#WWDC18

Quick Look

Previews from the ground up

Session 237

Raffael Hannemann, Software Engineer
Maxime Uzan, Software Engineer

Introduction to Quick Look

Introduction to Quick Look

Adopting Quick Look Preview Controller

Introduction to Quick Look

Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Introduction to Quick Look

Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Providing custom thumbnails

Introduction to Quick Look

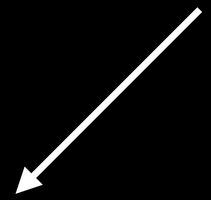
Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Providing custom thumbnails

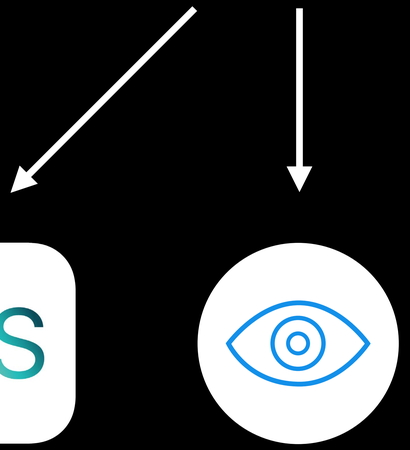
Quick Look

Quick Look



Previewing
Documents

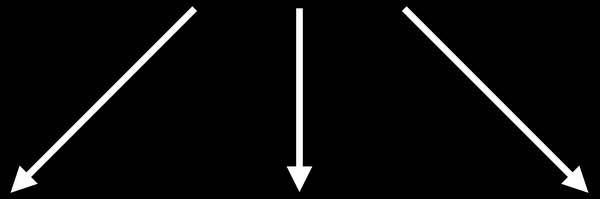
Quick Look



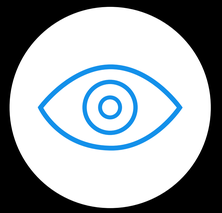
Previewing Documents

Providing Custom Previews

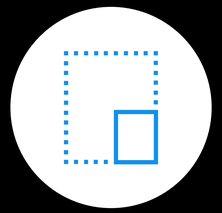
Quick Look



Previewing Documents

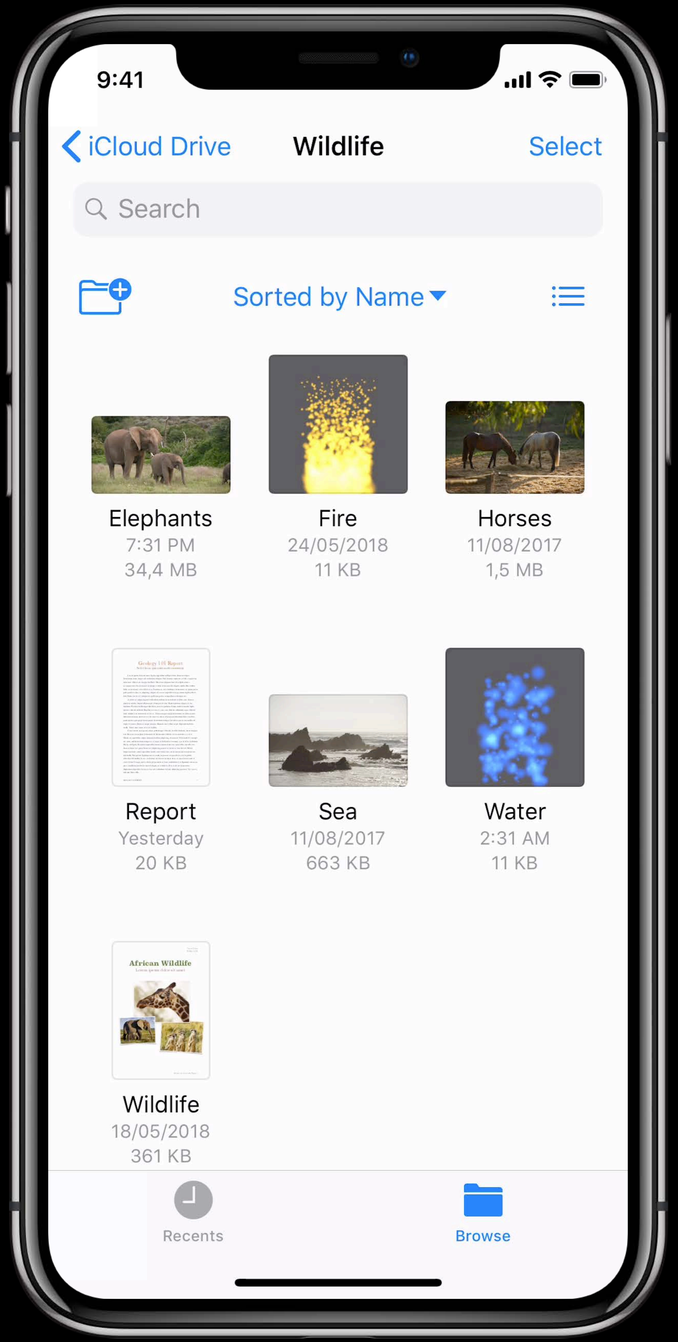


Providing Custom Previews

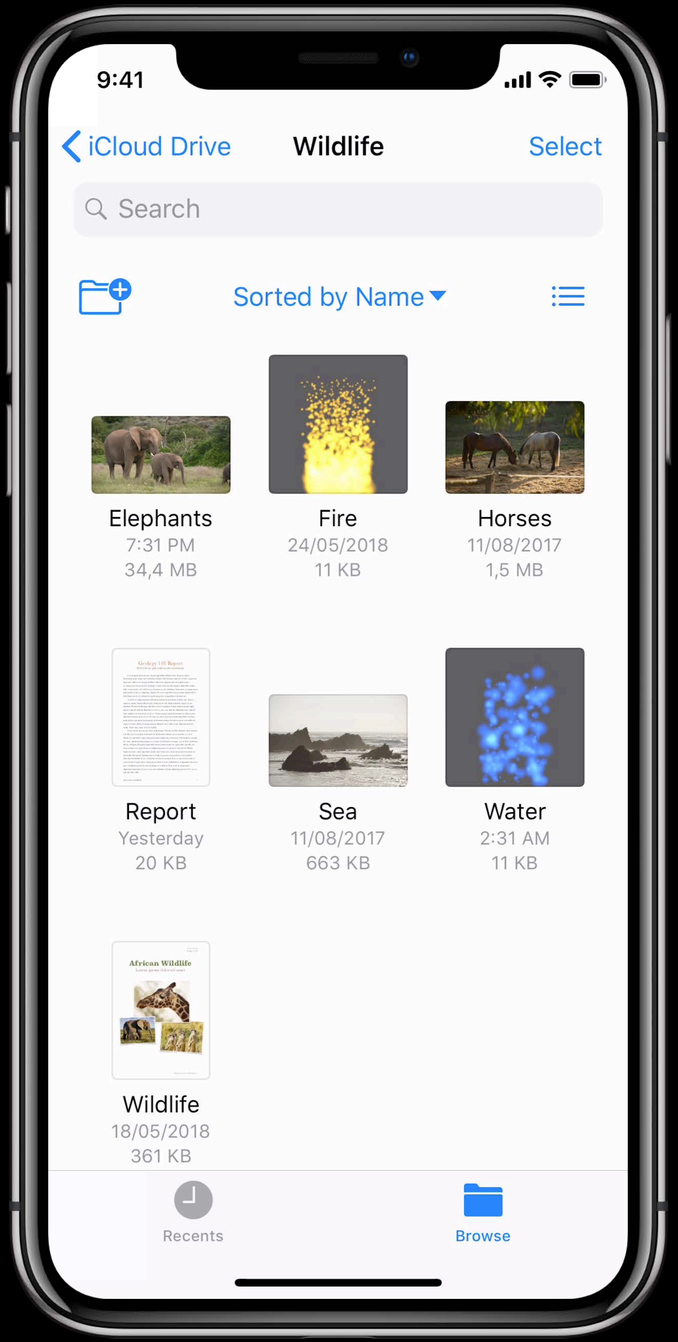


Providing Custom Thumbnails

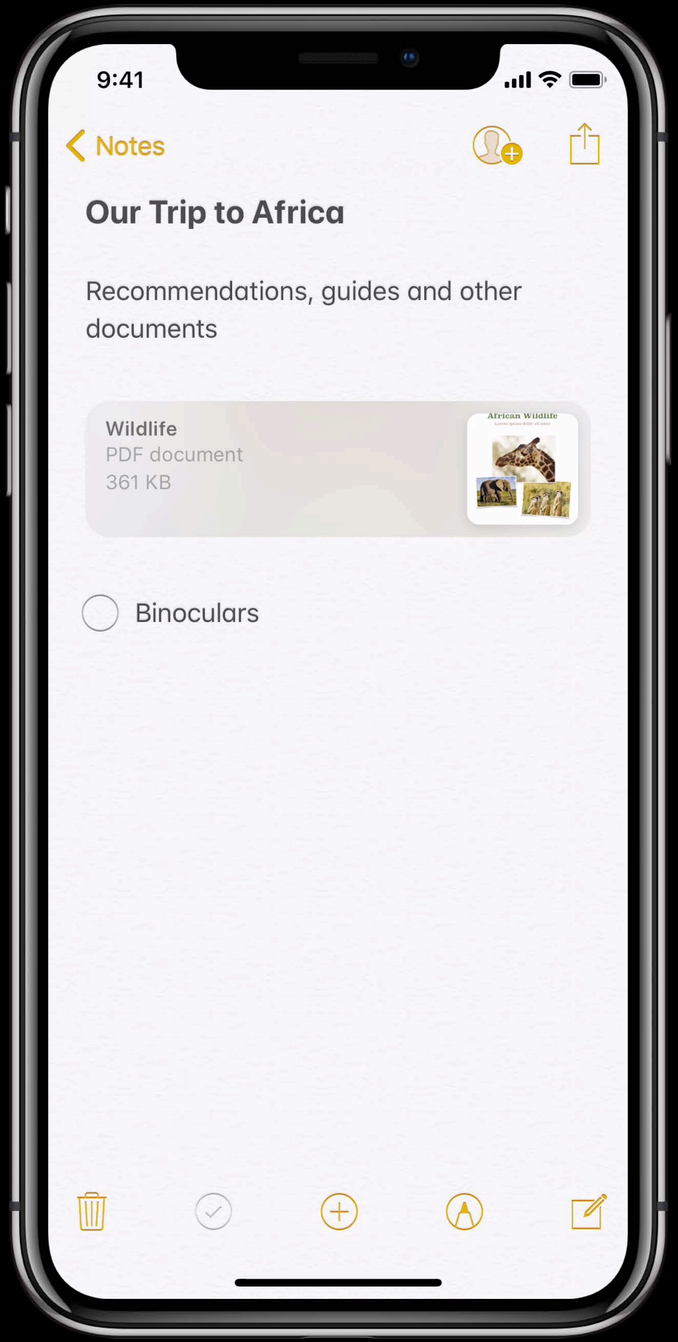
Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



