Media

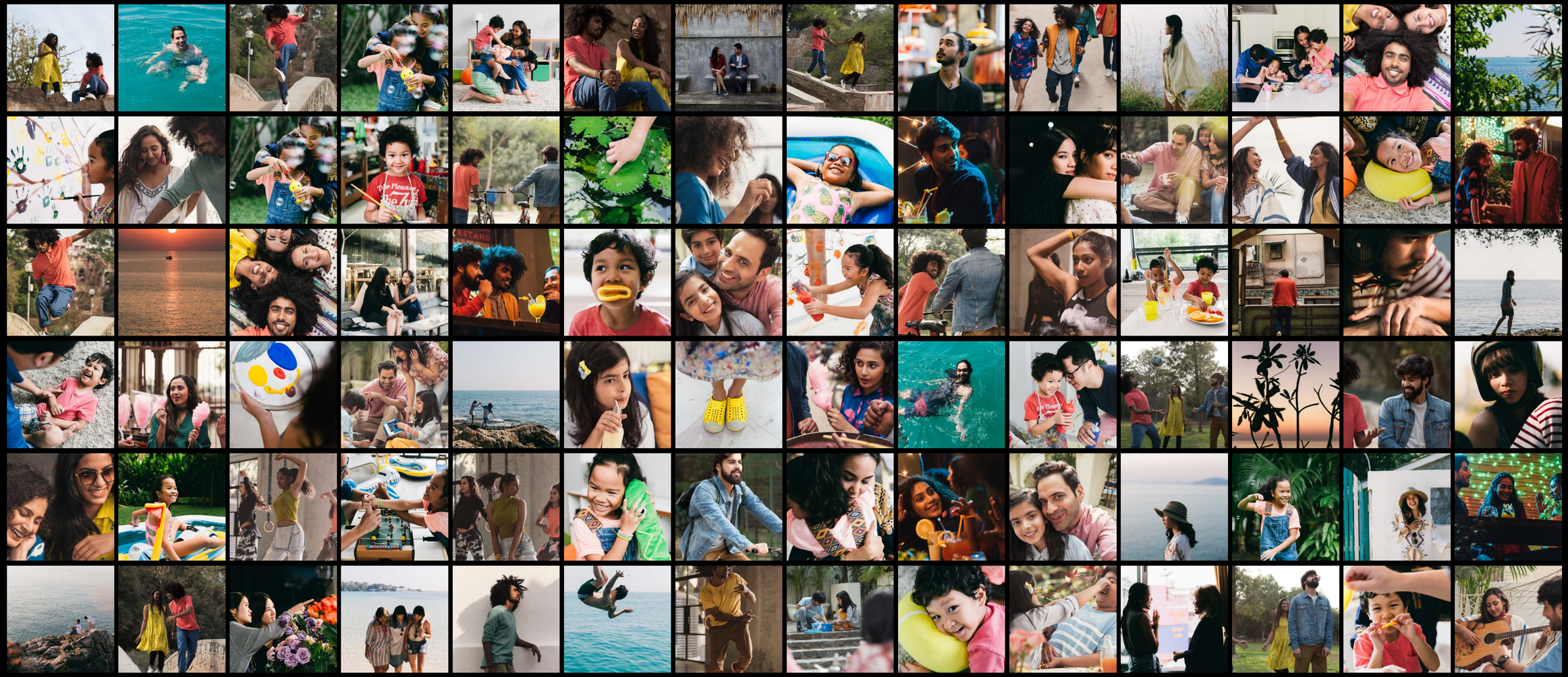
#WWDC17

What's New in Photos APIs

Session 505

Eric Hanson, Photos Platform Evangelist
Hasan Adil, Photos Engineer
Andreas Karlsson, Photos Engineer

© 2017 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.



