Optimizing App Assets

Session 227

Will Li, Cocoa Frameworks
Patrick Heynen, Cocoa Frameworks
Optimizing App Assets

Use new features of Asset Catalogs to optimize the deployment of your application assets
Compression
Image Compression
Choose the appropriate compression type

- Inherited (Automatic)
- Automatic
- Lossless
- Lossy
- Basic
- GPU Best Quality
- GPU Smallest Size
Automatic image packing

Lossy compression

Lossless compression

Deployment target and app thinning

App variant export
Automatic Image Packing

Loose images have hidden costs

Difficult to manage large number of files

Inconsistent and uncertain image formats
Automatic Image Packing

Loose image files

Disk space (KB)

Asset footprint on disk

Loose image files
Automatic Image Packing

![Diagram](image_url)

- Image atlas
- Loose image files

<table>
<thead>
<tr>
<th>Disk space (KB)</th>
<th>Image atlas</th>
<th>Loose image files</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Asset footprint on disk
Automatic Image Packing

- Image atlas

- Disk space (KB)
  - 0
  - 12.5
  - 25
  - 37.5
  - 50

- Asset footprint on disk
  - Image atlas
  - Loose image files
Automatic image packing
Lossy compression
Lossless compression
Deployment target and app thinning
App variant export
Lossy Compression

Quality versus size tradeoff

Best suited for certain scenarios
High Efficiency Image File Format
High Efficiency Image File Format

Better compression ratio than JPEG
Supports transparency
Automatic conversion from other formats
Automatic image packing
Lossy compression
Lossless compression
Deployment target and app thinning
App variant export
Lossless Compression

Icons and simple artwork

Complex artwork
Lossless Compression

Icons and simple artwork

Complex artwork
Apple Deep Pixel Image Compression
Apple Deep Pixel Image Compression

Adaptive to image color spectrum

Selects optimal compression algorithm

15–20% size improvement
Apple Deep Pixel Image Compression

Size improvement

![Bar chart showing size improvement](chart.png)

- **iOS**: Apple Deep Pixel Image Compression vs Current lossless compression
- **macOS**: Apple Deep Pixel Image Compression vs Current lossless compression
- **tvOS**: Apple Deep Pixel Image Compression vs Current lossless compression
Apple Deep Pixel Image Compression

Size improvement

Overall Assets Size (MB)

iOS macOS tvOS

Apple Deep Pixel Image Compression

Current lossless compression

20%
Apple Deep Pixel Image Compression

Decode time improvement

iOS 12

Current lossless compression
Automatic image packing
Lossy compression
Lossless compression
Deployment target and app thinning
App variant export
Deployment Target and App Thinning

- iOS 12
- iOS 11.x
- iOS 10.x
- iOS 9.x

Traditional Backward Deployment
Deployment Target and App Thinning

iOS 12

iOS 11.x

iOS 10.x

iOS 9.x

Traditional Backward Deployment
Deployment Target and App Thinning

Traditional Backward Deployment

- iOS 12
- iOS 11.x
- iOS 10.x
- iOS 9.x

Backward Deployment with OS Variant Thinning

- iOS 12
- iOS 11.x
- iOS 10.x
- iOS 9.x
Deployment Target and App Thinning

Traditional Backward Deployment
- iOS 12
- iOS 11.x
- iOS 10.x
- iOS 9.x

Backward Deployment with OS Variant Thinning
- iOS 12
- iOS 11.x
- iOS 10.x
- iOS 9.x
App Thinning Export
None

- All compatible device variants
  - iPad (5th generation)
  - iPad (6th generation)
  - iPad Air
  - iPad Air 2
  - iPad Pro (10.5-inch)
  - iPad Pro (12.9-inch)
  - iPad Pro (12.9-inch) (2nd generation)
  - iPad Pro (9.7-inch)
  - iPad mini 2
  - iPad mini 3
  - iPad mini 4
  - iPhone 5s
  - iPhone 6
  - iPhone 6 Plus
  - iPhone 6s
  - iPhone 6s Plus
  - iPhone 7
  - iPhone 7 Plus
  - iPhone 8
  - iPhone 8 Plus
  - iPhone SE
  - iPhone X
  - iPod touch (6th generation)
App Thinning Export

- Variants: iPhone 6, iPhone 7 - iPhone 8, iPhone 10, iPad 4 - iPad 6
- iOS 11 and earlier (MB)
  - iPhone 6: 94 MB
  - iPhone 7 - iPhone 8: 94 MB
  - iPhone 10: 125 MB
  - iPad 4 - iPad 6: 109 MB
- iOS 12 (MB)
  - iPhone 6: 109 MB
  - iPhone 7 - iPhone 8: 125 MB
  - iPhone 10: 94 MB
  - iPad 4 - iPad 6: 109 MB
App Thinning Export