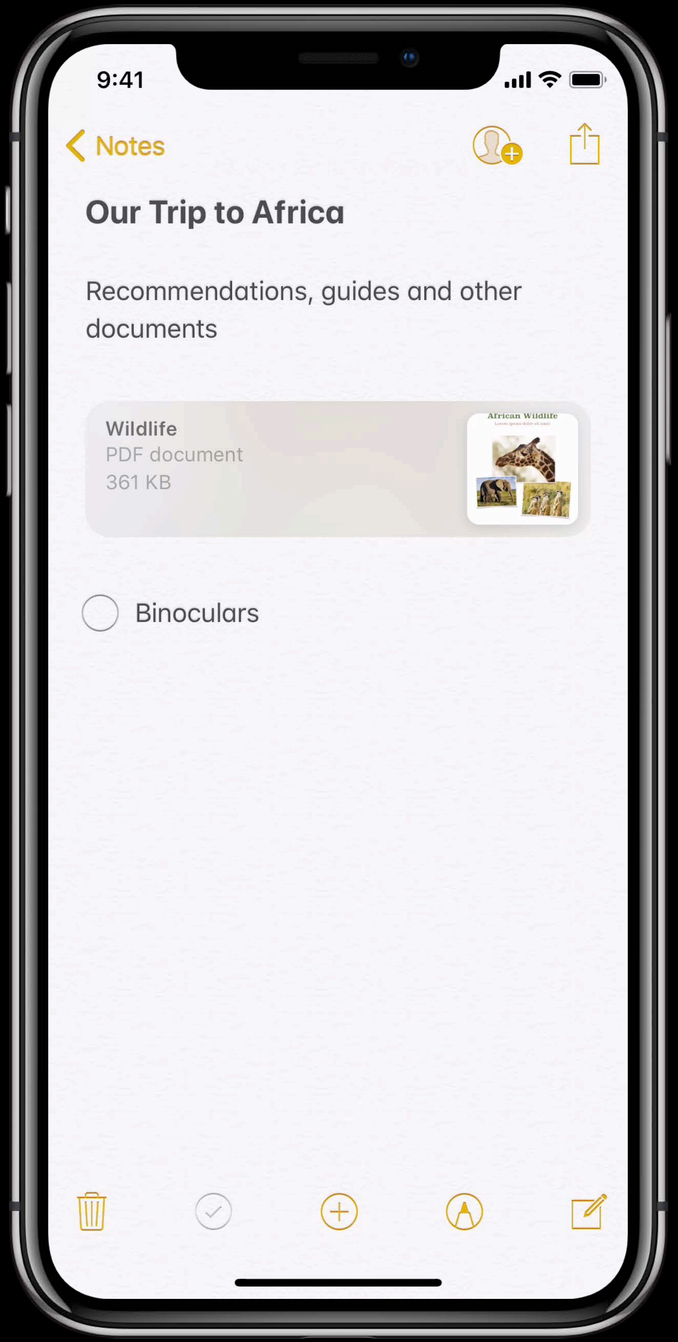
Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



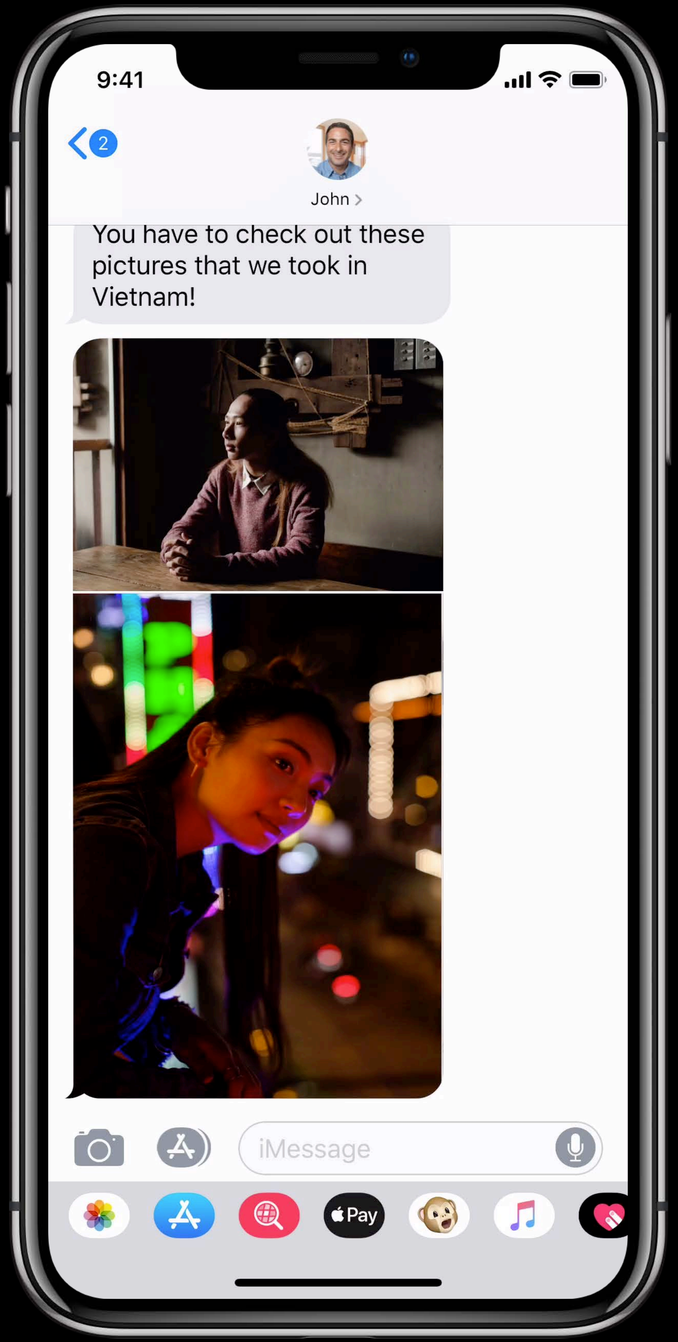
Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



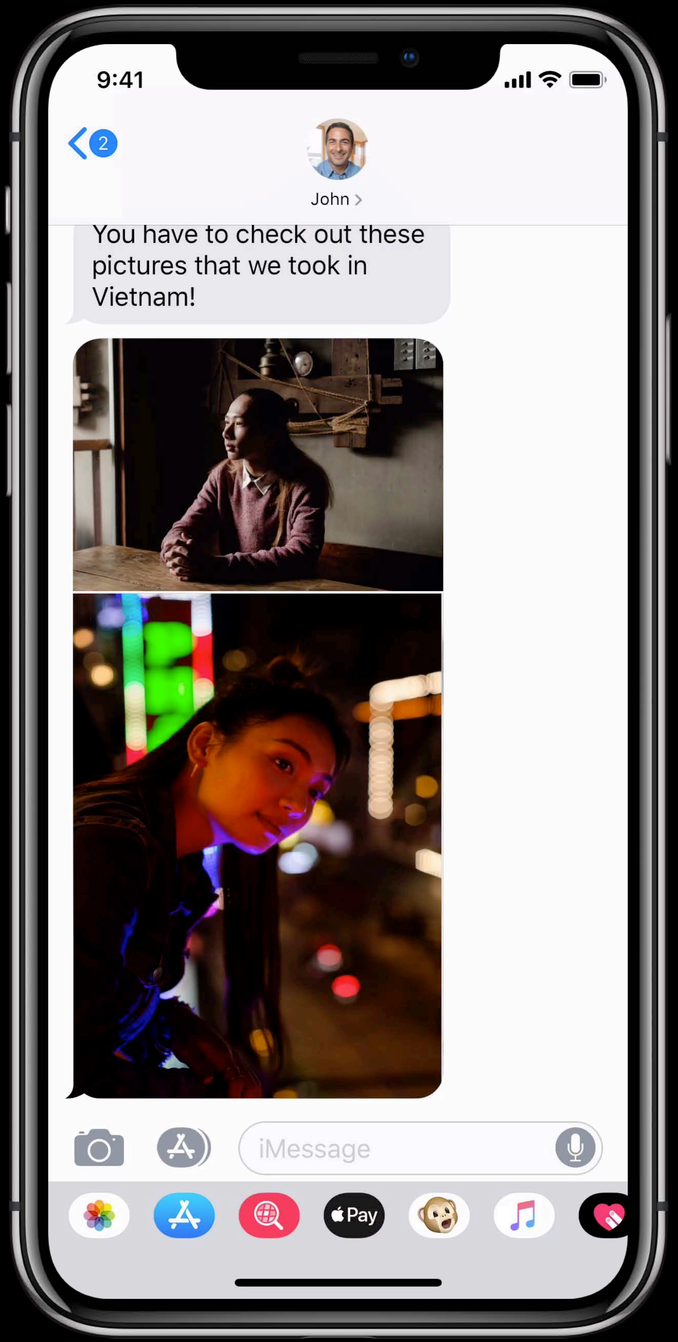
Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



Note: We muted the audio on the videos, but if it is intentional, you can turn the volume back up.



Introduction

Who is this session for?

Introduction

Who is this session for?

You want to know how to present documents without reinventing the wheel

Introduction

Who is this session for?

You want to know how to present documents without reinventing the wheel

You want your custom file format to be a first-class citizen in the system

Introduction

Who is this session for?

You want to know how to present documents without reinventing the wheel

You want your custom file format to be a first-class citizen in the system

You are new to iOS or already have an app

Introduction

Who is this session for?

You want to know how to present documents without reinventing the wheel

You want your custom file format to be a first-class citizen in the system

You are new to iOS or already have an app

Building Great Document-based Apps in iOS 11	Session 229	WWDC 2017
Managing Documents in your iOS Apps	Session 216	WWDC 2018

Introduction

Quick Look

Introduction

Quick Look

API introduced with iOS 4

Introduction

Quick Look

API introduced with iOS 4

View controller to preview documents

Introduction

Quick Look

API introduced with iOS 4

View controller to preview documents

Data source pattern

Introduction

Quick Look

API introduced with iOS 4

View controller to preview documents

Data source pattern

Delegate pattern

Introduction

Quick Look

API introduced with iOS 4

View controller to preview documents

Data source pattern

Delegate pattern

Easy view controller transitions

Introduction

Why using Quick Look?

Introduction

Why using Quick Look?

Works with most common file types

Introduction

Why using Quick Look?

Works with most common file types

Native user experience

Introduction

Why using Quick Look?

Works with most common file types

Native user experience

Optimized performance

Introduction

Why using Quick Look?

Works with most common file types

Native user experience

Optimized performance

Security is handled for you

Introduction

Supported file types

Images

Videos

Audio files

PDF documents

HTML documents

Rich Text Format (RTF) documents

Text files that conform to public.text

iWork and Office documents

ZIP archives

Introduction

Supported file types

Images

Videos

Audio files

PDF documents

HTML documents

Rich Text Format (RTF) documents

Text files that conform to public.text

iWork and Office documents

ZIP archives

File formats covered by 3rd Party Quick Look preview extensions

Introduction

Supported file types

NEW

- Images
- Videos
- Audio files
- PDF documents
- HTML documents
- Rich Text Format (RTF) documents
- Text files that conform to public.text
- iWork and Office documents
- ZIP archives
- File formats covered by 3rd Party Quick Look preview extensions
- USDZ

Introduction

User experience

Introduction

User experience

Pinch-to-zoom

Introduction

User experience

Pinch-to-zoom

Swipe-to-dismiss

Introduction

User experience

Pinch-to-zoom

Swipe-to-dismiss

Thumbnails for PDFs

Introduction

User experience

Pinch-to-zoom

Swipe-to-dismiss

Thumbnails for PDFs

Tabs in spreadsheets

Introduction

User experience

Pinch-to-zoom

Swipe-to-dismiss

Thumbnails for PDFs

Tabs in spreadsheets

Scrubber for audio and video files

Introduction

When to use Quick Look

Introduction

When to use Quick Look

You want to let the user zoom into photos

Introduction

When to use Quick Look

You want to let the user zoom into photos

Easily flick through a collection of photos

Introduction

When to use Quick Look

You want to let the user zoom into photos

Easily flick through a collection of photos

Play back an audio file or video

Introduction

When to use Quick Look

You want to let the user zoom into photos

Easily flick through a collection of photos

Play back an audio file or video

Present a user agreement document PDF

Introduction

When not to use Quick Look

Introduction

When not to use Quick Look

Introduction

When not to use Quick Look

Editing content	

Introduction

When not to use Quick Look

Editing content	Implement your own editor instead

Introduction

When not to use Quick Look

Editing content	Implement your own editor instead
Advanced video playback features	

Introduction

When not to use Quick Look

Editing content	Implement your own editor instead
Advanced video playback features	Use AVPlayer instead

Introduction

When not to use Quick Look

Editing content	Implement your own editor instead
Advanced video playback features	Use AVPlayer instead
Displaying contents inline (images, videos, etc.)	