Create great experiences

High-performance library access

Respecting user privacy

Image Picker Improvements

Easier access to photos from your iOS app

Hasan Adil, Photos Engineer

9:41 AM

Photos Cancel

Moments

Camera Roll
132

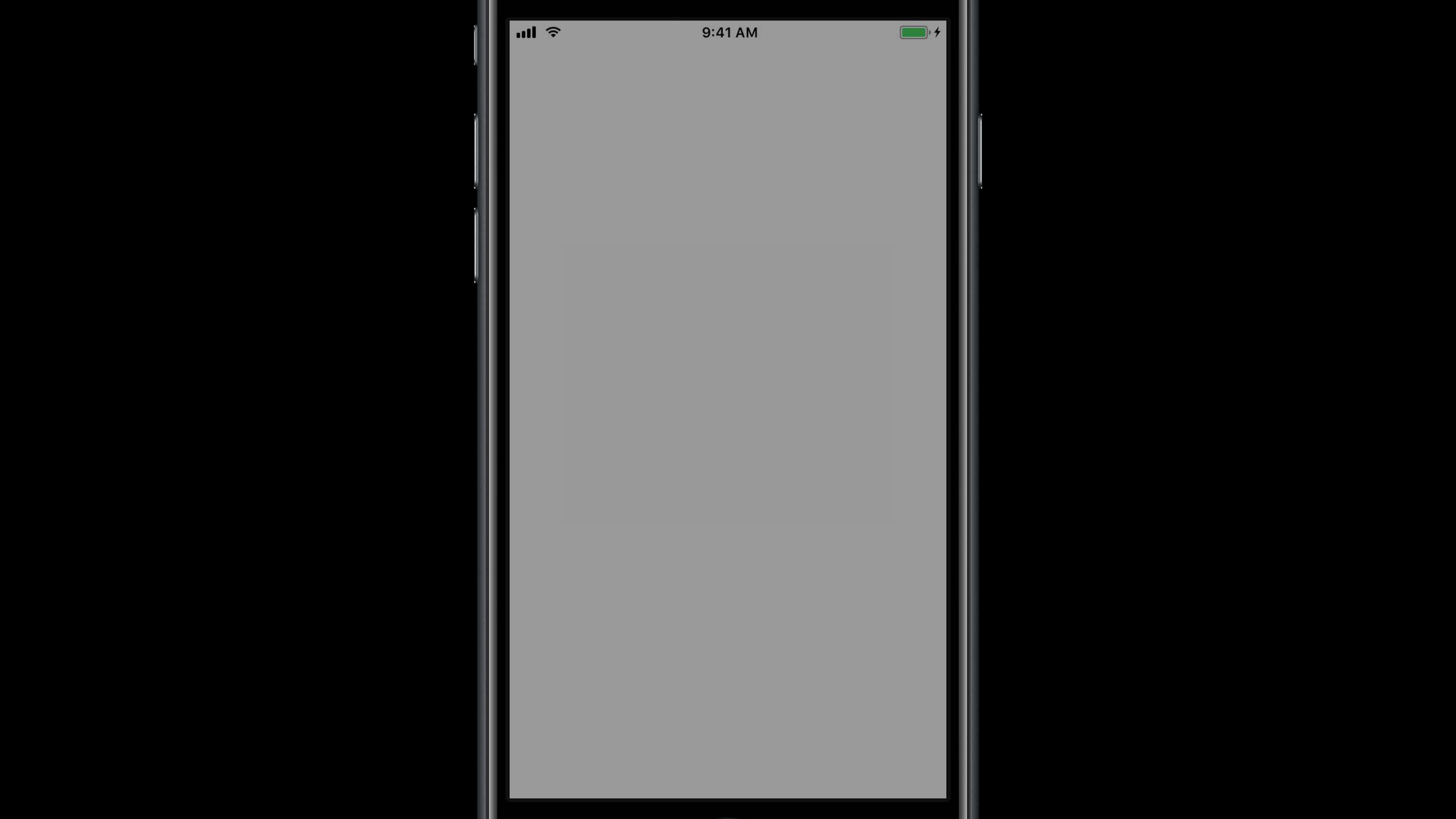
My Albums

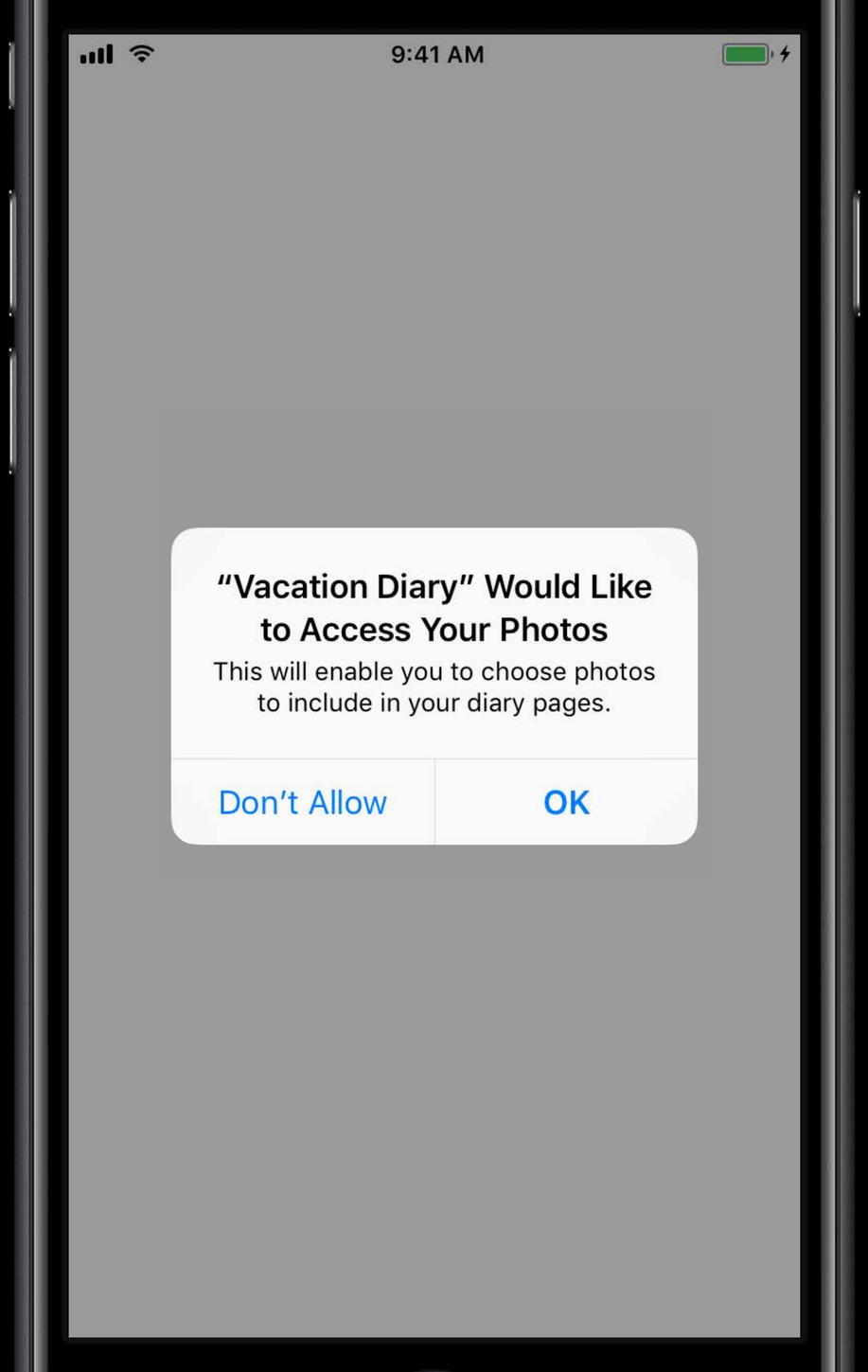
Friends
6

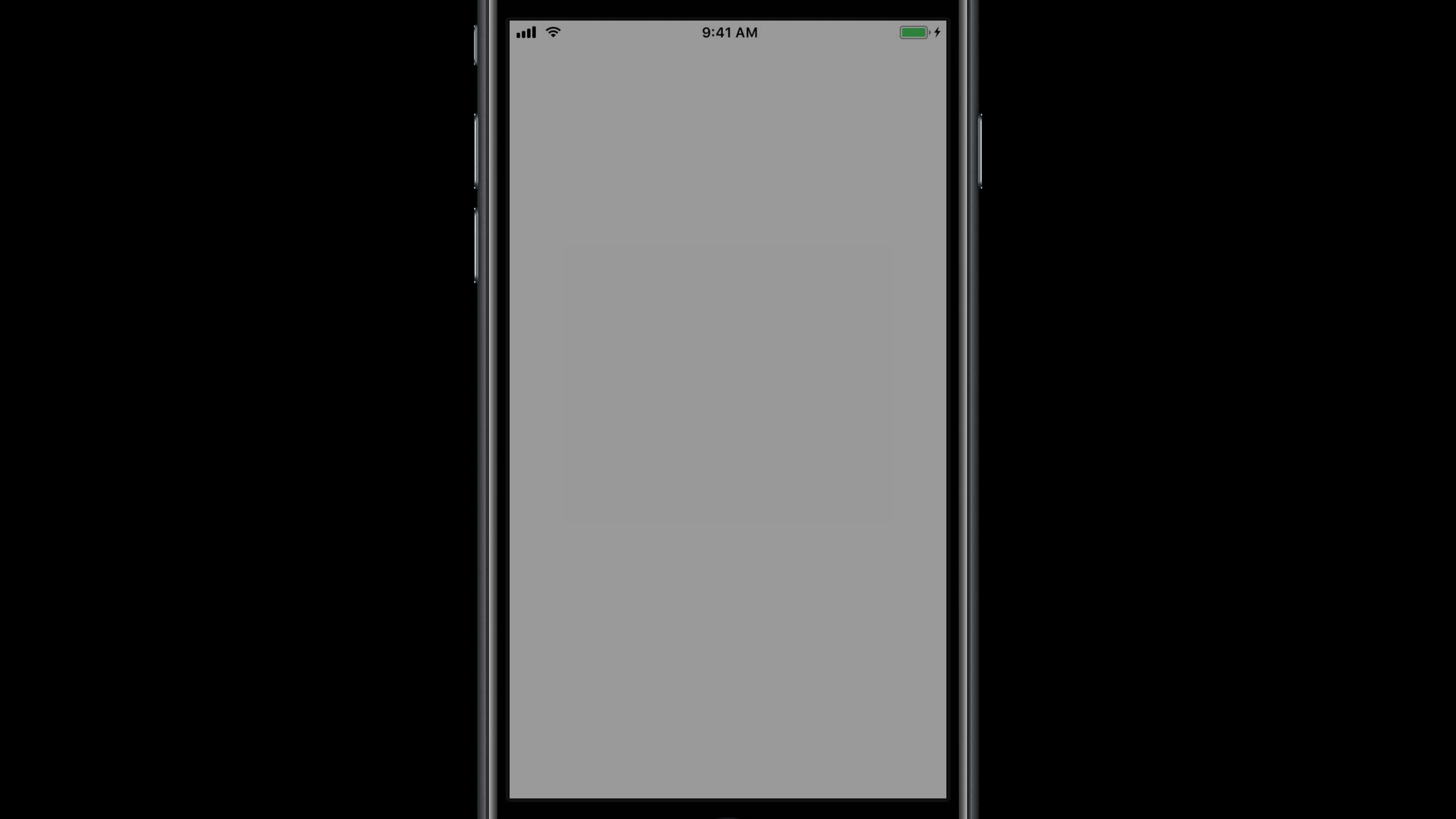
Football
21

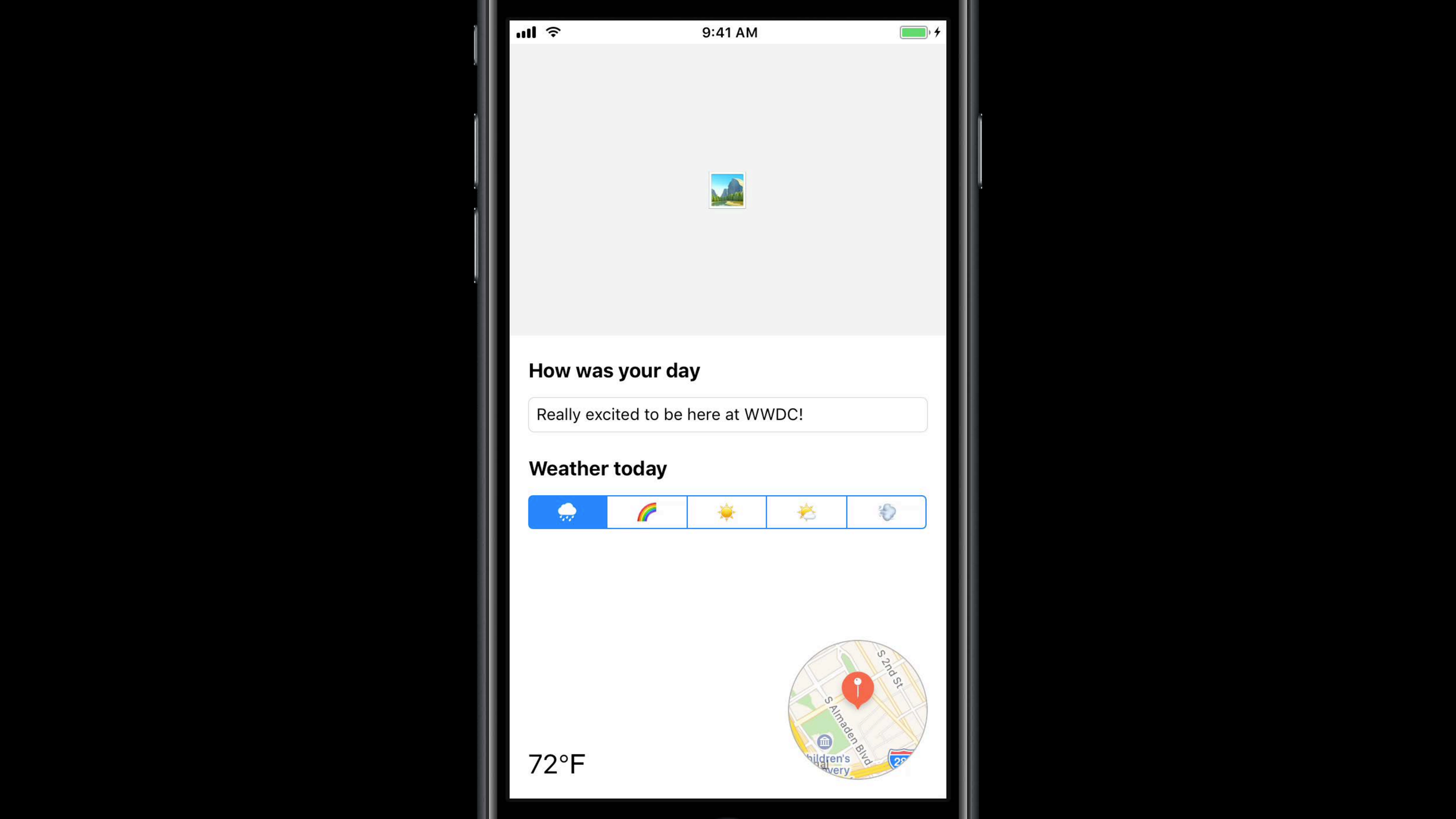
Beach
21

Us
12


9:41 AM


9:41 AM
“Vacation Diary” Would Like to Access Your Photos
This will enable you to choose photos to include in your diary pages.
Don’t Allow
OK


9:41 AM


9:41 AM
How was your day
Really excited to be here at WWDC!
Weather today
S 2nd St
S Almaden Blvd
72°F

Obtaining Metadata

Created Date

Format of the photo

Metadata

```
// protocol implementation
public func imagePickerController(_ picker: UIImagePickerController,
didFinishPickingMediaWithInfo info: [String : Any]) {
   if let imageURL = info[UIImagePickerControllerImageURL] as? URL {
      print(imageURL)
   }
}
```

Obtaining Metadata

Created Date

Format of the photo

Metadata

```
// protocol implementation
public func imagePickerController(_ picker: UIImagePickerController,
didFinishPickingMediaWithInfo info: [String : Any]) {
    if let imageURL = info[UIImagePickerControllerImageURL] as? URL {
        print(imageURL)
    }
}
```

Accessing New Formats

```
var imageExportPreset: UIImagePickerControllerImportExportPreset { get set }
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .compatible
self.present(imagePicker, animated: true, completion: nil)
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .current
self.present(imagePicker, animated: true, completion: nil)
```

Accessing New Formats

```
var imageExportPreset: UIImagePickerControllerImportExportPreset { get set }
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .compatible
self.present(imagePicker, animated: true, completion: nil)
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .current
self.present(imagePicker, animated: true, completion: nil)
```

Accessing New Formats

```
var imageExportPreset: UIImagePickerControllerImportExportPreset { get set }
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .compatible
self.present(imagePicker, animated: true, completion: nil)
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .current
self.present(imagePicker, animated: true, completion: nil)
```

Accessing New Formats

```
var imageExportPreset: UIImagePickerControllerImportExportPreset { get set }
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .compatible
self.present(imagePicker, animated: true, completion: nil)
```

```
let imagePicker = UIImagePickerController()
imagePicker.imageExportPreset = .current
self.present(imagePicker, animated: true, completion: nil)
```

Getting Videos

```
var videoExportPreset: String { get set }
```

```
import AVFoundation

let imagePicker = UIImagePickerController()
imagePicker.videoExportPreset = AVAssetExportPresetHighestQuality
self.present(imagePicker, animated: true, completion: nil)
```

```
// See AVAssetExportSession
```

Getting Videos

```
var videoExportPreset: String { get set }
```

```
import AVFoundation

let imagePicker = UIImagePickerController()
imagePicker.videoExportPreset = AVAssetExportPresetHighestQuality
self.present(imagePicker, animated: true, completion: nil)


// See AVAssetExportSession
```

Getting Videos

```
var videoExportPreset: String { get set }
```

```
import AVFoundation

let imagePicker = UIImagePickerController()
imagePicker.videoExportPreset = AVAssetExportPresetHighestQuality
self.present(imagePicker, animated: true, completion: nil)
```

```
// See AVAssetExportSession
```

Getting Videos

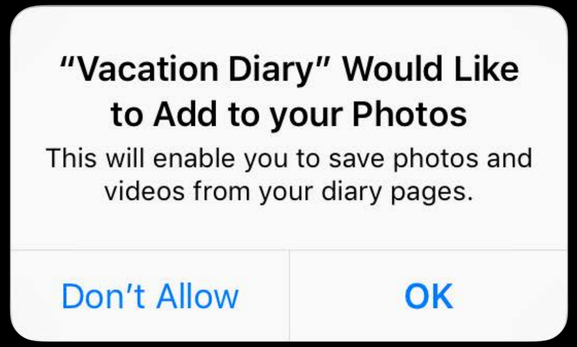
```
var videoExportPreset: String { get set }
```

```
import AVFoundation

let imagePicker = UIImagePickerController()
imagePicker.videoExportPreset = AVAssetExportPresetHighestQuality
self.present(imagePicker, animated: true, completion: nil)
```

```
// See AVAssetExportSession
```

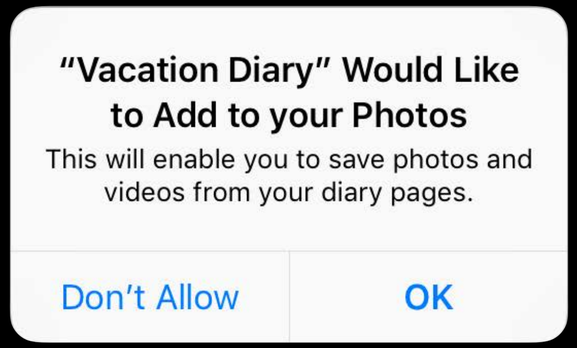
Saving Photos and Videos

```
public func UIImageWriteToSavedPhotosAlbum(_ image: UIImage, _ completionTarget: Any?, _
    completionSelector: Selector?, _ contextInfo: UnsafeMutableRawPointer?)

public func UISaveVideoAtPathToSavedPhotosAlbum(_ videoPath: String, _ completionTarget: Any?,
_ completionSelector: Selector?, _ contextInfo: UnsafeMutableRawPointer?)
```



Saving Photos and Videos

```
public func UIImageWriteToSavedPhotosAlbum(_ image: UIImage, _ completionTarget: Any?, _
    completionSelector: Selector?, _ contextInfo: UnsafeMutableRawPointer?)

public func UISaveVideoAtPathToSavedPhotosAlbum(_ videoPath: String, _ completionTarget: Any?,
_ completionSelector: Selector?, _ contextInfo: UnsafeMutableRawPointer?)
```



