</td>
<td>30MB</td>
</tr>
<tr>
<td>Level 1</td>
<td>60MB</td>
</tr>
</tbody>
</table>
Automate Post-Installation

Downloading additional content

Consider what is required next

• Ex. Level 2 of game
Automate Post-Installation

Downloading additional content

Consider what is required next
• Ex. Level 2 of game
• Use Xcode to assign a prefetch order

<table>
<thead>
<tr>
<th>Prefetched Tag Order (120MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 2 Part 1 (40MB)</td>
</tr>
<tr>
<td>Level 2 Part 2 (45MB)</td>
</tr>
<tr>
<td>Level 2 Part 3 (35MB)</td>
</tr>
</tbody>
</table>
Caching

User behavior and developer hints inform caching
Caching

User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low
Caching

User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low

Attributes effecting purging
Caching

User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low

Attributes effecting purging

- Last used timestamp
Caching

User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low

Attributes effecting purging

• Last used timestamp
• Preservation priority
Caching

User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low

Attributes effecting purging

- Last used timestamp
- Preservation priority
  - A ranking of the most important tags (0.0 to 1.0)
Caching
User behavior and developer hints inform caching

On Demand Resources content may be purged when storage space is low

Attributes effecting purging

• Last used timestamp
• Preservation priority
  - A ranking of the most important tags (0.0 to 1.0)
• Application running state
Caching
Preserving On Demand Resources Content
Caching

Preserving On Demand Resources Content

Avoid large capacity tags in order to avoid over-purging

- 64MB per tag is recommended
Caching
Preserving On Demand Resources Content

Avoid large capacity tags in order to avoid over-purging
• 64MB per tag is recommended

Properly set preservation priority
• Avoid marking all tags with highest preservation priority of 1.0
Caching

Preserving On Demand Resources Content

Avoid large capacity tags in order to avoid over-purging
• 64MB per tag is recommended

Properly set preservation priority
• Avoid marking all tags with highest preservation priority of 1.0

Always indicate when done with a tag
• Call “... endAccessingResources” when done with a tag
• Allow the request to dealloc
Performance Considerations

Tradeoffs
Performance Considerations

Tradeoffs

On Demand Resources downloads balance resource utilization with speed of download

• Downloads may be done during game play while not overly consuming CPU
Performance Considerations

Tradeoffs

On Demand Resources downloads balance resource utilization with speed of download

- Downloads may be done during game play while not overly consuming CPU

Urgent loading priority

- Used when speed of download is most important, regardless of CPU

request.loadingPriority = NSBundleResourceRequestLoadingPriorityUrgent
Performance Testing
Analyzing download performance and latency
Performance Testing

Analyzing download performance and latency

Real world testing
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
Performance Testing
Analyzing download performance and latency

Real world testing
- Host assets via TestFlight or Xcode Server
- Use the Developer Tools “Network Link Conditioner”
Performance Testing
Analyzing download performance and latency

Real world testing
• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

- Host assets via TestFlight or Xcode Server
- Use the Developer Tools “Network Link Conditioner”
Performance Testing
Analyzing download performance and latency

Real world testing

- Host assets via TestFlight or Xcode Server
- Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing
Analyzing download performance and latency

Real world testing
• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing
Analyzing download performance and latency

Real world testing

- Host assets via TestFlight or Xcode Server
- Use the Developer Tools “Network Link Conditioner”
Performance Testing

Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Performance Testing
Analyzing download performance and latency

Real world testing

• Host assets via TestFlight or Xcode Server
• Use the Developer Tools “Network Link Conditioner”
Networking Errors
Properly handle errors for a better user experience
Networking Errors

Properly handle errors for a better user experience

No network connection at time of request

NSURLErrorNotConnectedToInternet
Networking Errors

Properly handle errors for a better user experience

No network connection at time of request

*NSURLErrorNotConnectedToInternet*

Resource does not exist

*NSURLErrorResourceUnavailable*
Storage Space Errors

Properly handle errors for a better user experience
Storage Space Errors

Properly handle errors for a better user experience

Requesting more than 2GB “in use” result in an error
Storage Space Errors

Properly handle errors for a better user experience

Requesting more than 2GB “in use” result in an error
Insufficient local storage results in notification

NSBundleResourceRequestLowDiskSpaceNotification
Cellular Data

Accounting for On Demand Resources downloads

On Demand Resources downloads
Cellular Data

Accounting for On Demand Resources downloads

On Demand Resources downloads

• Controlled by application’s cellular data switch
Cellular Data

Accounting for On Demand Resources downloads

On Demand Resources downloads

• Controlled by application’s cellular data switch
• Counted towards cellular data usage of application
Cellular Data

Accounting for On Demand Resources downloads

On Demand Resources downloads

• Controlled by application’s cellular data switch
• Counted towards cellular data usage of application

100MB cellular download install limit
Cellular Data

Accounting for On Demand Resources downloads

On Demand Resources downloads
• Controlled by application’s cellular data switch
• Counted towards cellular data usage of application

100MB cellular download install limit
• On Demand Resources with tags under “Initial Install Tags” count towards this limit
Vital Statistics
Numbers to remember
Vital Statistics

Numbers to remember

Maximum app size

- 2GB for .app bundle
- 20GB including On Demand Resources content
Vital Statistics
Numbers to remember

Maximum app size
• 2GB for .app bundle
• 20GB including On Demand Resources content
2GB of Initial and Prefetched On Demand Resources content (sliced, uncompressed)
Vital Statistics

Numbers to remember

Maximum app size

- 2GB for .app bundle
- 20GB including On Demand Resources content

2GB of Initial and Prefetched On Demand Resources content (sliced, uncompressed)

2GB of On Demand Resources “in use” content (sliced, uncompressed)

- “In use” On Demand Resources content is included in the size of your application via usage settings
Vital Statistics

Numbers to remember

Maximum app size

• 2GB for .app bundle
• 20GB including On Demand Resources content

2GB of Initial and Prefetched On Demand Resources content (sliced, uncompressed)
2GB of On Demand Resources “in use” content (sliced, uncompressed)
• “In use” On Demand Resources content is included in the size of your application via usage settings

Single asset pack must be no larger than 512MB (sliced, uncompressed)
On Demand Resources

Summary

Dynamically loaded content
On Demand Resources

Summary

Dynamically loaded content

Hosted on the App Store
On Demand Resources

Summary

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
On Demand Resources

Summary

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Prioritized downloads
On Demand Resources

Summary

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Prioritized downloads
Intelligent content caching
On Demand Resources

Summary

Dynamically loaded content
Hosted on the App Store
Downloadable during app install and by request
Prioritized downloads
Intelligent content caching
Max app size with On Demand Resources increases to 20GB
More Information

Documentation
On-Demand Resources Guide
NSBundleResourceRequest Class Reference
App Distribution Guide

Sample Code
DemoBots

Technical Support
Apple Developer Forums
Developer Technical Support

General Inquiries
Curt Rothert, App Frameworks Evangelist
rothert@apple.com

http://developer.apple.com/library
<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>App Thinning in Xcode</td>
<td>Presidio</td>
<td>Wednesday 9:00AM</td>
</tr>
<tr>
<td>Continuous Integration and Code Coverage in Xcode</td>
<td>Presidio</td>
<td>Thursday 10:00AM</td>
</tr>
<tr>
<td>Best Practices for Progress Reporting</td>
<td>Pacific Heights</td>
<td>Friday 1:30PM</td>
</tr>
<tr>
<td>On Demand Resources Lab</td>
<td>Frameworks Lab B</td>
<td>Thursday 11:00AM</td>
</tr>
</tbody>
</table>