Introduction

When not to use Quick Look

Editing content	Implement your own editor instead
Advanced video playback features	Use AVPlayer instead
Displaying contents inline (images, videos, etc.)	Use AVPlayer/UIImageView/WKWebView instead

Introduction

When not to use Quick Look

✗	✓
Editing content	Implement your own editor instead
Advanced video playback features	Use AVPlayer instead
Displaying contents inline (images, videos, etc.)	Use AVPlayer/UIImageView/WKWebView instead
Adding views on top of the preview	

Introduction

When not to use Quick Look

✗	✓
Editing content	Implement your own editor instead
Advanced video playback features	Use AVPlayer instead
Displaying contents inline (images, videos, etc.)	Use AVPlayer/UIImageView/WKWebView instead
Adding views on top of the preview	Not supported

Introduction to Quick Look

Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Providing custom thumbnails

Getting Started

Setting up a QLPreviewController

This looks ok to us. Adjusted centered positioning. Let us know if there was something specific you wanted us to fix.

```
let previewController = QLPreviewController()
previewController.dataSource = dataSource
present(previewController, animated: true, completion: nil)
```

Getting Started

Setting up a QLPreviewController

This looks ok to us. Adjusted centered positioning. Let us know if there was something specific you wanted us to fix.

```
let previewController = QLPreviewController()
previewController.dataSource = dataSource
present(previewController, animated: true, completion: nil)
```

Getting Started

Setting up a QLPreviewController

This looks ok to us. Adjusted centered positioning. Let us know if there was something specific you wanted us to fix.

```
let previewController = QLPreviewController()
previewController.dataSource = dataSource
present(previewController, animated: true, completion: nil)
```

Getting Started

Setting up a QLPreviewController

This looks ok to us. Adjusted centered positioning. Let us know if there was something specific you wanted us to fix.

```
let previewController = QLPreviewController()
previewController.dataSource = dataSource
present(previewController, animated: true, completion: nil)
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
protocol QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int
    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem
}
```

```
protocol QLPreviewItem {
    var previewItemURL: URL? { get }
    optional var previewItemTitle: String? { get }
}
```

```
extension NSURL: QLPreviewItem { }
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Providing Items

QLPreviewControllerDataSource and QLPreviewItem protocol

```
extension MyViewController: QLPreviewControllerDataSource {
    func numberOfPreviewItems(in controller: QLPreviewController) -> Int {
        return urlList.count
    }

    func previewController(_ controller: QLPreviewController,
                           previewItemAt index: Int) -> QLPreviewItem {
        return urlList[index]
    }
}
```

Presenting Quick Look

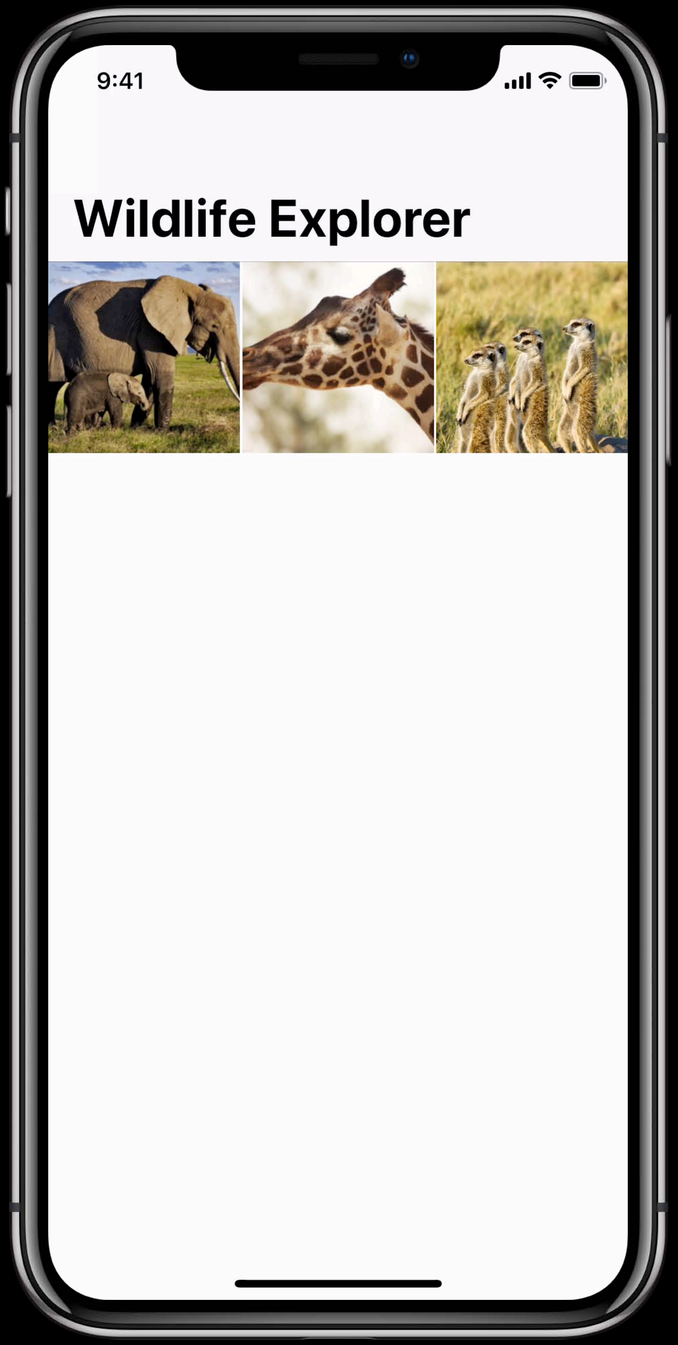
Presenting a QLPreviewViewController modally

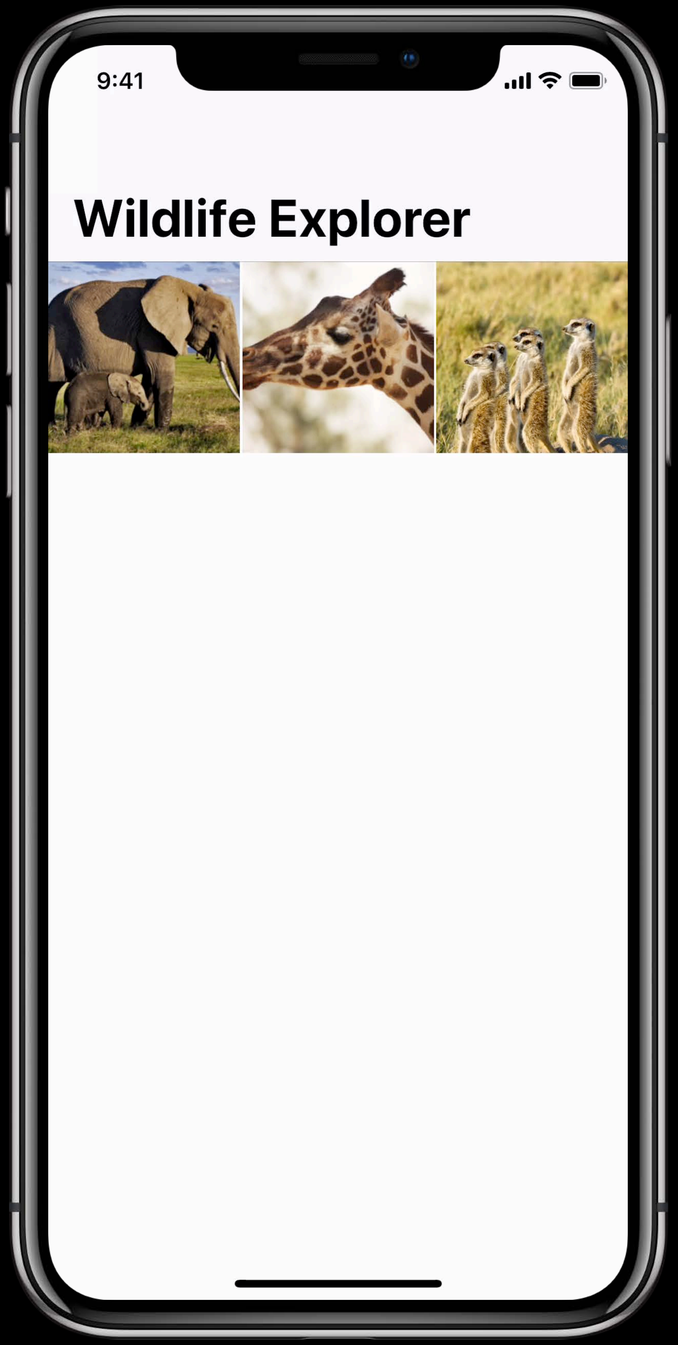
```
present(previewController, animated: true, completion: nil)
```

Presenting Quick Look

Presenting a QLPreviewViewController modally

```
present(previewController, animated: true, completion: nil)
```


9:41
Wildlife Explorer


9:41
Wildlife Explorer

Presenting Quick Look

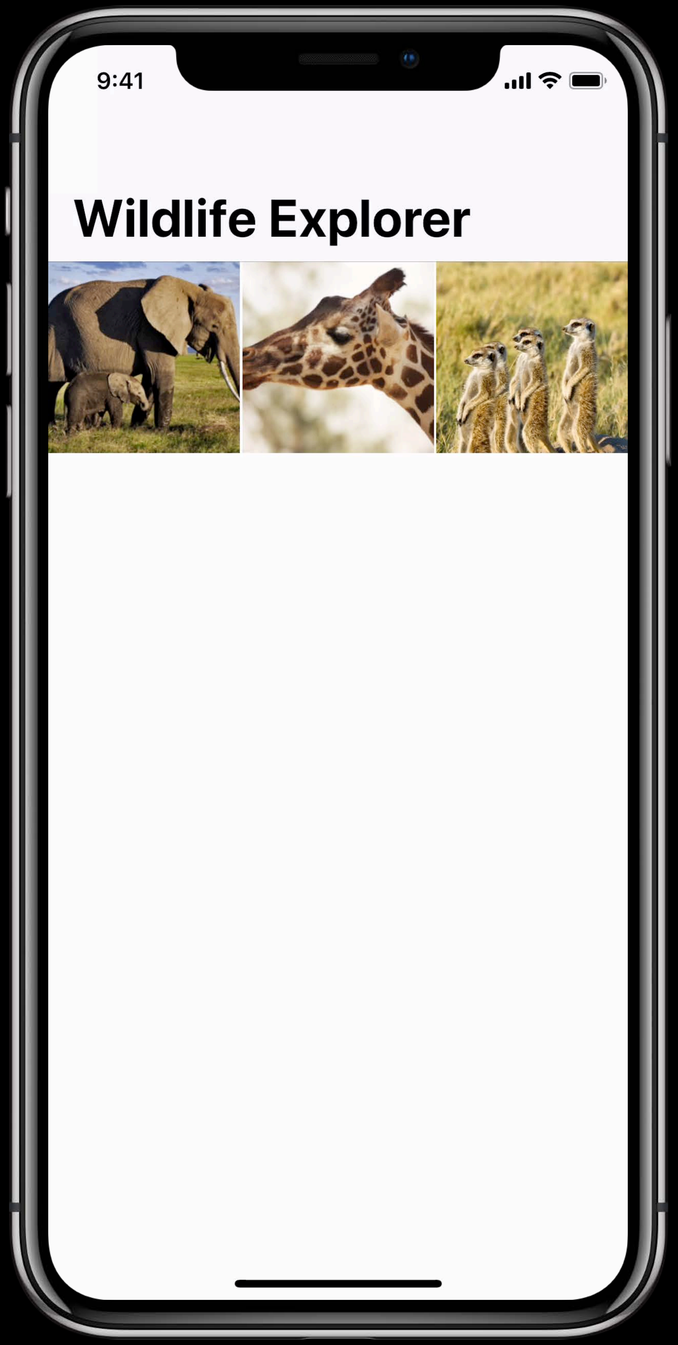
Pushing a QLPreviewViewController into UINavigationController

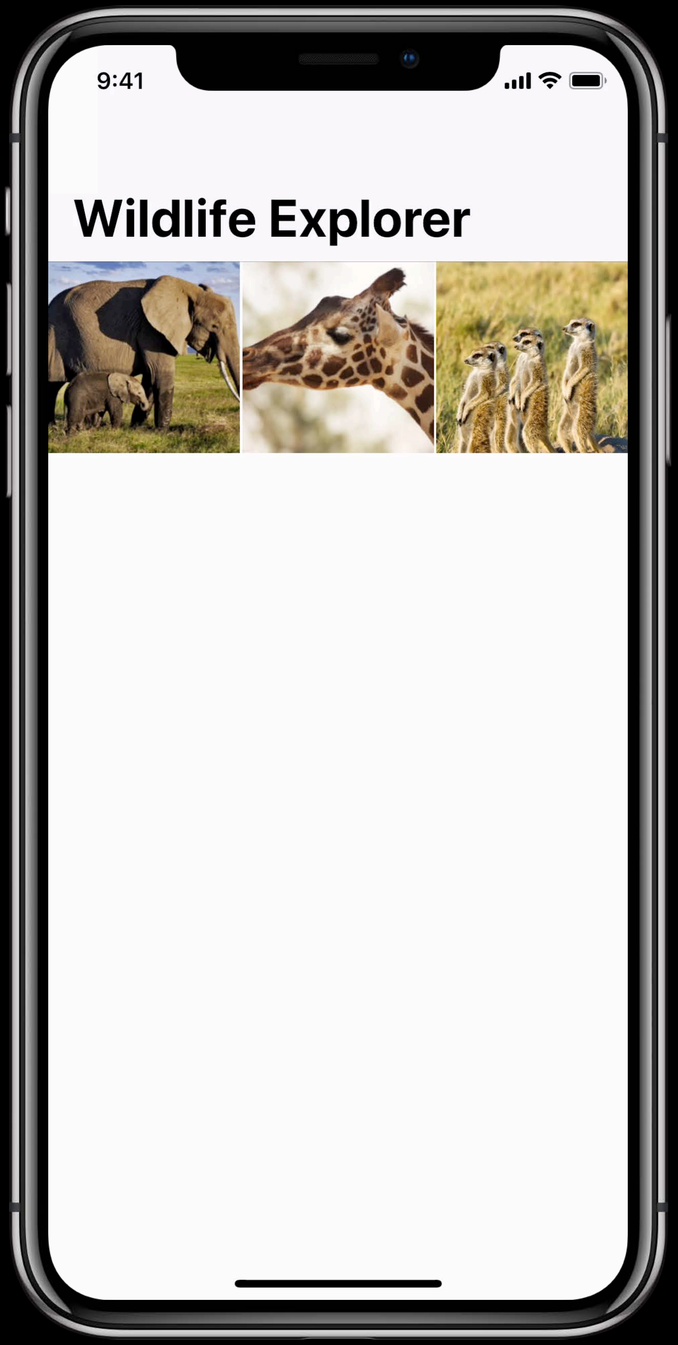
```
navigationController.pushViewController(previewController, animated: true)
```

Presenting Quick Look

Pushing a QLPreviewViewController into UINavigationController