Getting a PHAsset

Support for PhotoKit clients

Easy way to get the PHAsset object

```
public func imagePickerController(_ picker: UIImagePickerController,
    didFinishPickingMediaWithInfo info: [String : Any]) {
        if let asset = info[UIImagePickerControllerPHAsset] as? PHAsset {
            print(asset)
        }
    }
}
```

Getting a PHAsset

Support for PhotoKit clients

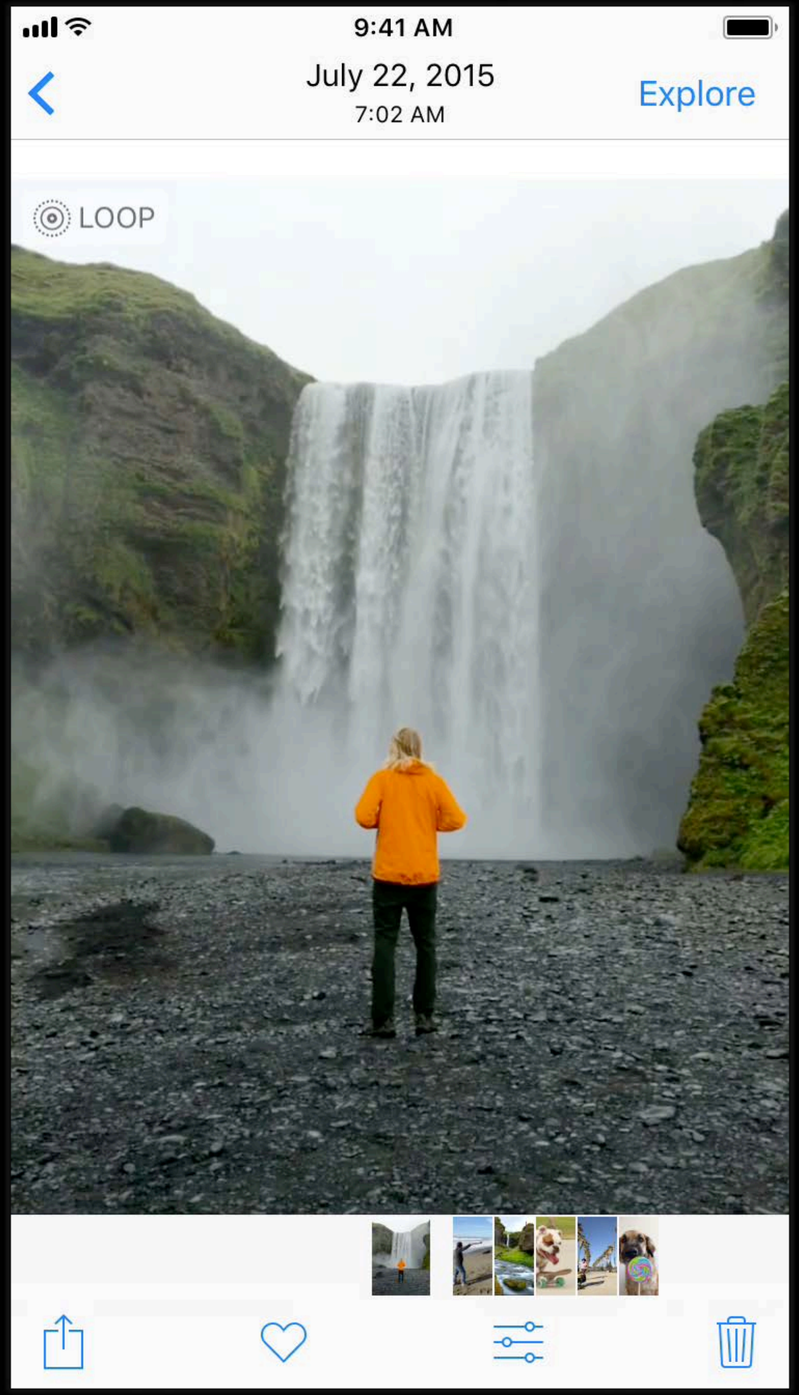
Easy way to get the PHAsset object

```
public func imagePickerController(_ picker: UIImagePickerController,
   didFinishPickingMediaWithInfo info: [String : Any]) {
       if let asset = info[UIImagePickerControllerPHAsset] as? PHAsset {
           print(asset)
       }
    }
}
```

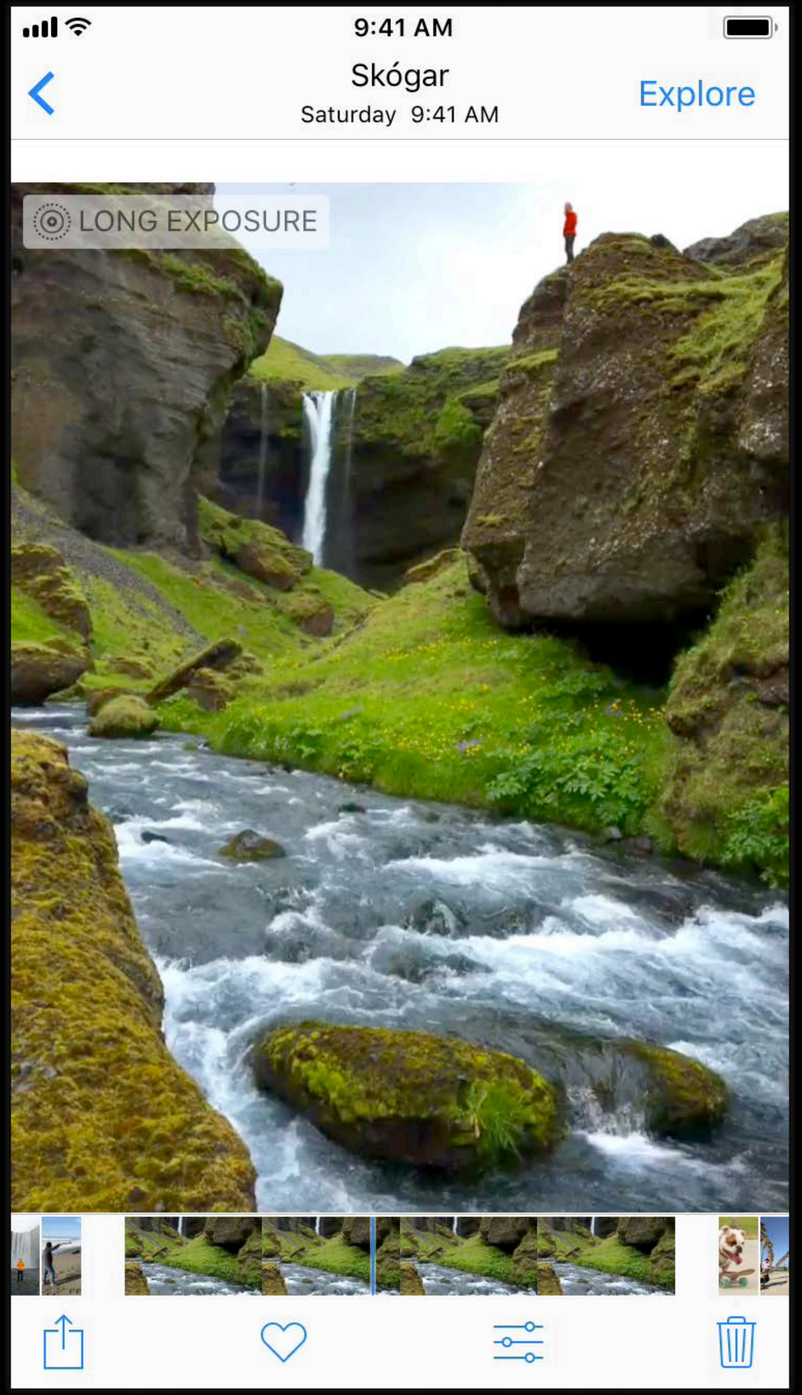
PhotoKit

Additions and best practices

Andreas Karlsson, Photos Engineer


9:41 AM
July 22, 2015
7:02 AM
Explore
LOOP


9:41 AM
May 19
4:10 PM
Explore
BOUNCE


9:41 AM
Skógar
Saturday 9:41 AM
Explore
LONG EXPOSURE

```
enum PHAssetMediaType : Int {
    case unknown
    case image
    case video
    case audio
}

struct PHAssetMediaSubtype : OptionSet {
    static var photoPanorama
    static var photoHDR
    static var photoScreenshot
    static var photoLive
    static var photoDepthEffect
    static var videoStreamed
    static var videoHighFrameRate
    static var videoTimelapse
}
```

```
enum PHAssetMediaType : Int {
    case unknown
    case image
    case video
    case audio
}

struct PHAssetMediaSubtype : OptionSet {
    static var photoPanorama
    static var photoHDR
    static var photoScreenshot
    static var photoLive
    static var photoDepthEffect
    static var videoStreamed
    static var videoHighFrameRate
    static var videoTimelapse
}
```

```
// PHAsset Playback Style

class PHAsset : PHObject {
    var playbackStyle: PHAssetPlaybackStyle { get }
}

enum PHAssetPlaybackStyle : Int {
    case unsupported
    case image
    case imageAnimated
    case livePhoto
    case video
    case videoLooping
}
```

NEW

```
// PHAsset Playback Style

class PHAsset : PHObject {
    var playbackStyle: PHAssetPlaybackStyle { get }
}

enum PHAssetPlaybackStyle : Int {
    case unsupported
    case image
    case imageAnimated
    case livePhoto
    case video
    case videoLooping
}
```

NEW

```
// PHAsset Playback Style

class PHAsset : PHObject {
    var playbackStyle: PHAssetPlaybackStyle { get }
}

enum PHAssetPlaybackStyle : Int {
    case unsupported
    case image
    case imageAnimated
    case livePhoto
    case video
    case videoLooping
}
```

NEW

Animated Image

```
// PlaybackStyle.imageAnimated
imageManager.requestImageData(for: asset,
                              options: options)
{ (data, dataUTI, orientation, info) in
    // Using AnimatedImageView from Sample App
    let animatedImage = AnimatedImage(data: data)
    animatedImageView.animatedImage = animatedImage
}
```

Animated Image

```
// PlaybackStyle.imageAnimated
imageManager.requestImageData(for: asset,
                              options: options)
{ (data, dataUTI, orientation, info) in
    // Using AnimatedImageView from Sample App
    let animatedImage = AnimatedImage(data: data)
    animatedImageView.animatedImage = animatedImage
}
```

Live Photo

```
// PlaybackStyle.livePhoto
imageManager.requestLivePhoto(for: asset,
                        targetSize: pixelSize,
                       contentMode: .aspectFill,
                           options: options)
{ (livePhoto, info) in
    // Using PHLivePhotoView
    livePhotoView.livePhoto = livePhoto
}
```