<table>
<thead>
<tr>
<th>Variants</th>
<th>iOS 11 and earlier (MB)</th>
<th>iOS 12 (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone 6</td>
<td>94</td>
<td>74</td>
</tr>
<tr>
<td>iPhone 7 - iPhone 8</td>
<td>94</td>
<td>72</td>
</tr>
<tr>
<td>iPhone 10</td>
<td>125</td>
<td>111</td>
</tr>
<tr>
<td>iPad 4 - iPad 6</td>
<td>109</td>
<td>95</td>
</tr>
</tbody>
</table>

iOS 11 and earlier (MB) vs iOS 12 (MB) for different variants.
App Thinning Export

<table>
<thead>
<tr>
<th>Variants</th>
<th>iOS 11 and earlier (MB)</th>
<th>iOS 12 (MB)</th>
<th>Size reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone 6</td>
<td>94</td>
<td>74</td>
<td>22%</td>
</tr>
<tr>
<td>iPhone 7 - iPhone 8</td>
<td>94</td>
<td>72</td>
<td>23%</td>
</tr>
<tr>
<td>iPhone 10</td>
<td>125</td>
<td>111</td>
<td>11%</td>
</tr>
<tr>
<td>iPad 4 - iPad 6</td>
<td>109</td>
<td>95</td>
<td>13%</td>
</tr>
</tbody>
</table>
Compression → Design and Production → Cataloging → Deployment
Design and Production
Artwork Assets

Assets come from many sources

They all come from humans!

Being organized pays big dividends
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Color Management

Pixels without color are just bytes!
Color profiles supply intended appearance
Maintain color profiles
Keep designer intent
Color Management

Pixels without color are just bytes!

Color profiles supply intended appearance

Maintain color profiles

Keep designer intent
Color Management

Apps run on broad range of displays

Color matching maps colors to device
Color Management

Asset Catalogs perform color matching at build time

Artwork ready on device

Bonus: Color Profile eliminated
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Working Space

Use consistent color settings for all design files
Use consistent color settings for all design files

sRGB / 8 bits for broad applicability

Display P3 / 16 bits for vibrant designs

Flexible Wide Color options
Working Space

Use consistent color settings for all design files
sRGB / 8 bits for broad applicability
Display P3 / 16 bits for vibrant designs
Flexible Wide Color options
Working with Wide Color

Working with P3

WWDC 2016

iOS Design Resources
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Stretchable Images

Adaptive UI uses stretching elements
Stretchable Images

Adaptive UI uses stretching elements
Stretchable Images

Design tools support slicing

Identify stretchable portion of image
Stretchable Images

Design tools support slicing

Identify stretchable portion of image
Stretchable Images

Split image resources require complex drawing to reassemble
Stretchable Images

Single image + metadata = smooth GPU animation
Stretchable Images

Single image + metadata = smooth GPU animation
Asset Catalog “Show Slicing”
Asset Catalog “Show Slicing”
Asset Catalog “Show Slicing”

Slicing

- Slices: Horizontal
- Left: 14
- Right: 14
- Center: Stretches
- Width: 10
Asset Catalog “Show Slicing”

- Slices: Horizontal
- Left: 14
- Right: 14
- Center: Stretches
- Width:

Options:
- Stretches
- Tiles
Asset Catalog Slicing

Keeps stretching metadata close to artwork

Better positioned for design updates
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Vector Assets

Distinct assets for different displays (1x, 2x, 3x)
Vector Assets

Distinct assets for different displays (1x, 2x, 3x)

Address all needs with one vector asset in PDF format
Vector Assets

Xcode generates optimized bitmaps per scale
Preserve Vector Data enables runtime resizing
Works better with Dynamic Type!
Color management
Working space
Stretchable images
Vector assets
Design for 2x
Design for 2x

Retina 2x is the most common display density
Strokes landing between pixels still look fuzzy
Design for 2x

Point boundary snapping ensures device pixel alignment.

Vector assets can use a 2x grid for optimal stroke placement.
Design for 2x

Point boundary snapping ensures device pixel alignment

Vector assets can use a 2x grid for optimal stroke placement
Design for 2x

Point boundary snapping ensures device pixel alignment

Vector assets can use a 2x grid for optimal stroke placement
Design for 2x

Drop asset into ‘2x’ slot in Xcode

Remaining scales are prepared automatically
Hinted Assets

More control of results

Drop into any slot
Cataloging
Lots of options
Use only what makes sense
Bundles

Namespaces
Bundles

Namespaces
Bundles

Large projects present challenges
Solve it with multiple bundles
Effective reuse strategy!
Bundles

Retrieve with:

```swift
UIImage(named: UIImage.Name, in: Bundle, compatibleWith: UITraitCollection)
Bundle.image(forResource: NSImage.Name) -> NSImage?
```

Uniqueness only within bundle scope
Bundles

Namespaces
Namespaces

Large collections are a naming challenge
Namespaces

Large collections are a naming challenge

Folders with namespace for grouping!
Design and
Production

Cataloging

Deployment
Deployment
App Thinning

You provide all the content variants

App Thinning picks the right subset per device
Performance classes
Sprite atlases
Performance classes

Sprite atlases
Performance Classes

Hardware capabilities vary

Don’t constrain to least capable device!