```
navigationController.pushViewController(previewController, animated: true)
```


9:41
Wildlife Explorer


9:41
Wildlife Explorer

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

```
class QLPreviewController : UIViewController {

    class func canPreview(_ item: QLPreviewItem) -> Bool

    weak var dataSource: QLPreviewControllerDataSource?

    func reloadData()

    var currentPreviewItemIndex: Int

    weak var delegate: QLPreviewControllerDelegate?
}
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewControllerWillDismiss(_ controller: QLPreviewController)

optional func previewControllerDidDismiss(_ controller: QLPreviewController)
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewControllerWillDismiss(_ controller: QLPreviewController)

optional func previewControllerDidDismiss(_ controller: QLPreviewController)
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewControllerWillDismiss(_ controller: QLPreviewController)

optional func previewControllerDidDismiss(_ controller: QLPreviewController)
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewController(_ controller: QLPreviewController,
                                shouldOpen url: URL,
                                      for item: QLPreviewItem) -> Bool
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewController(_ controller: QLPreviewController,
                                shouldOpen url: URL,
                                      for item: QLPreviewItem) -> Bool
```

Controlling Quick Look

QLPreviewControllerDelegate protocol

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?

optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```

Controlling Quick Look

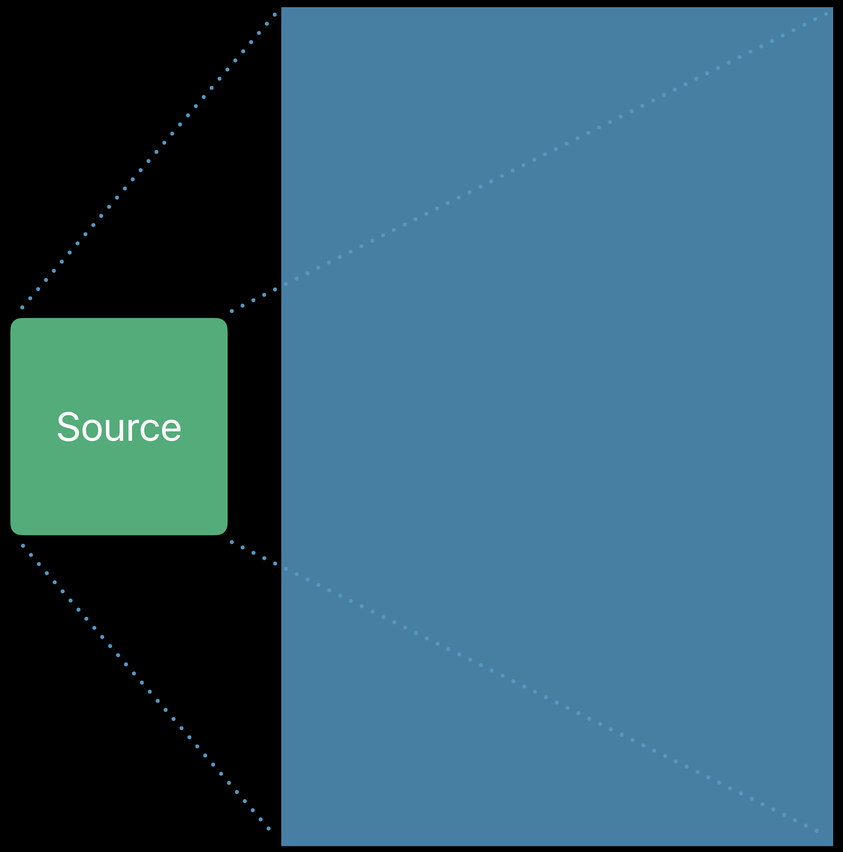
Animated transitions

Controlling Quick Look

Animated transitions

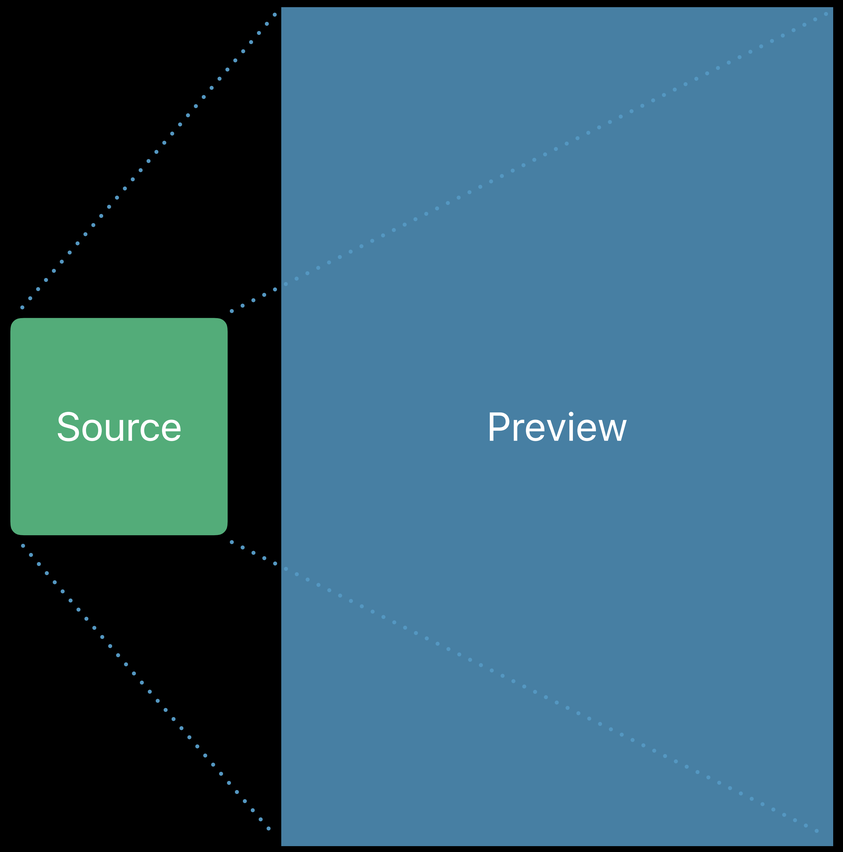
Controlling Quick Look

Animated transitions



Controlling Quick Look

Animated transitions



Controlling Quick Look

Animated transitions

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?


optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```

Controlling Quick Look

Animated transitions

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?


optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```

Controlling Quick Look

Animated transitions

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?


optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```

Controlling Quick Look

Animated transitions

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?


optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```

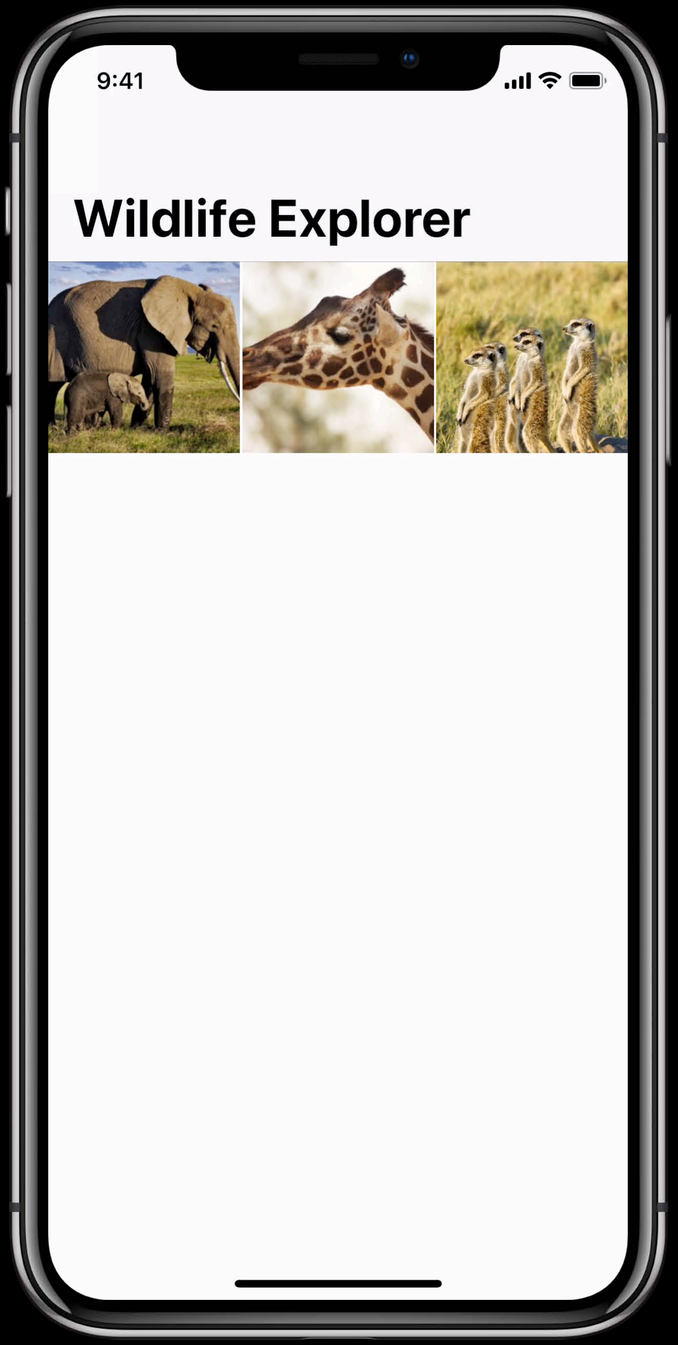
Controlling Quick Look

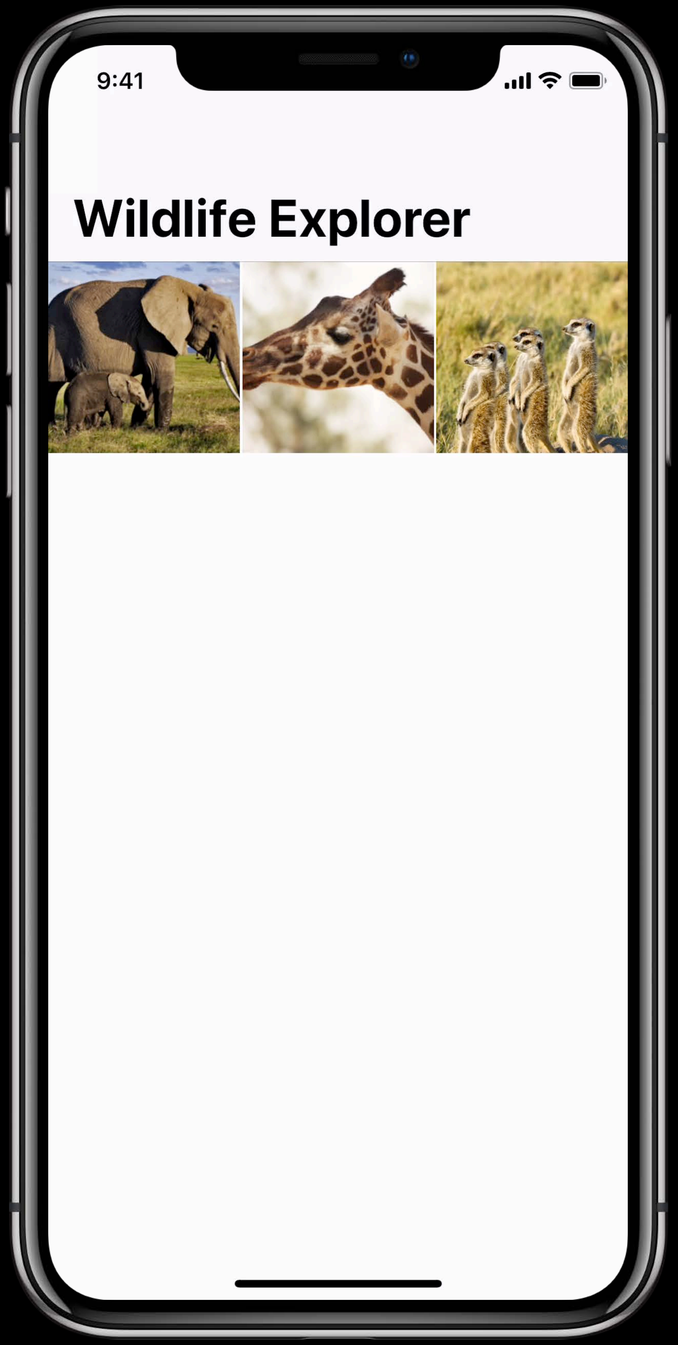
Animated transitions