Live Photo

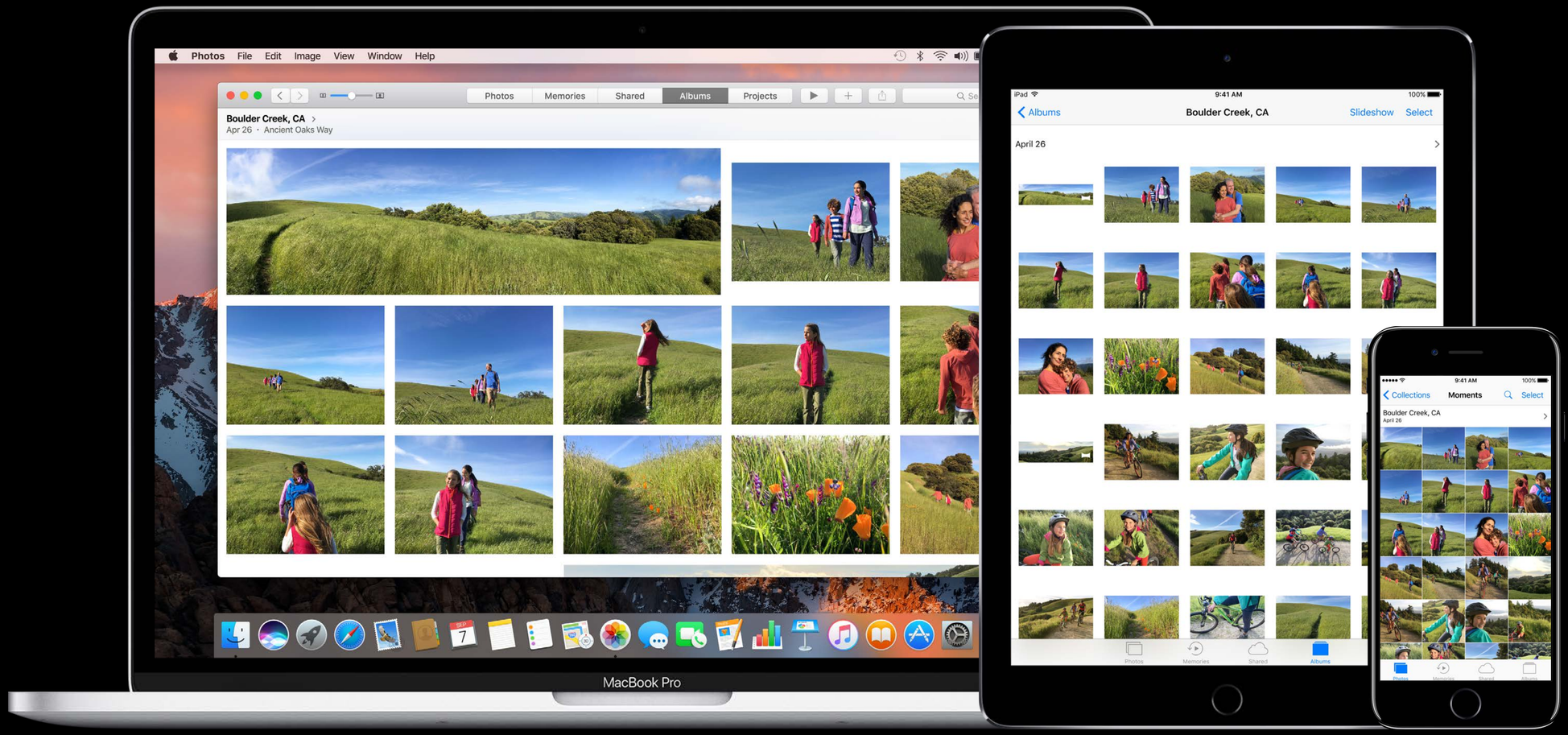
```
// PlaybackStyle.livePhoto
imageManager.requestLivePhoto(for: asset,
                        targetSize: pixelSize,
                       contentMode: .aspectFill,
                           options: options)
{ (livePhoto, info) in
    // Using PHLivePhotoView
    livePhotoView.livePhoto = livePhoto
}
```

Looping Video

```
// PlaybackStyle.videoLooping
imageManager.requestPlayerItem(forVideo: asset,
                               options: options)
{ playerItem, info in
    DispatchQueue.main.async {
        let player = AVQueuePlayer()
        playerLooper = AVPlayerLooper(player: player,
                                      templateItem: playerItem)
        playerLayer.player = player
        player.play()
    }
}
```

Looping Video

```
// PlaybackStyle.videoLooping
imageManager.requestPlayerItem(forVideo: asset,
                               options: options)
{ playerItem, info in
    DispatchQueue.main.async {
        let player = AVQueuePlayer()
        playerLooper = AVPlayerLooper(player: player,
                                      templateItem: playerItem)
        playerLayer.player = player
        player.play()
    }
}
```


Photos File Edit Image View Window Help
Photos Memories Shared Albums Projects
Boulder Creek, CA
Apr 26 · Ancient Oaks Way
MacBook Pro
iPad
9:41 AM
100%
Albums
Boulder Creek, CA
Slideshow Select
April 26
Photos Memories Shared Albums
Collections Moments Select
Boulder Creek, CA
April 26

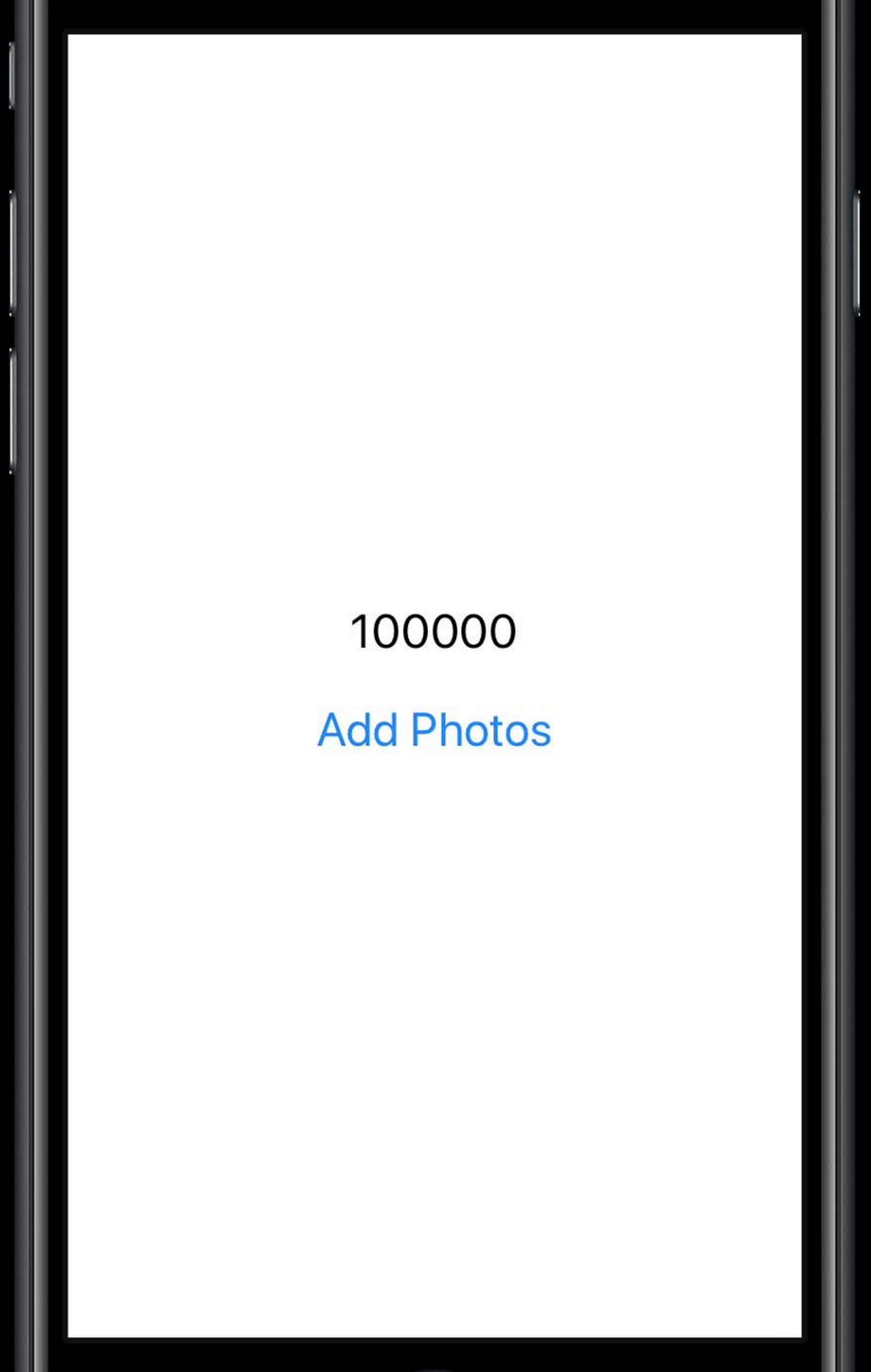
Test your app with a large photo library

Large Photo Libraries for Testing

New Sample App—Photo Library Filler

Download and run on oldest device

Generate 100,000 photos



Fetching Assets

```
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

```
let smartAlbums = PHAssetCollection.fetchAssetCollections(with: .smartAlbum,
                                                          subtype: .smartAlbumFavorites,
                                                          options: nil)
let assets = PHAsset.fetchAssets(in: smartAlbums.firstObject, options: nil)
```

Fetching Assets

```
let options = PHFetchOptions()
options.predicate = NSPredicate(format: "isFavorite = %d", true)
options.sortDescriptors = [NSSortDescriptor(key: "creationDate", ascending: true)]
let assets = PHAsset.fetchAssets(with: options)
```

```
let smartAlbums = PHAssetCollection.fetchAssetCollections(with: .smartAlbum,
                                                          subtype: .smartAlbumFavorites,
                                                          options: nil)
let assets = PHAsset.fetchAssets(in: smartAlbums.firstObject, options: nil)
```

PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
```

PHFetchResult

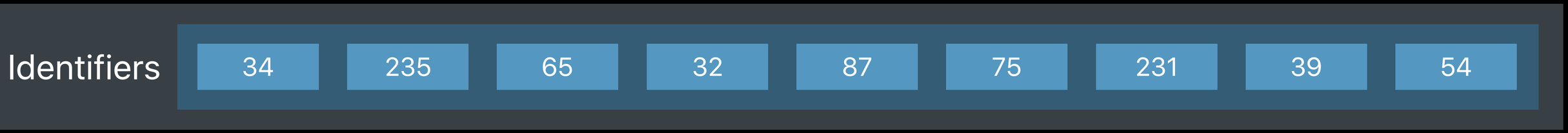
```
let assets = PHAsset.fetchAssets(with: options)
```

PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
```

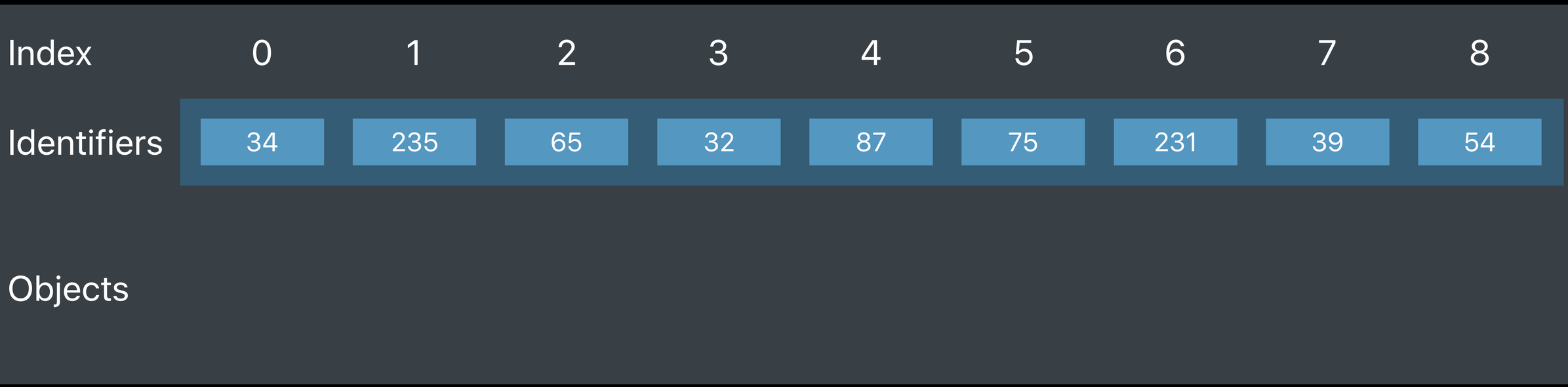
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



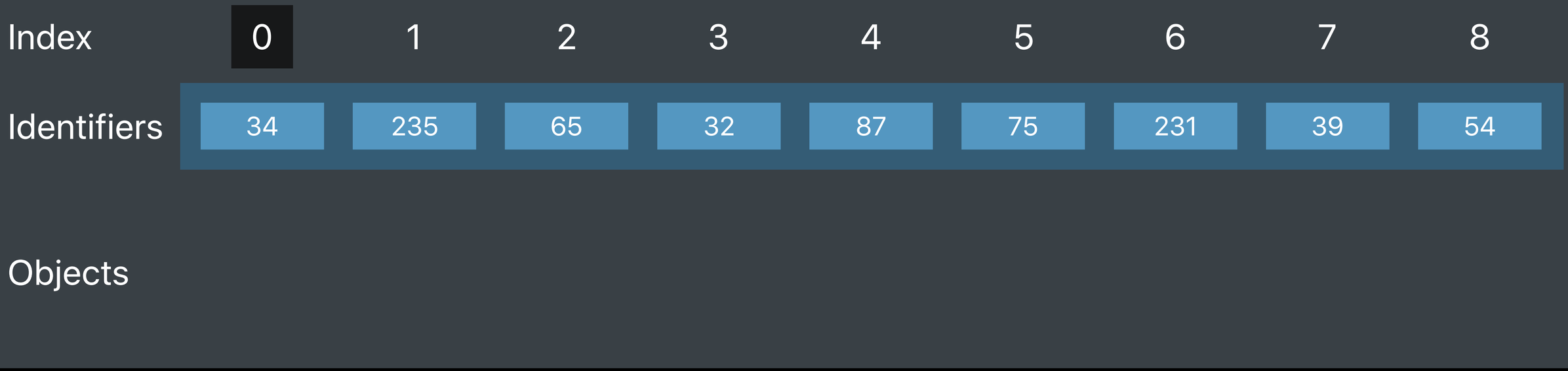
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



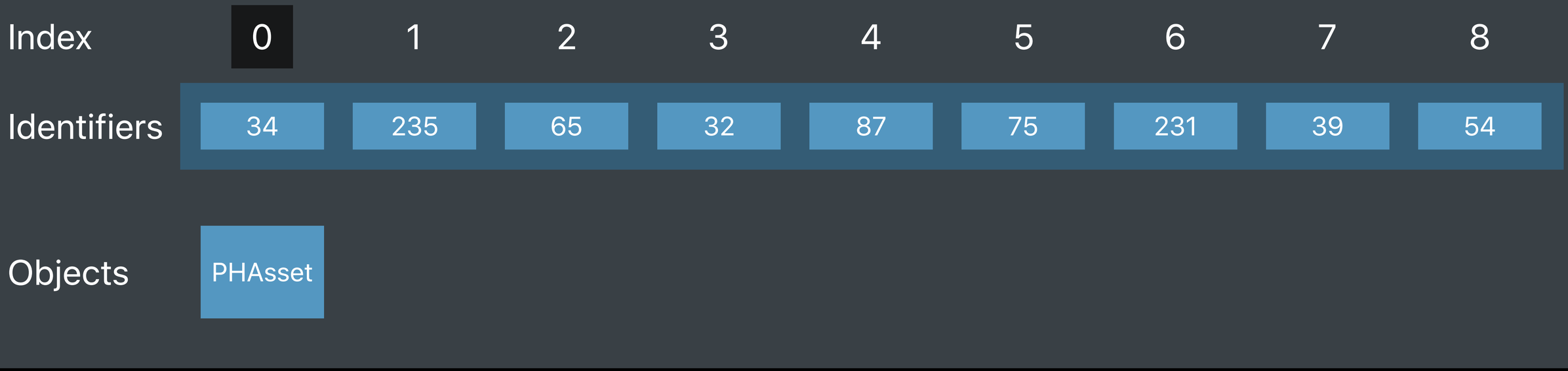
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



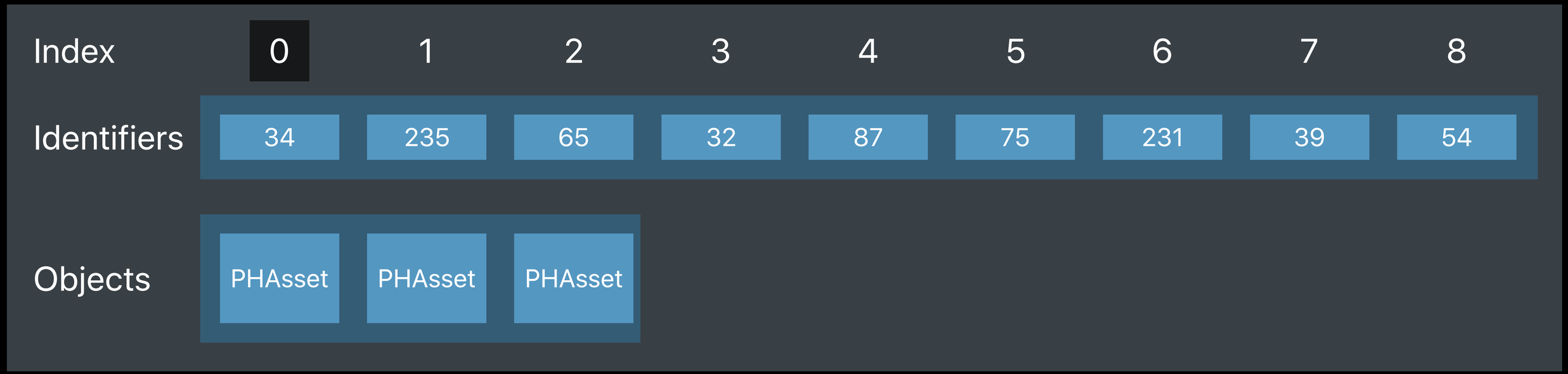
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



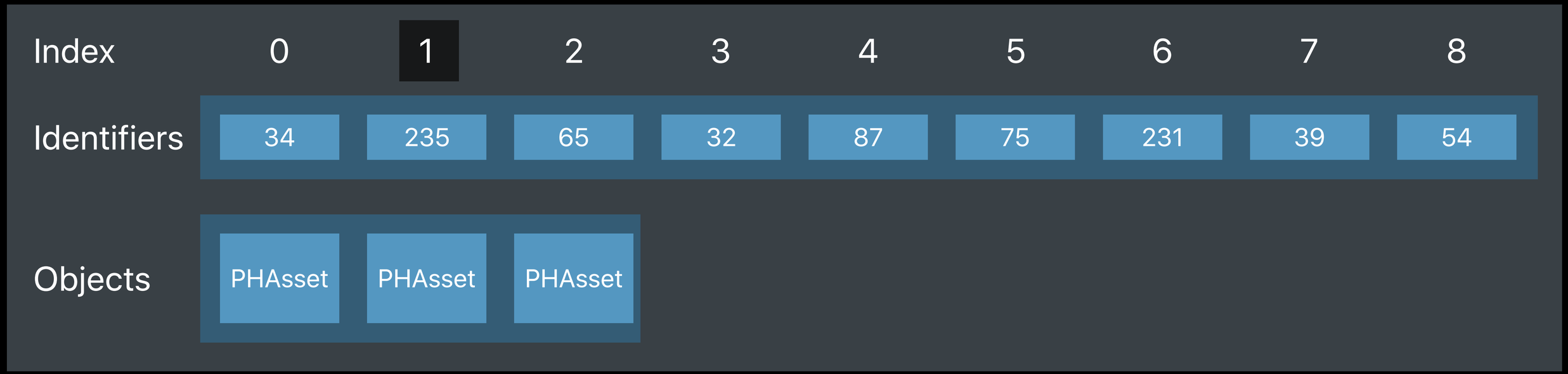
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



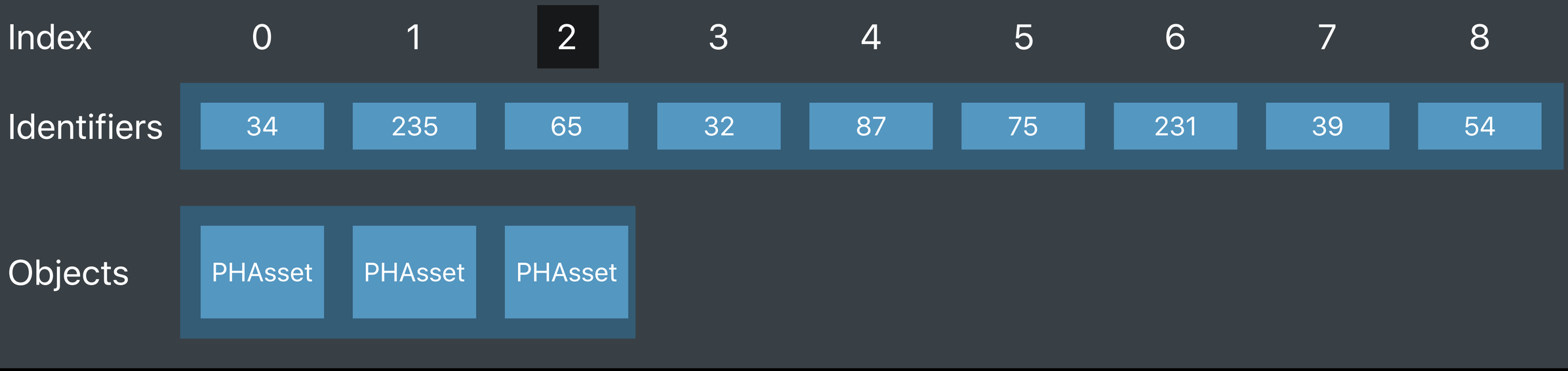
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



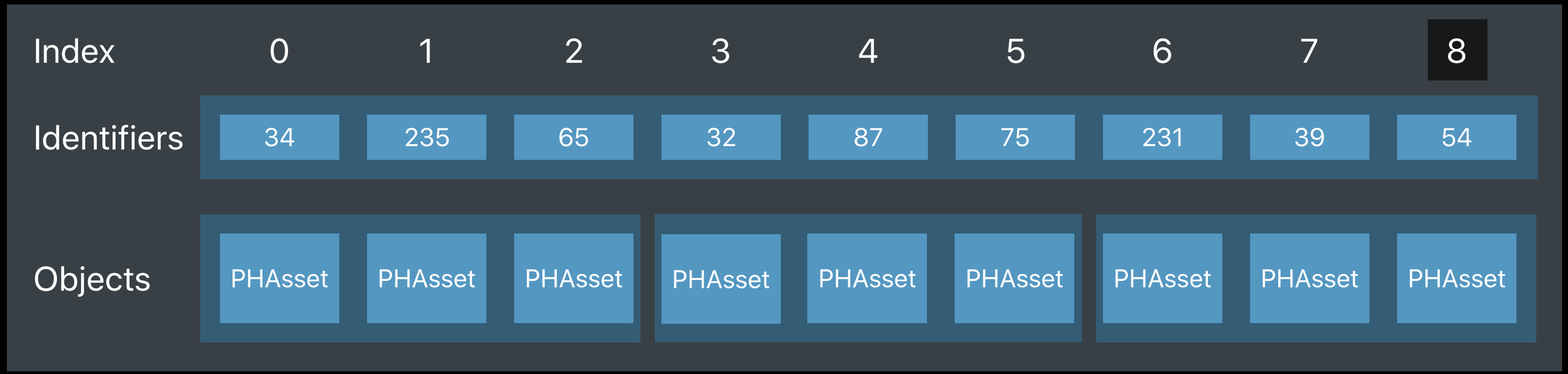
PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```



PHFetchResult