Solve with adaptive resources
<table>
<thead>
<tr>
<th>Memory Classes</th>
<th>1GB</th>
<th>2GB</th>
<th>3GB</th>
<th>4GB</th>
</tr>
</thead>
</table>

# Graphics Classes

<table>
<thead>
<tr>
<th>Metal 1</th>
<th>Metal 2</th>
<th>Metal 3</th>
<th>Metal 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>A7</td>
<td>A8</td>
<td>A9:A10</td>
<td>A11</td>
</tr>
</tbody>
</table>
## Full Capability Matrix

<table>
<thead>
<tr>
<th></th>
<th>Metal 4</th>
<th>Metal 3</th>
<th>Metal 2</th>
<th>Metal 1</th>
<th>Any</th>
</tr>
</thead>
<tbody>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Full Capability Matrix

<table>
<thead>
<tr>
<th></th>
<th>Metal 4</th>
<th>Metal 3</th>
<th>Metal 2</th>
<th>Metal 1</th>
<th>Any</th>
</tr>
</thead>
<tbody>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>♦</td>
</tr>
<tr>
<td>2GB</td>
<td></td>
<td>♦</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>♦</td>
</tr>
</tbody>
</table>

- ♦ indicates compatibility.
<table>
<thead>
<tr>
<th></th>
<th>Metal 4</th>
<th>Metal 3</th>
<th>Metal 2</th>
<th>Metal 1</th>
<th>Any</th>
</tr>
</thead>
<tbody>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Metal 4</td>
<td>Metal 3</td>
<td>Metal 2</td>
<td>Metal 1</td>
<td>Any</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2GB</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td></td>
<td></td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Metal 4</td>
<td>Metal 3</td>
<td>Metal 2</td>
<td>Metal 1</td>
<td>Any</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-----</td>
</tr>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><img src="red" alt="Circle" /></td>
</tr>
<tr>
<td>2GB</td>
<td><img src="red" alt="Circle" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td><img src="red" alt="Circle" /></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Metal 1</td>
<td>Metal 2</td>
<td>Metal 3</td>
<td>Metal 4</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Metal 4</td>
<td>Metal 3</td>
<td>Metal 2</td>
<td>Metal 1</td>
<td>Any</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>4GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3GB</td>
<td></td>
<td></td>
<td><img src="image" alt="Red Circle" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2GB</td>
<td><img src="image" alt="Red Circle" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1GB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td></td>
<td><img src="image" alt="Red Circle" /></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Metal 4</td>
<td>Metal 3</td>
<td>Metal 2</td>
<td>Metal 1</td>
<td>Any</td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>4GB</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3GB</strong></td>
<td></td>
<td><img src="green-circle.png" alt="Green Circle" /></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2GB</strong></td>
<td><img src="red-circle.png" alt="Red Circle" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1GB</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Any</strong></td>
<td></td>
<td><img src="red-circle.png" alt="Red Circle" /></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using Performance Classes

Higher memory —> larger / richer assets
Higher graphics —> more complex assets
Using Performance Classes

NSDataAsset provides flexible container

Example: Cut scene video

Example: Scene configuration via plist
Using Performance Classes

NSDataAsset provides flexible container

Example: Cut scene video

Example: Scene configuration via plist
Using Performance Classes

NSDataAsset provides flexible container

Example: Cut scene video

Example: Scene configuration via plist
Performance classes

Sprite atlases
Sprite Atlases

Pack related images into a single unit
Entire atlas gets loaded at once
Images reference location within atlas
Sprite Atlases

Access images using UIImage / NSImage

Use asynchronous loading for special cases

```swift
SKTextureAtlas.preloadTextureAtlasesNamed(_: [String],
    withCompletionHandler: (Error?, [SKTextureAtlas]) -> Void)
```
Sprite Atlases

Xcode automatically optimizes for App Thinning

Split by pixel format, device traits and compression type
Compression → Design and Production → Cataloging → Deployment
Optimizing App Assets

Best choice for image resources

10–20% less space for iOS 12 apps

App Thinning optimizes for latest OS

Use cataloging to adapt resources to your app
More Information

https://developer.apple.com/wwdc18/227