```
optional func previewController(_ controller: QLPreviewController,
                                frameFor item: QLPreviewItem,
                                inSourceView view: AutoreleasingUnsafeMutablePointer<UIView?>)
                                -> CGRect

optional func previewController(_ controller: QLPreviewController,
                                transitionImageFor item: QLPreviewItem,
                                contentRect: UnsafeMutablePointer<CGRect>) -> UIImage?


optional func previewController(_ controller: QLPreviewController,
                                transitionViewFor item: QLPreviewItem) -> UIView?
```


9:41
Wildlife Explorer


9:41
Wildlife Explorer

Demo

Presenting Quick Look Preview Controller

Introduction to Quick Look

Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Providing custom thumbnails

Preview Extension

Introduction

Preview Extension

Introduction

Extends Quick Look's previewing capabilities

Preview Extension

Introduction

Extends Quick Look's previewing capabilities

Quick Look support for your custom file formats

Preview Extension

Introduction

Extends Quick Look's previewing capabilities

Quick Look support for your custom file formats

Great for shareable files

Preview Extension

Introduction

Extends Quick Look's previewing capabilities

Quick Look support for your custom file formats

Great for shareable files

QLPreviewController uses your extension


9:41
Done
1 of 3
Fire
Particles document


9:41
Done
1 of 3

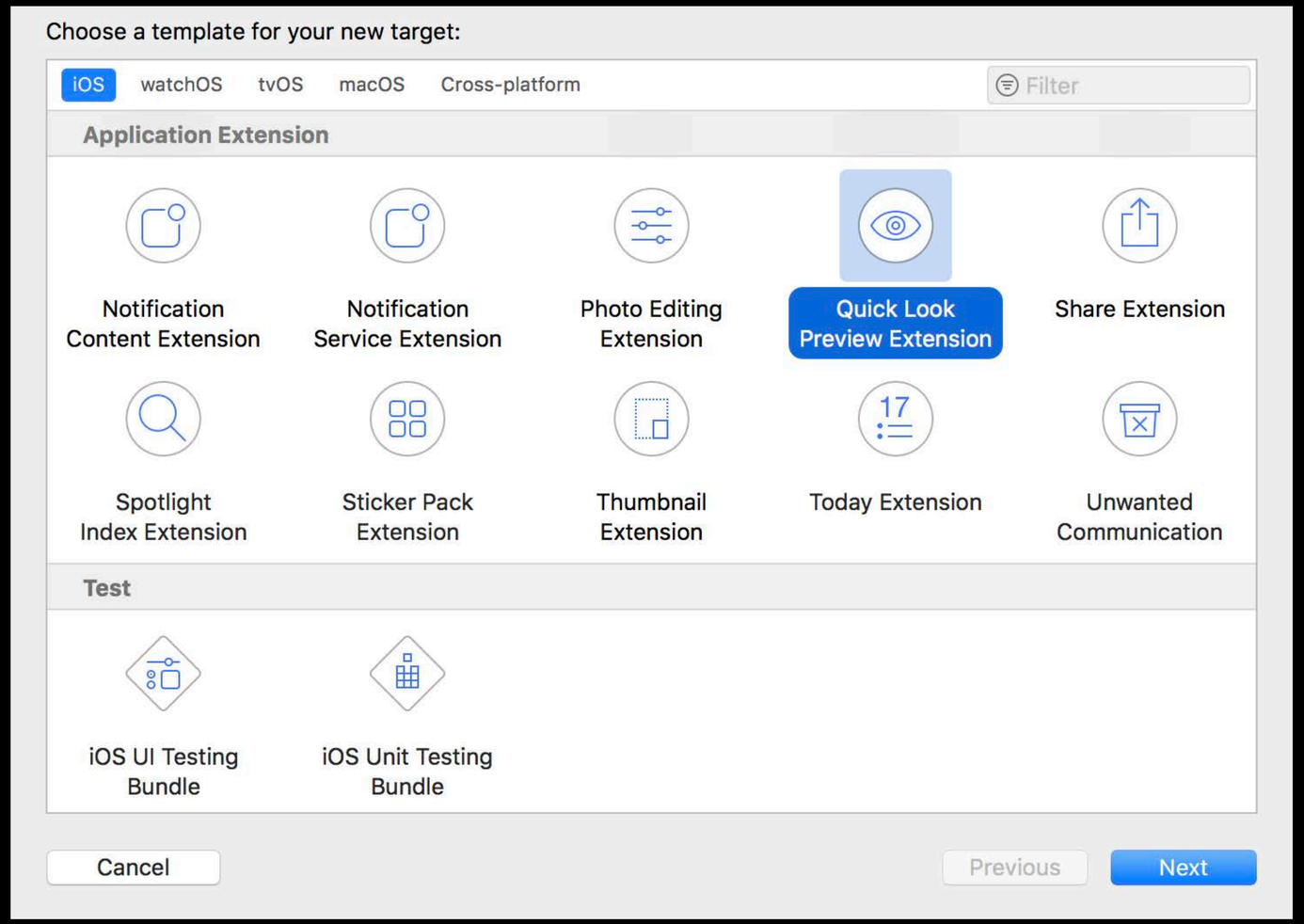

9:41
Done
1 of 3

Creating a Preview Extension

Step 1—Creation using Xcode template

Creating a Preview Extension

Step 1—Creation using Xcode template



Creating a Preview Extension

Template generated files

Creating a Preview Extension

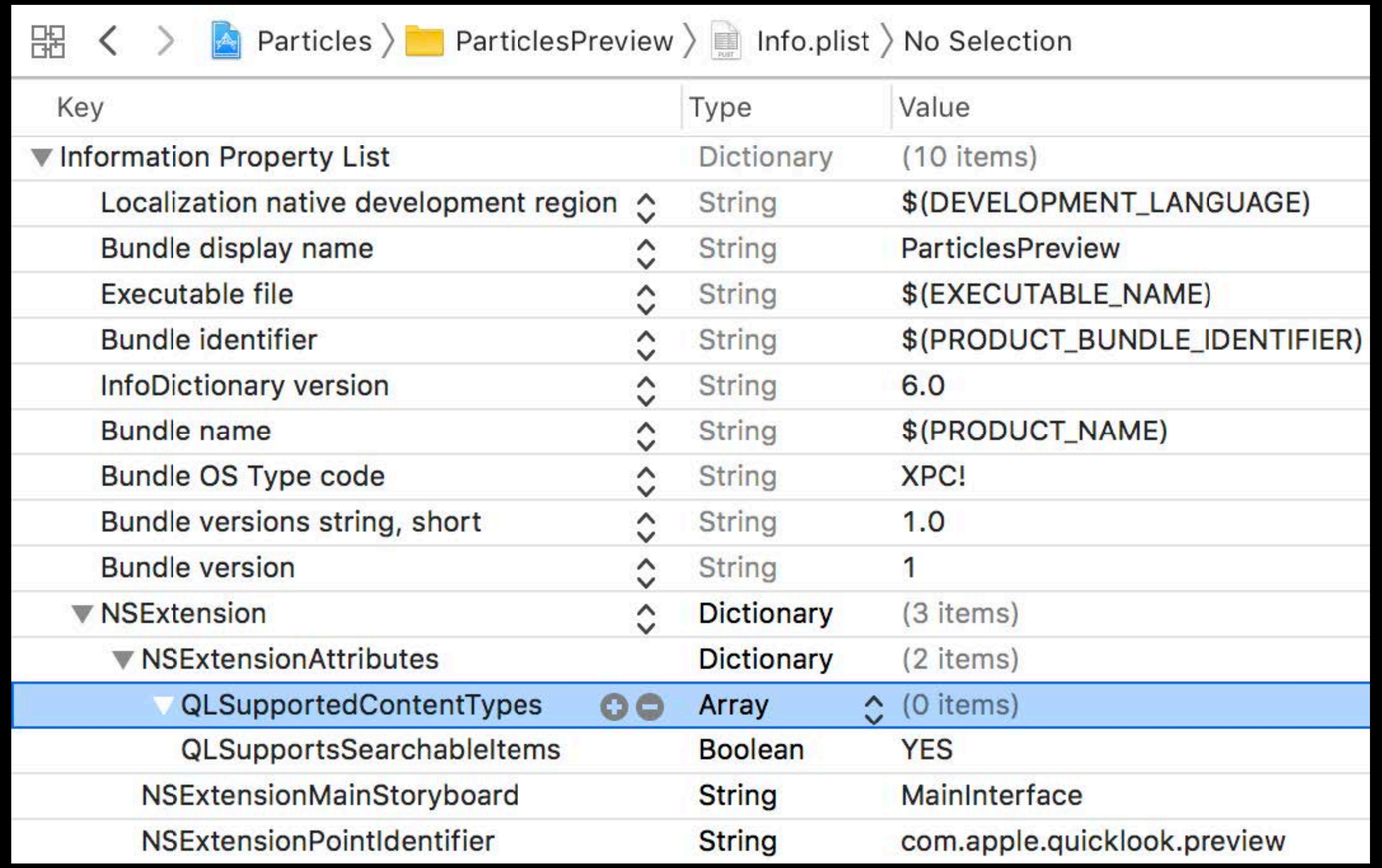
Template generated files

Creating a Preview Extension

Step 2—Editing the Info.plist

Creating a Preview Extension

Step 2—Editing the Info.plist

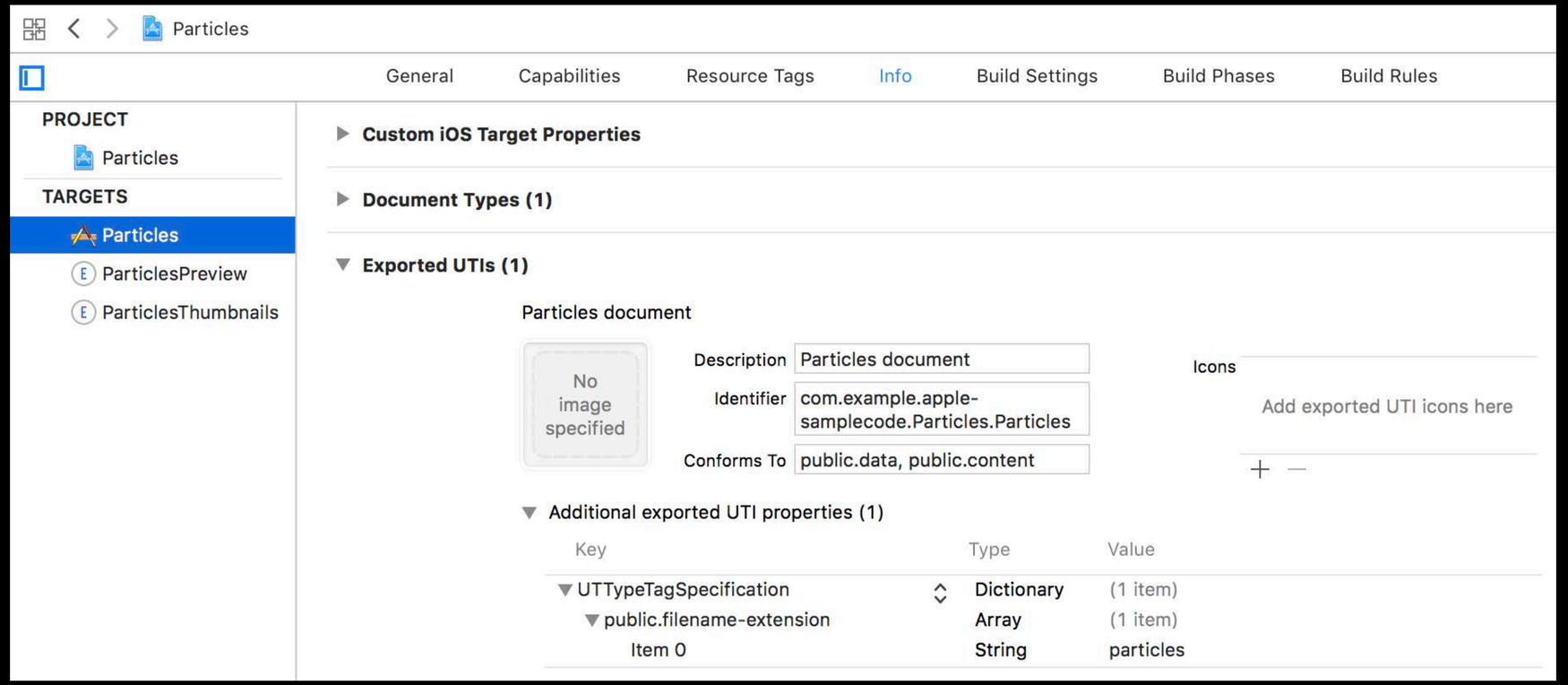


Creating a Preview Extension

Step 2—Editing the Info.plist

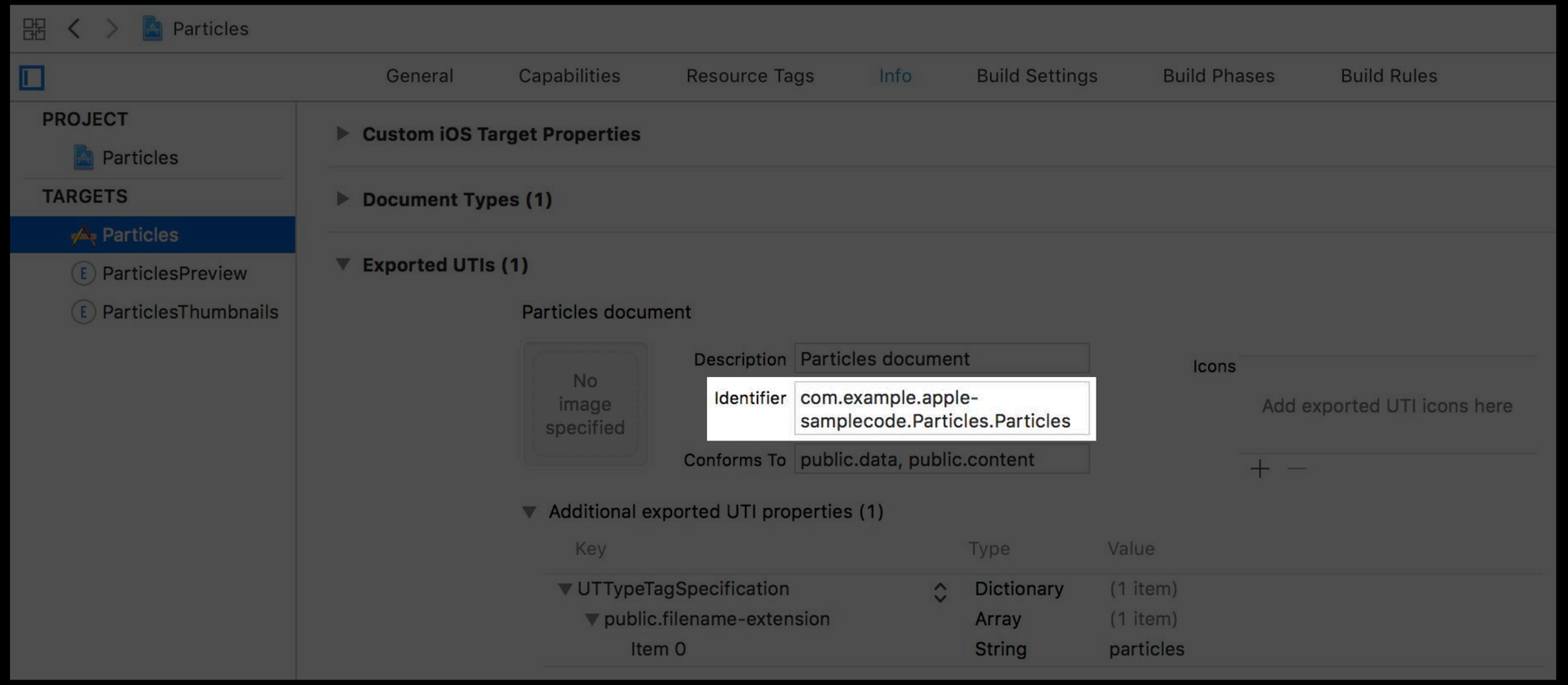
Creating a Preview Extension

Step 2—Editing the Info.plist



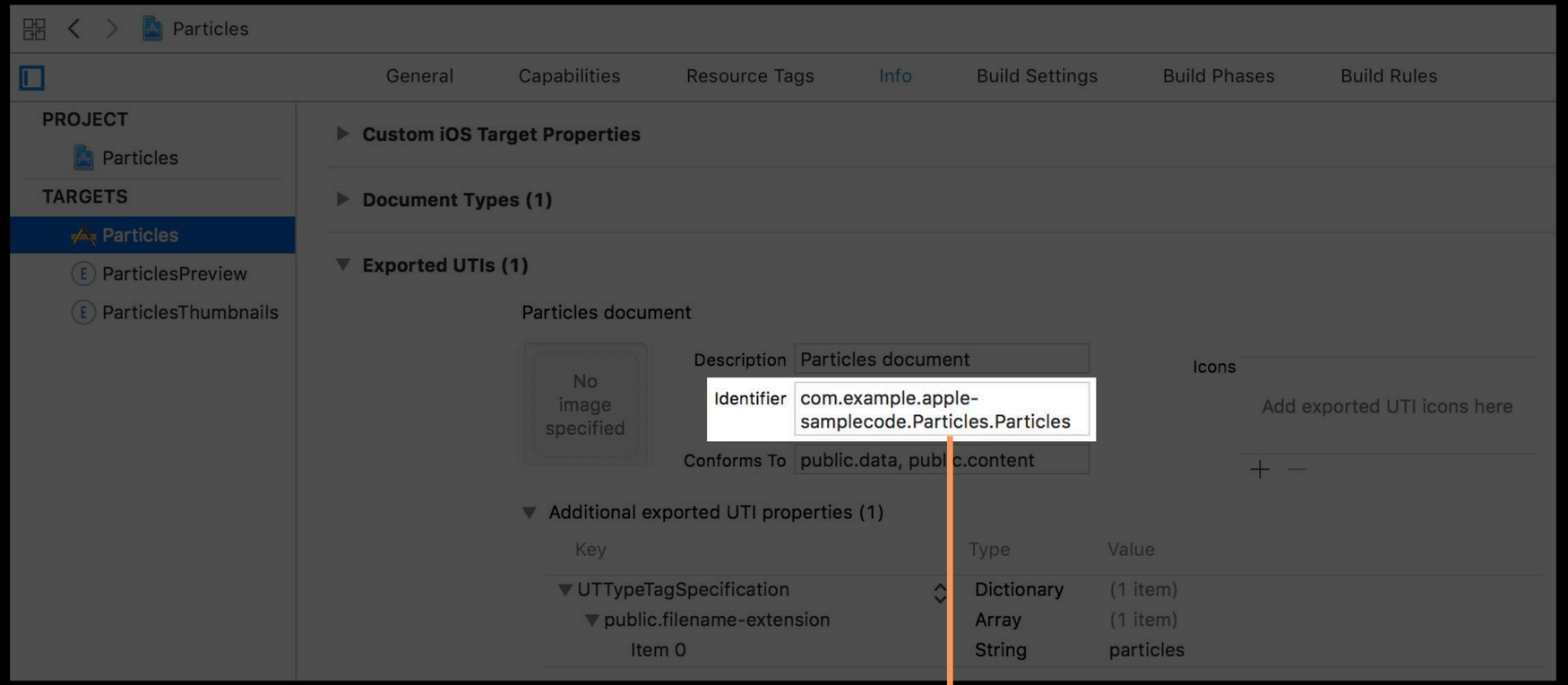
Creating a Preview Extension

Step 2—Editing the Info.plist



Creating a Preview Extension

Step 2—Editing the Info.plist



NSExtension	Dictionary	(3 items)
NSExtensionAttributes	Dictionary	(2 items)
QLSupportedContentTypes	Array	(1 item)
Item 0	String	com.example.apple-samplecode.Particles.Particles

Creating a Preview Extension

Step 3—QLPreviewingController protocol