```
let assets = PHAsset.fetchAssets(with: options)
assets.enumerateObjects { (asset, index, stop) in
    // do something with the asset
}
```

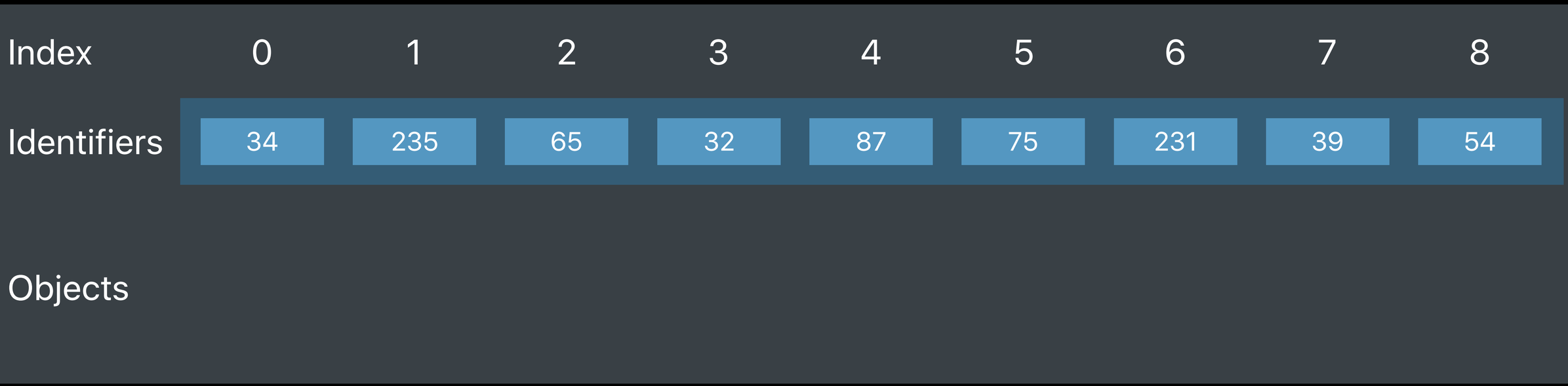


10s

to enumerate a large fetch result

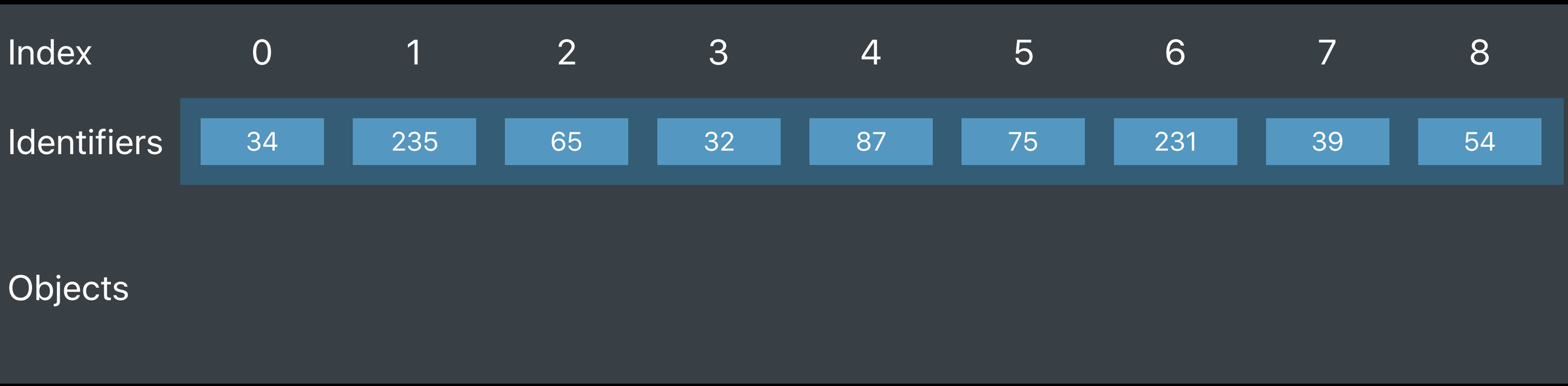
Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



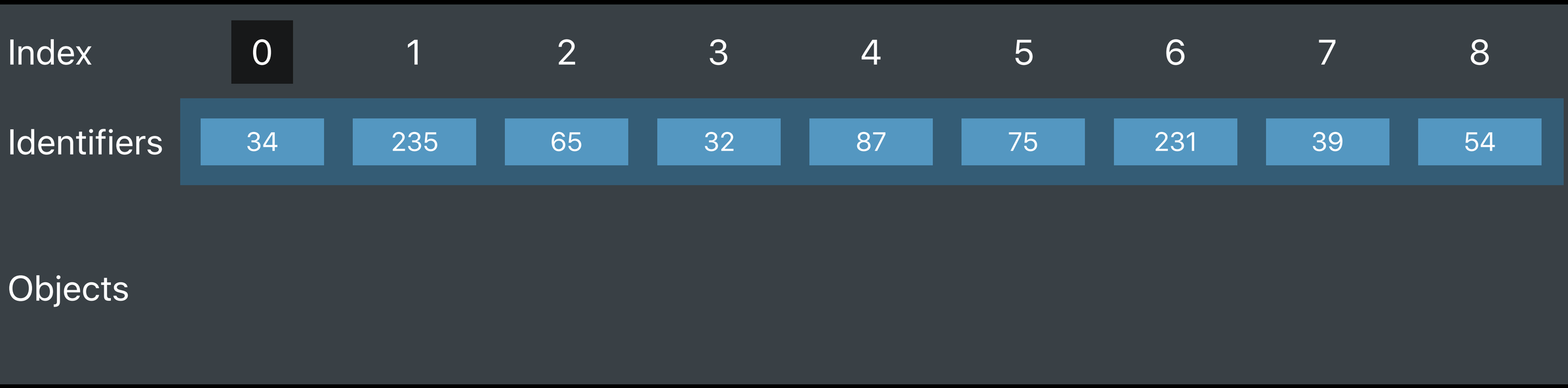
Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



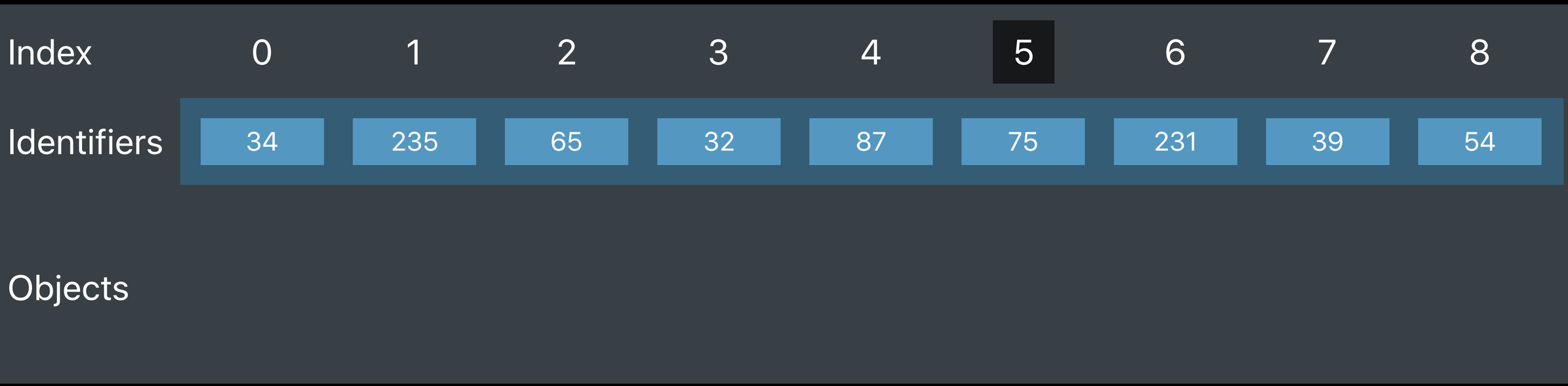
Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



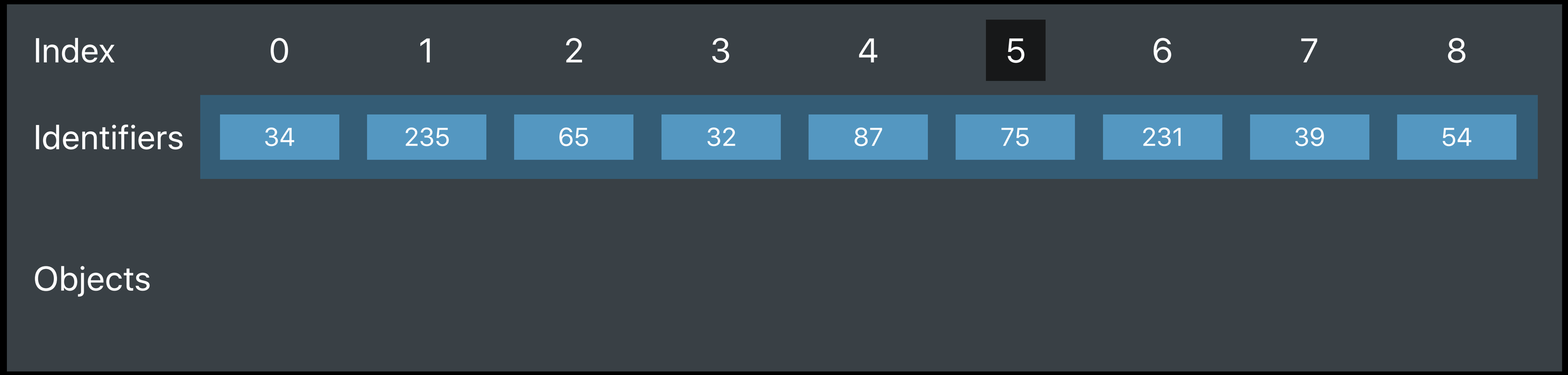
Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



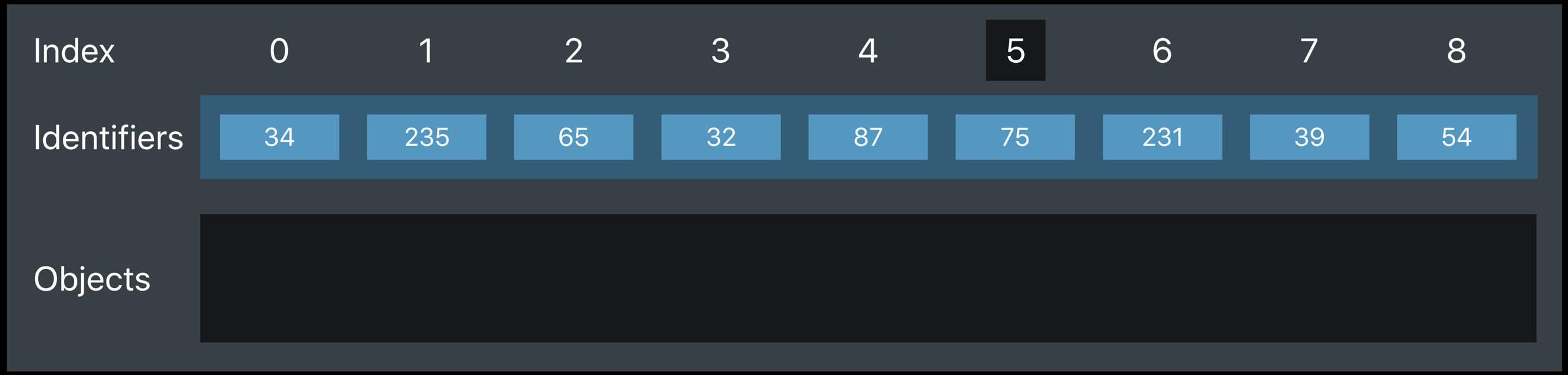
Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



Finding Assets in a PHFetchResult

```
let someAsset = // Asset 75
let assets = PHAsset.fetchAssets(with: options)
let index = assets.indexOfObject(someAsset)
let contains = assets.containsObject(someAsset)
```



Test your app with a large photo library

NEW

Photos Project Extensions

Bring your app to macOS Photos

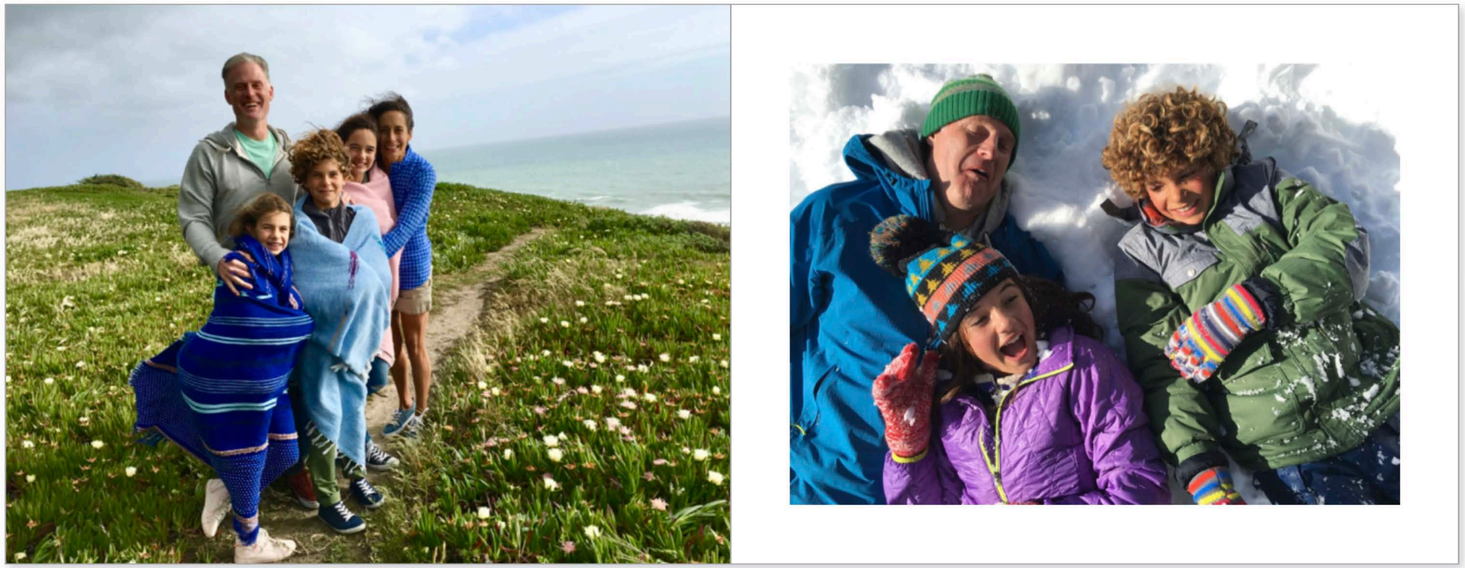
Eric Hanson, Photos Platform Evangelist


Buy Book
Library
Photos
Memories
Favorites
People
Places
Imports
Recently Deleted
Albums
Projects
My Projects
Santa Cruz Beach Boa...
The Holden Family 2017
I Love You
On the Road
Grandma's birthday ca...
Utah Trip
Family Ski Trip
Our New Puppy
Journey to Iceland
Holiday Card
Summer in Tahoe
Happy Anniversary
A Year in the Garden
On the Road
Santa Cruz
Santa Cruz Beach Boardwalk
Cover & Flap
30 Pages – $64.89
SANTA CRUZ VACATION
BOARDWALK
Options
Options
Clear Placed Photos
Auto-Fill
Photos
Unused Photos
Add Photos
MacBook Pro

Photos - Create with Shutterfly

Done

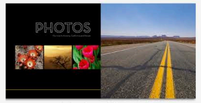
Shutterfly | 8 x 11 Hardcover Photo Book | Change | From $39.99 | Personalize | Order



6 7

Styles | Density

Modern White



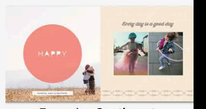
Modern Black

Everyday Happiness

Hello Spring

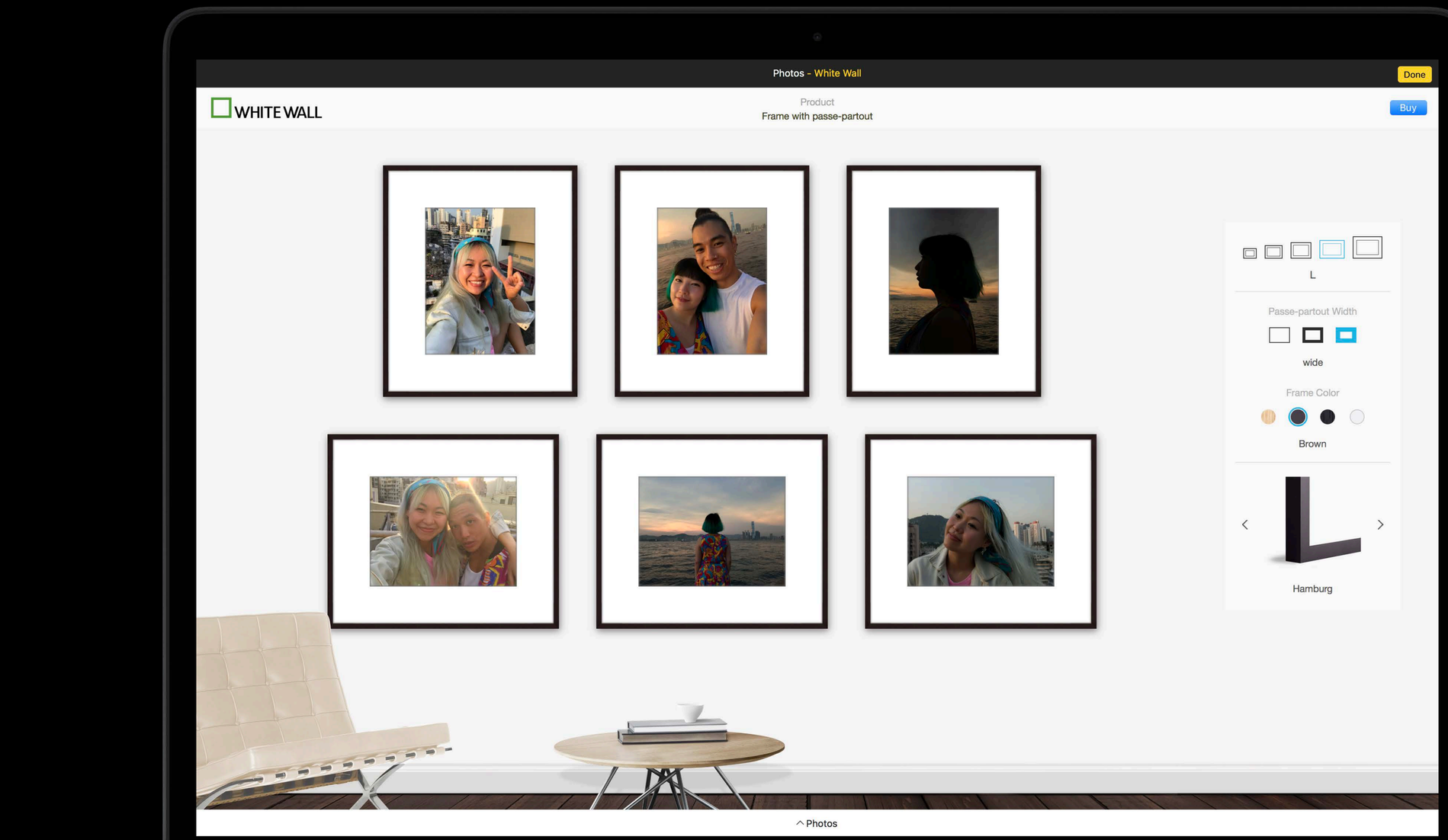
Disney Adventures

Kraft Pop



Everyday Sentiments

MacBook Pro


Photos - White Wall
Done
WHITE WALL
Product
Frame with passe-partout
Buy
L
Passe-partout Width
wide
Frame Color
Brown
Hamburg
Photos
MacBook Pro

Photos - Wix Web Albums

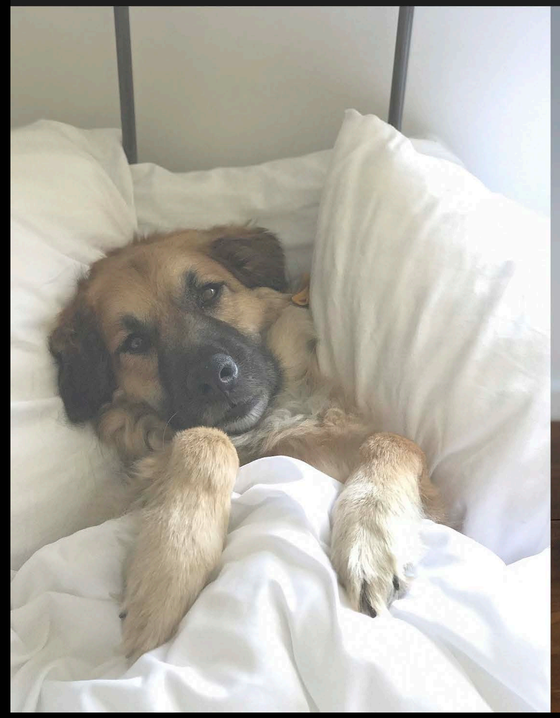
Done

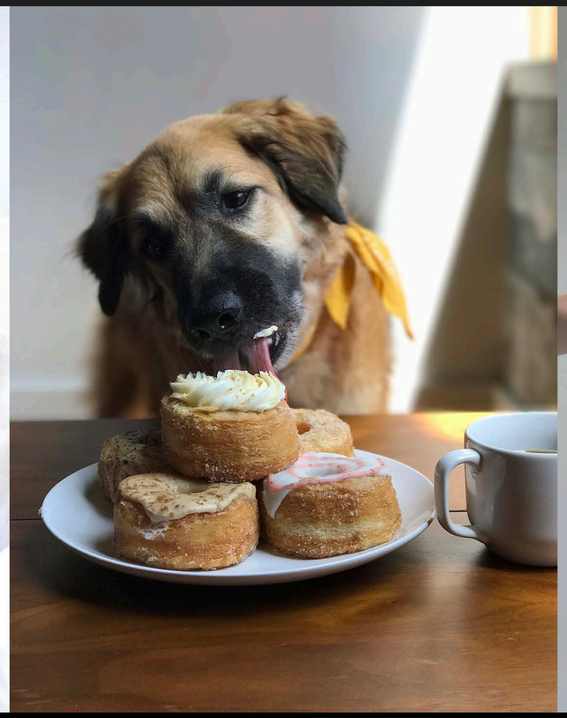
WiX | Photos | Layouts | Info | All changes saved | Settings | Help | Publish Now

Dog Show

May 26, 2017

View Photos





MacBook Pro