```
class PreviewViewController: UIViewController, QLPreviewingController {

    func preparePreviewOfFile(at url: URL,
                              completionHandler handler: @escaping (Error?) -> Void) {

        // Load the preview and call the completion handler asynchronously when ready.
        loadPreviewOfFile(at: url, completionHandler: handler)
    }
}
```

Creating a Preview Extension

Step 3—QLPreviewingController protocol

```
class PreviewViewController: UIViewController, QLPreviewingController {

    func preparePreviewOfFile(at url: URL,
                              completionHandler handler: @escaping (Error?) -> Void) {

        // Load the preview and call the completion handler asynchronously when ready.
        loadPreviewOfFile(at: url, completionHandler: handler)
    }
}
```

Creating a Preview Extension

Step 3—QLPreviewingController protocol

```
class PreviewViewController: UIViewController, QLPreviewingController {

    func preparePreviewOfFile(at url: URL,
                              completionHandler handler: @escaping (Error?) -> Void) {

        // Load the preview and call the completion handler asynchronously when ready.
        loadPreviewOfFile(at: url, completionHandler: handler)
    }
}
```

Introduction to Quick Look

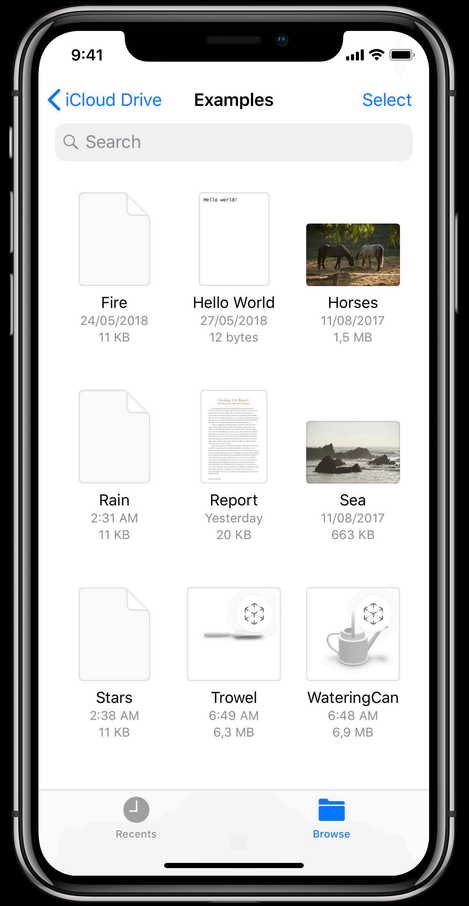
Adopting Quick Look Preview Controller

Providing custom Quick Look previews

Providing custom thumbnails

Thumbnail Extension

iOS native thumbnail generation support



Thumbnail Extension

iOS native thumbnail generation support

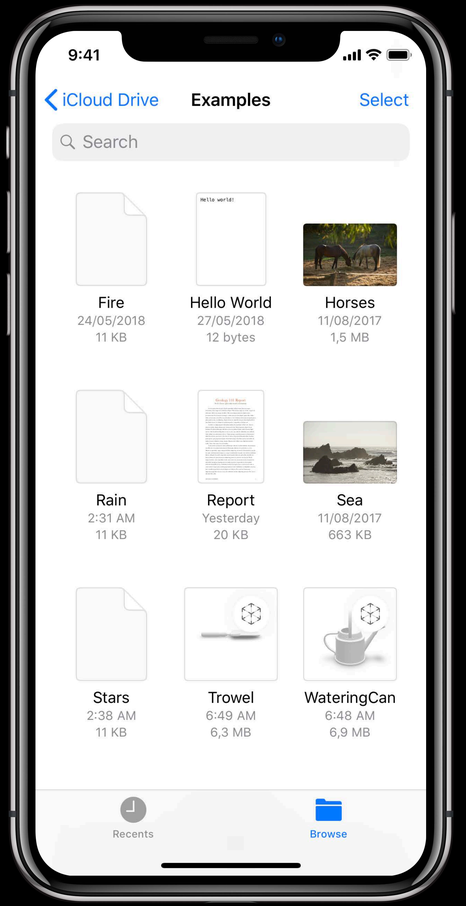
Images

Videos

PDF documents

Text files

USDZ



Thumbnail Extension

iOS native thumbnail generation support

Images

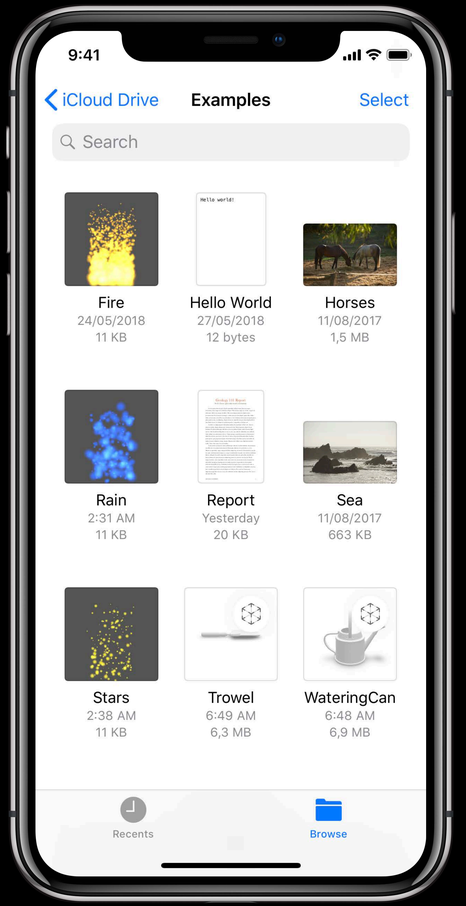
Videos

PDF documents

Text files

USDZ

Any file covered by 3rd Party
Quick Look thumbnail extensions



Thumbnail Extension

Visibility

Thumbnail Extension

Visibility

In the Files app and any UIDocumentBrowserViewController-based app

Thumbnail Extension

Visibility

In the Files app and any UIDocumentBrowserViewController-based app

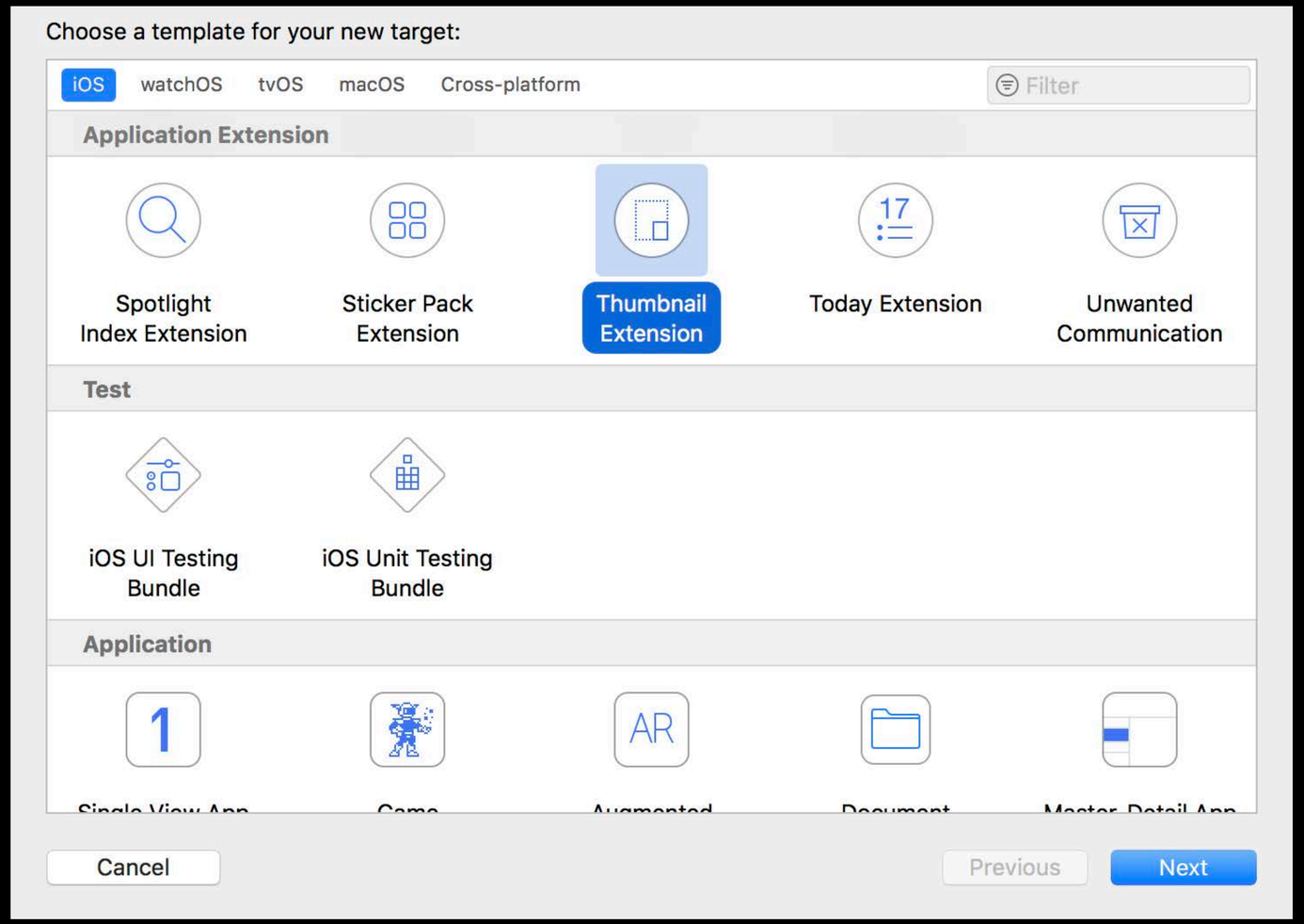
In Quick Look's list view

Creating a Thumbnail Extension

Step 1—Creation using Xcode template

Creating a Thumbnail Extension

Step 1—Creation using Xcode template

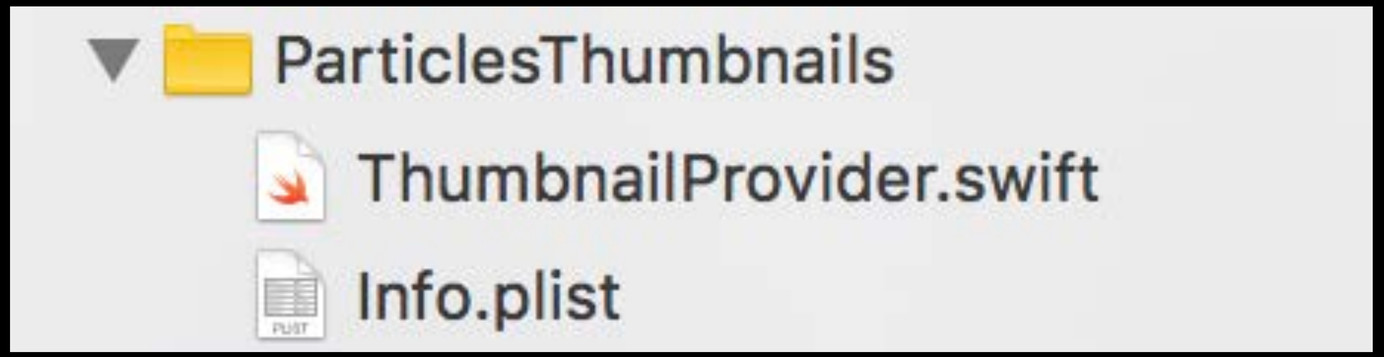


Creating a Thumbnail Extension

Template generated files

Creating a Thumbnail Extension

Template generated files

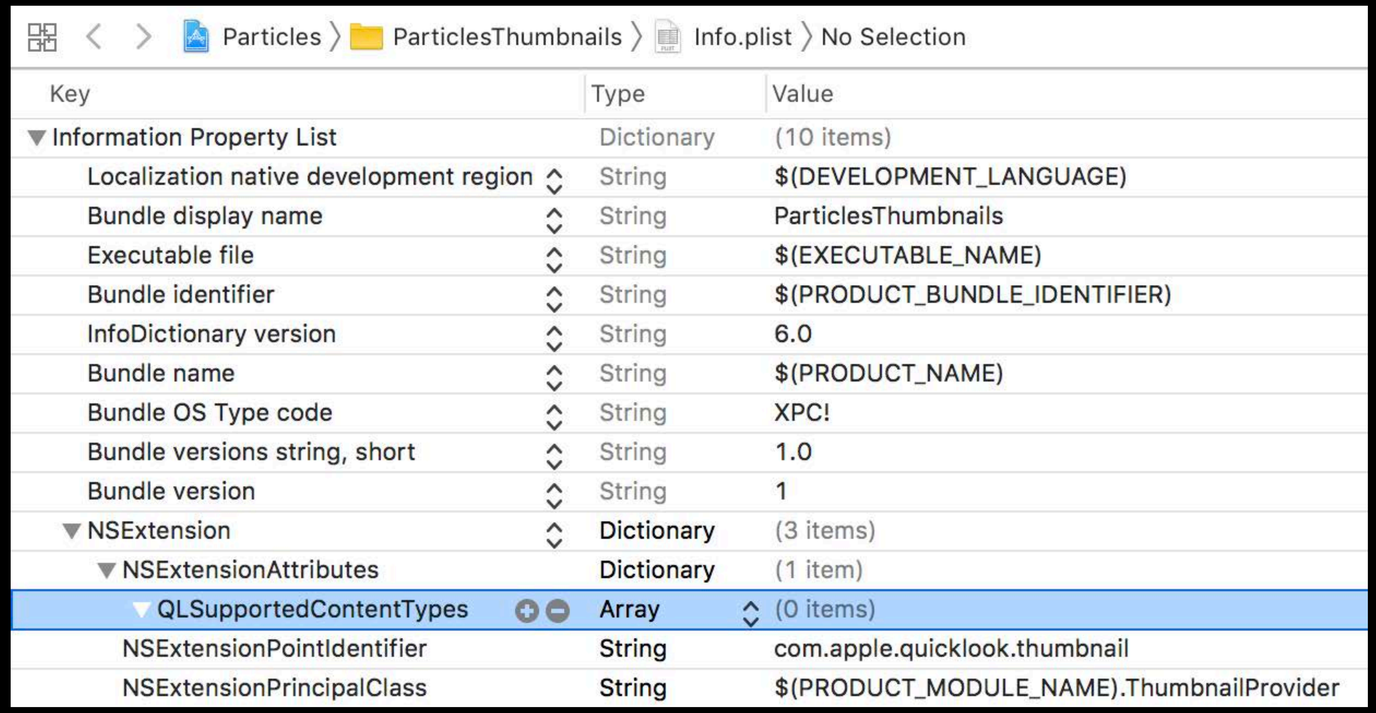


Creating a Thumbnail Extension

Step 2—Editing the Info.plist

Creating a Thumbnail Extension

Step 2—Editing the Info.plist

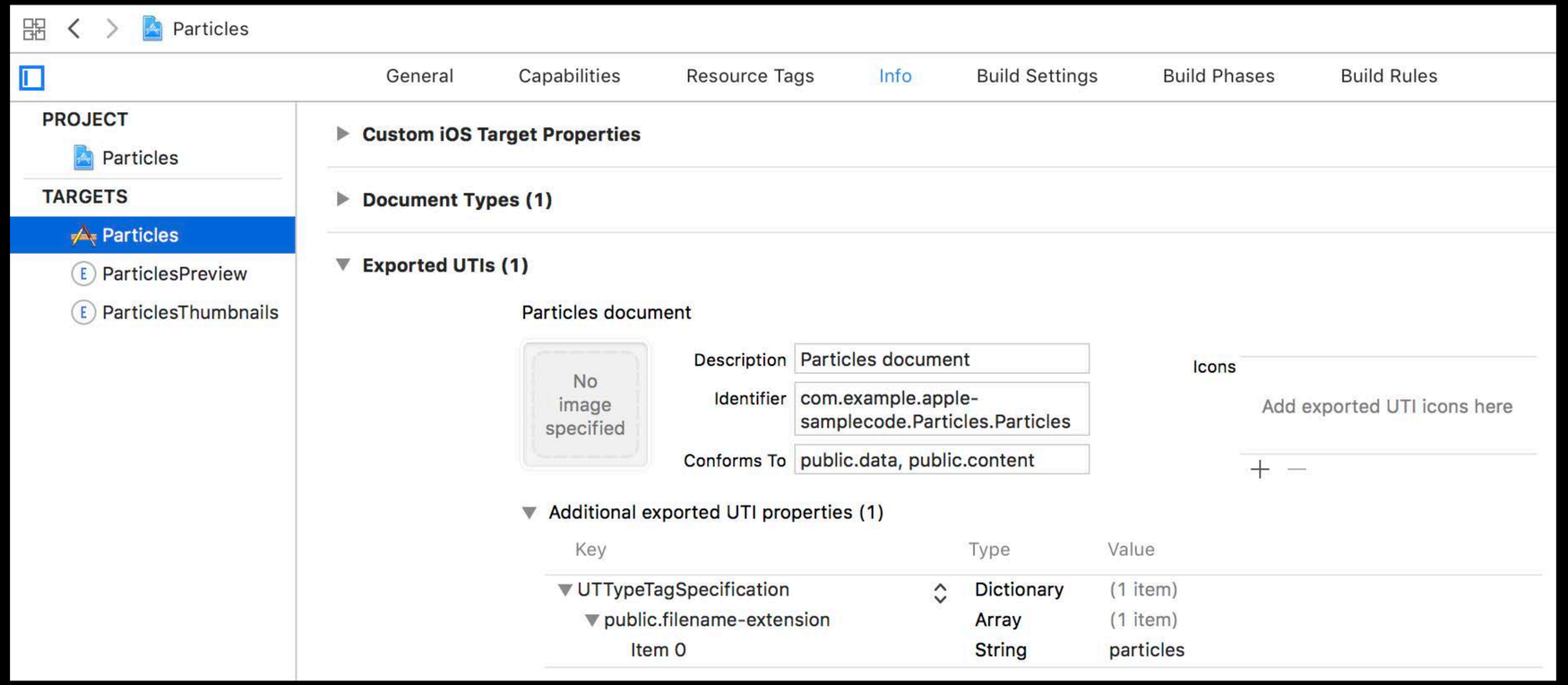


Creating a Thumbnail Extension

Step 2—Editing the Info.plist

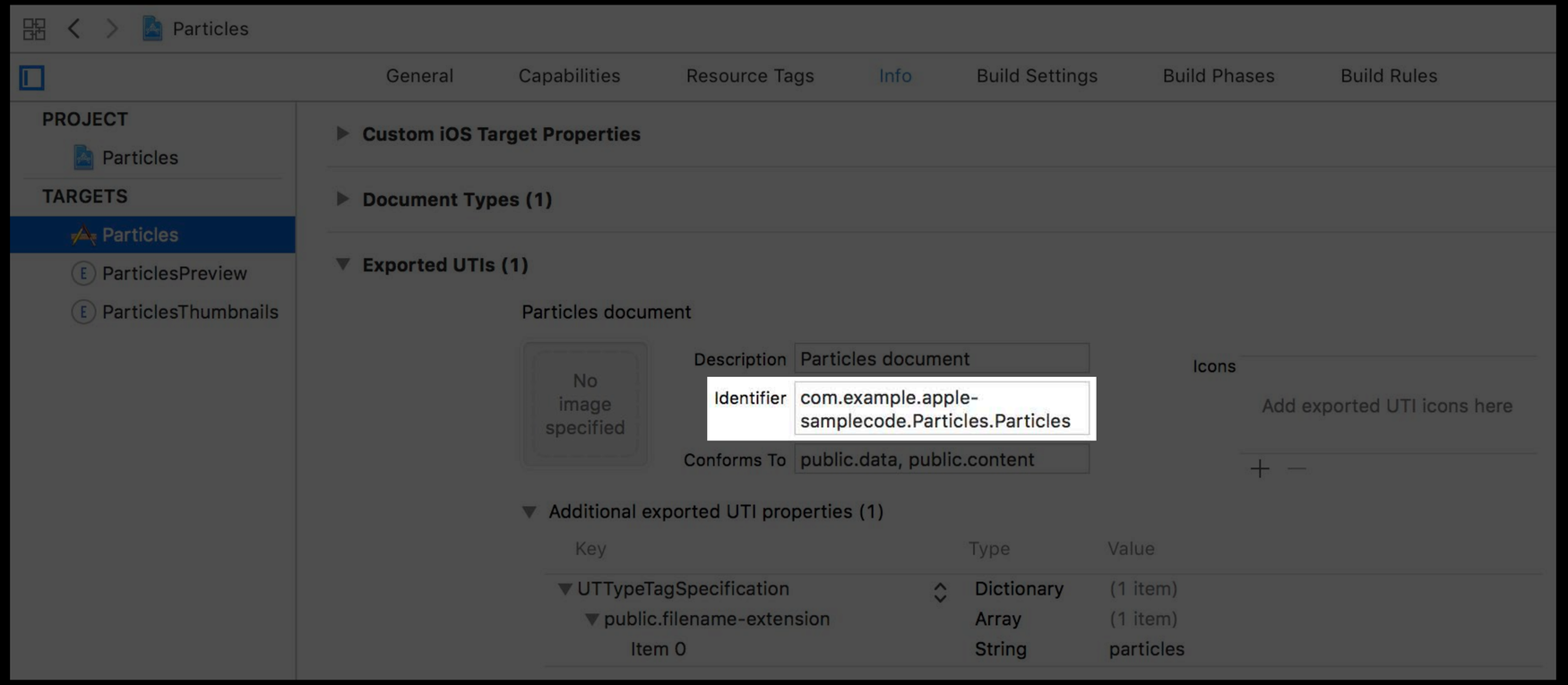
Creating a Thumbnail Extension

Step 2—Editing the Info.plist



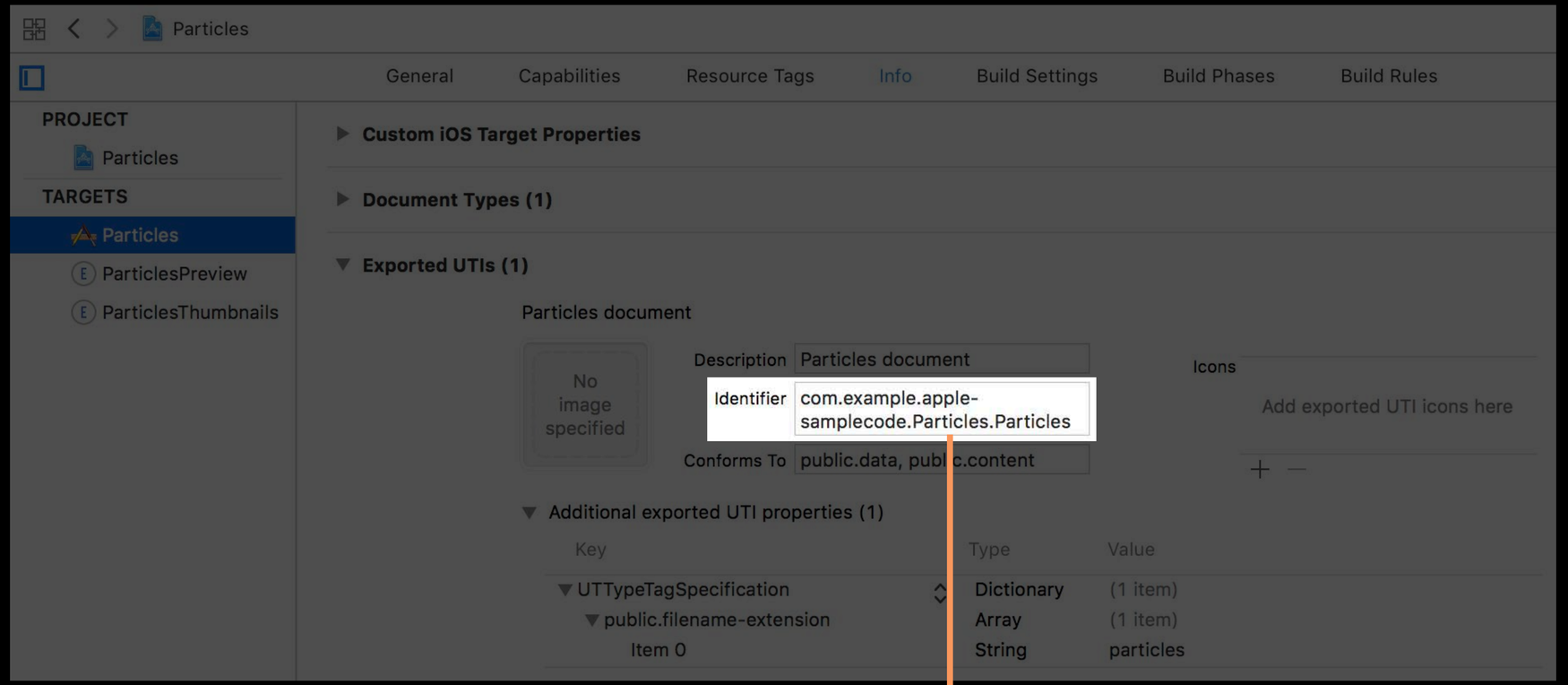
Creating a Thumbnail Extension

Step 2—Editing the Info.plist



Creating a Thumbnail Extension

Step 2—Editing the Info.plist



▼ NSExtension	Dictionary	(3 items)
▼ NSExtensionAttributes	Dictionary	(1 item)
▼ QLSupportedContentTypes	Array	(1 item)
Item 0	String	com.example.apple-samplecode.Particles.Particles

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

Draw thumbnails

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

Draw thumbnails

- Using CoreGraphics

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

Draw thumbnails

- Using CoreGraphics
- Using UIKit

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

Draw thumbnails

- Using CoreGraphics
- Using UIKit

Return an image file URL

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

```
class ThumbnailProvider: QLThumbnailProvider {

    override func provideThumbnail(for request: QLFileThumbnailRequest,
                                   _ handler: @escaping (QLThumbnailReply?, Error?) -> Void) {

        // Create a QLThumbnailReply for the request that will generate a thumbnail
        // and pass it through the handler.
        let reply = thumbnailReply(for: request)
        handler(reply, nil)
    }
}
```

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

```
class ThumbnailProvider: QLThumbnailProvider {

    override func provideThumbnail(for request: QLFileThumbnailRequest,
                                   _ handler: @escaping (QLThumbnailReply?, Error?) -> Void) {

        // Create a QLThumbnailReply for the request that will generate a thumbnail
        // and pass it through the handler.
        let reply = thumbnailReply(for: request)
        handler(reply, nil)
    }
}
```

Providing Thumbnails

Step 3—Implementing your QLThumbnailProvider subclass

```
class ThumbnailProvider: QLThumbnailProvider {

    override func provideThumbnail(for request: QLFileThumbnailRequest,
                                   _ handler: @escaping (QLThumbnailReply?, Error?) -> Void) {

        // Create a QLThumbnailReply for the request that will generate a thumbnail
        // and pass it through the handler.
        let reply = thumbnailReply(for: request)
        handler(reply, nil)
    }
}
```

Demo

Providing custom previews and thumbnails

Summary

Preview documents in your app with Quick Look easily

Extend Quick Look's capabilities system-wide with preview and thumbnail extensions

More Information

https://developer.apple.com/session237


WWDC18