Photos - Animoto

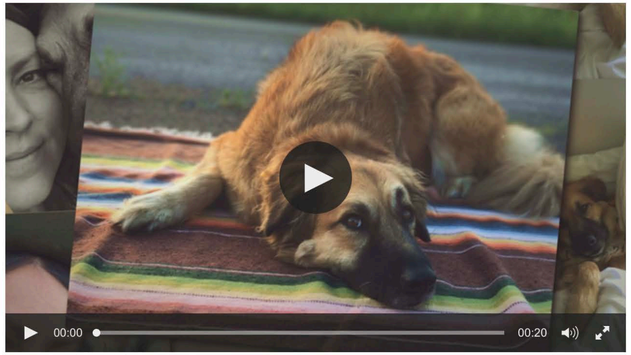
Done

Our Buddy

ANIMOTO

Our Buddy

June 05, 2017



STATS COMMENTS

Like 0

0 Comments

Sort by Oldest

Add a comment...

Facebook Comments Plugin

BY

Johnny Appleseed

ABOUT

Add a description in your video settings.

SHARE

Email Share Tweet Link

YouTube Embed More

OPTIONS

Edit Video

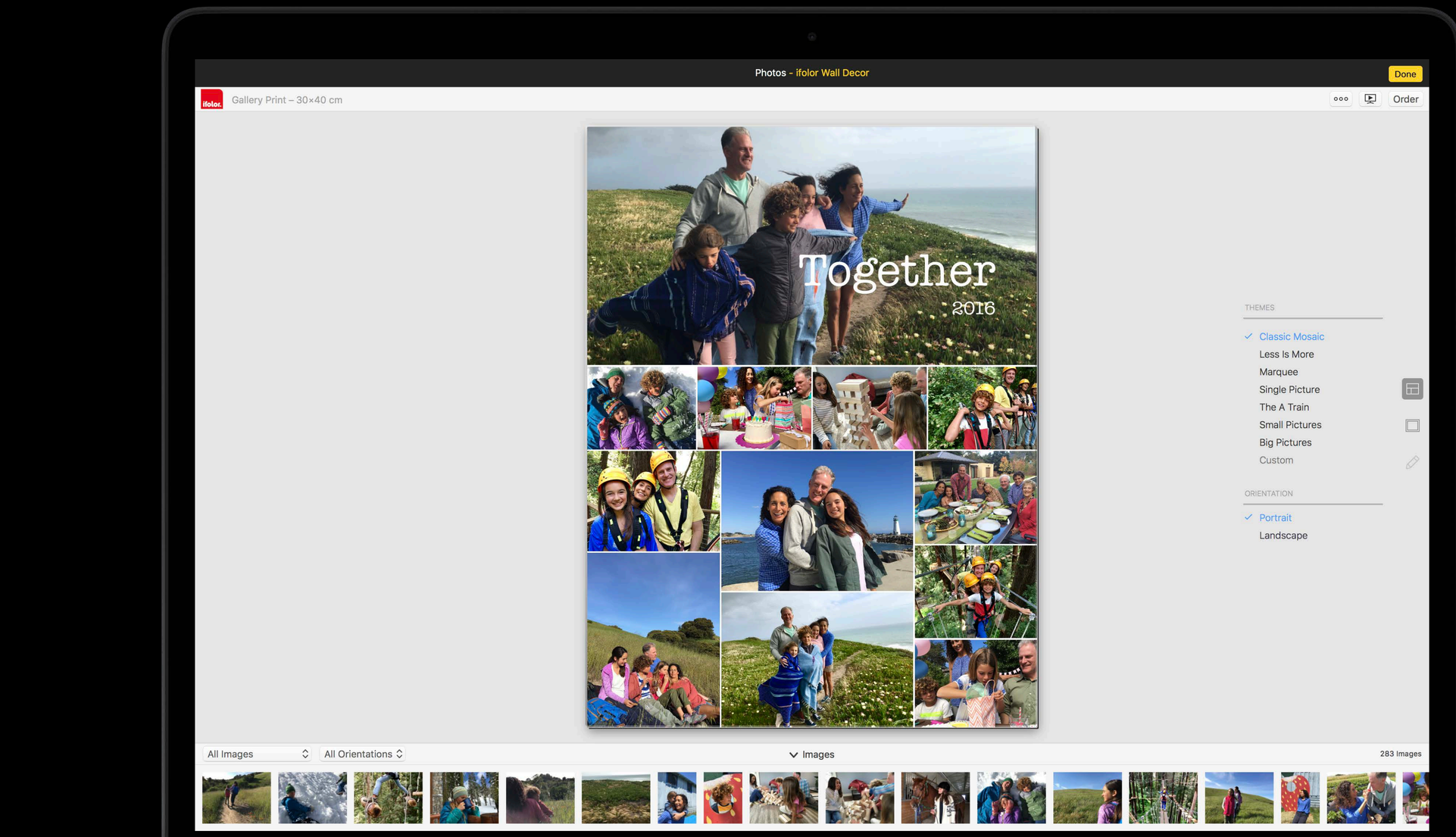
Upgrade to HD

Download

Settings

1 Page View

MacBook Pro


Photos - ifolor Wall Decor
Done
Gallery Print – 30×40 cm
Order
Together
2016
THEMES
Classic Mosaic
Less Is More
Marquee
Single Picture
The A Train
Small Pictures
Big Pictures
Custom
ORIENTATION
Portrait
Landscape
All Images
All Orientations
Images
283 Images
MacBook Pro

Demo

One more. In action.

How it all works

NSExtension Point

Similar to Photos editing extensions

NSExtension Point

Similar to Photos editing extensions

New extension point: `com.apple.photo-project`

NSExtension Point

Similar to Photos editing extensions

New extension point: `com.apple.photo-project`

New Xcode template

NSExtension Point

Similar to Photos editing extensions

New extension point: `com.apple.photo-project`

New Xcode template

Automatic discovery of your app inside Photos

NSExtension Point

Similar to Photos editing extensions

New extension point: `com.apple.photo-project`

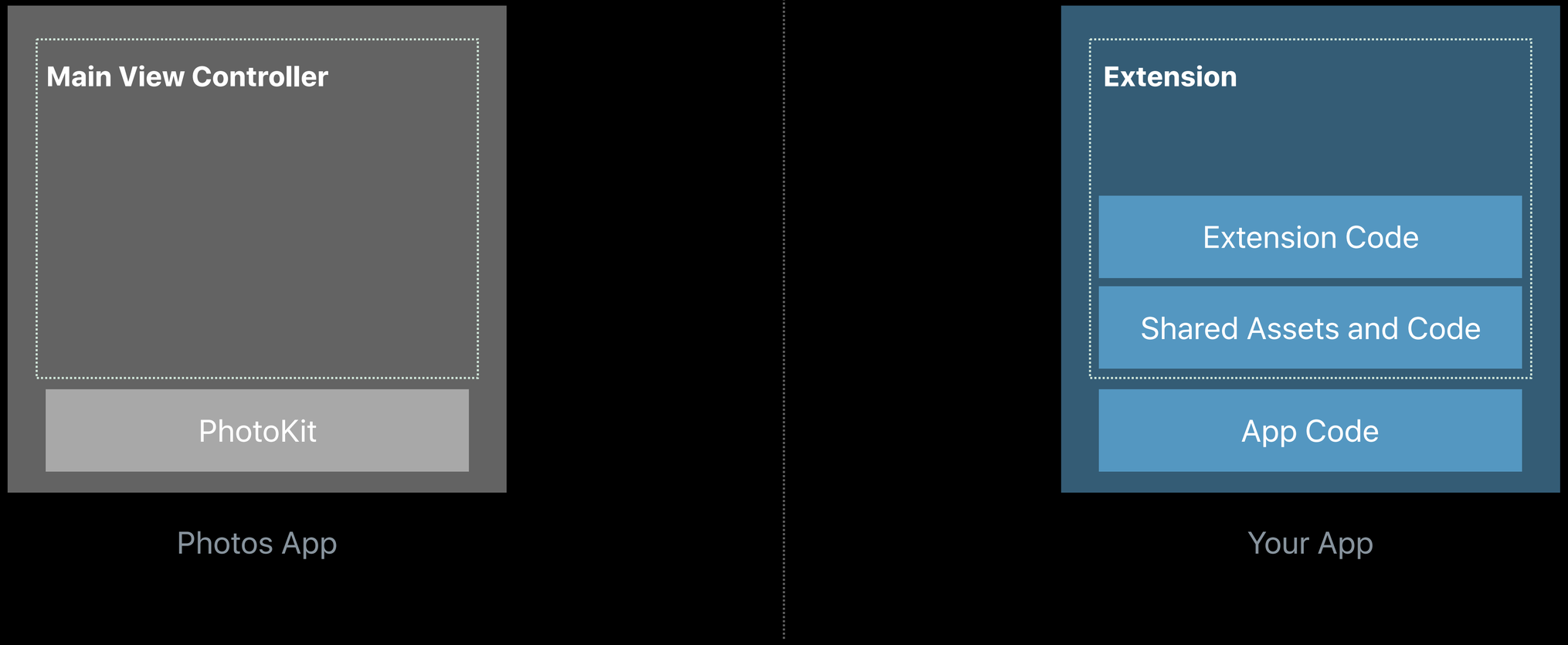
New Xcode template

Automatic discovery of your app inside Photos

Mac App Store link to find extension apps

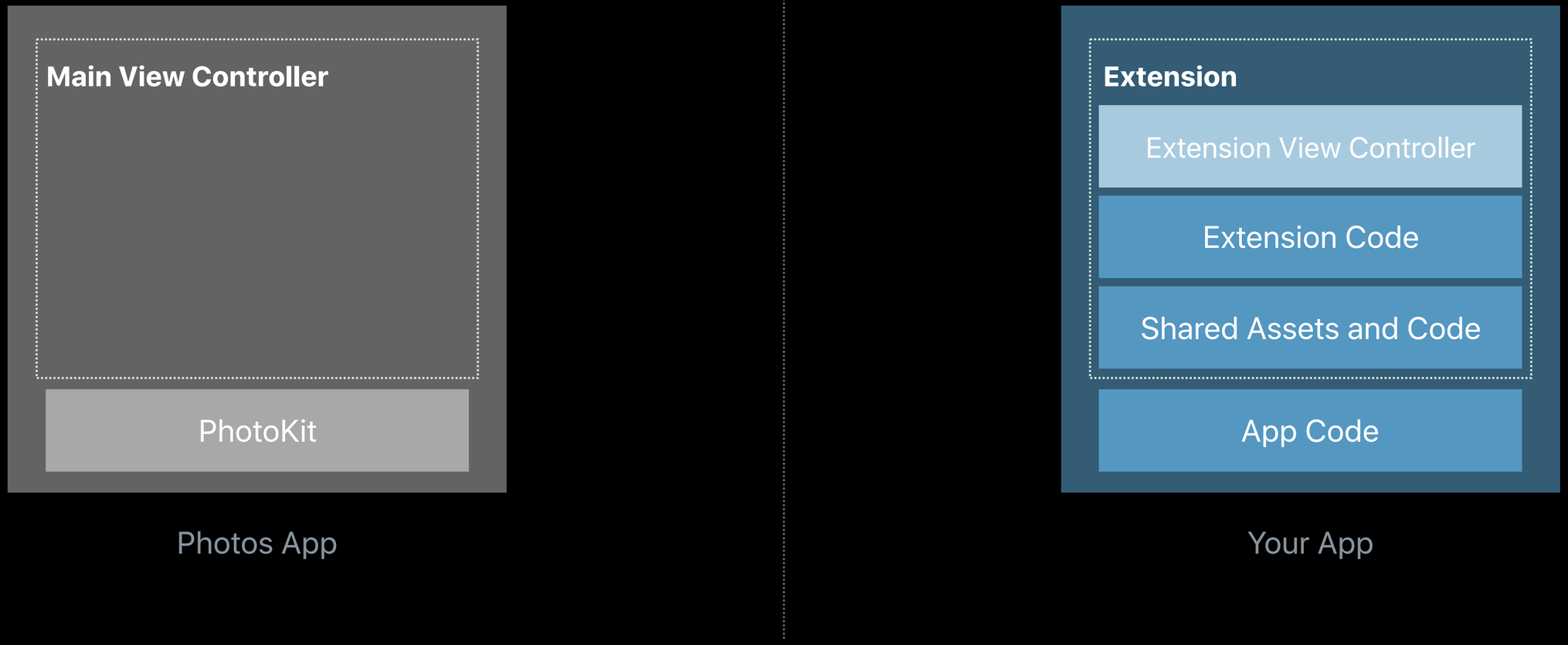
PHProjectExtensionController

Principal controller in your extension



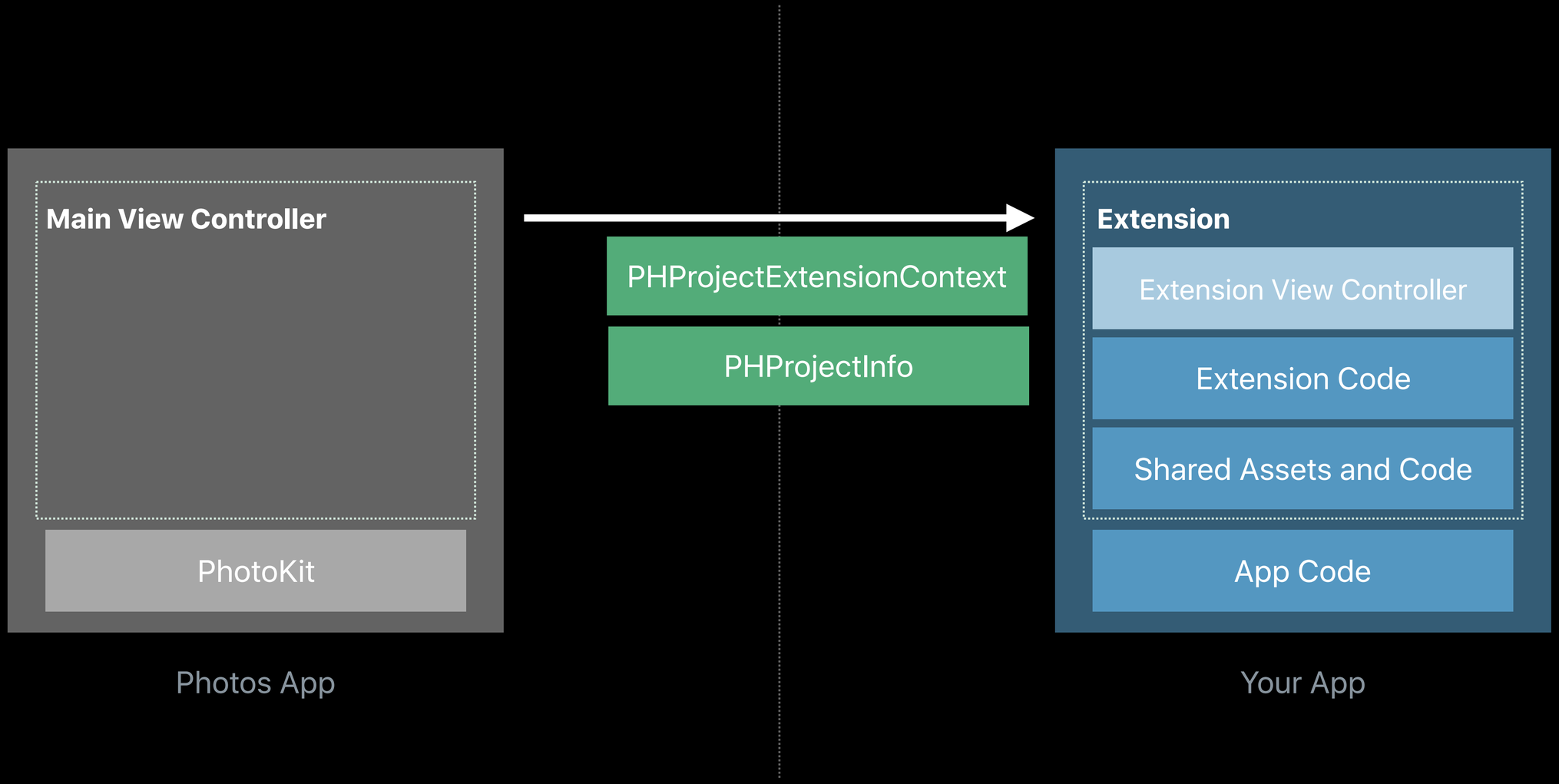
PHProjectExtensionController

Principal controller in your extension



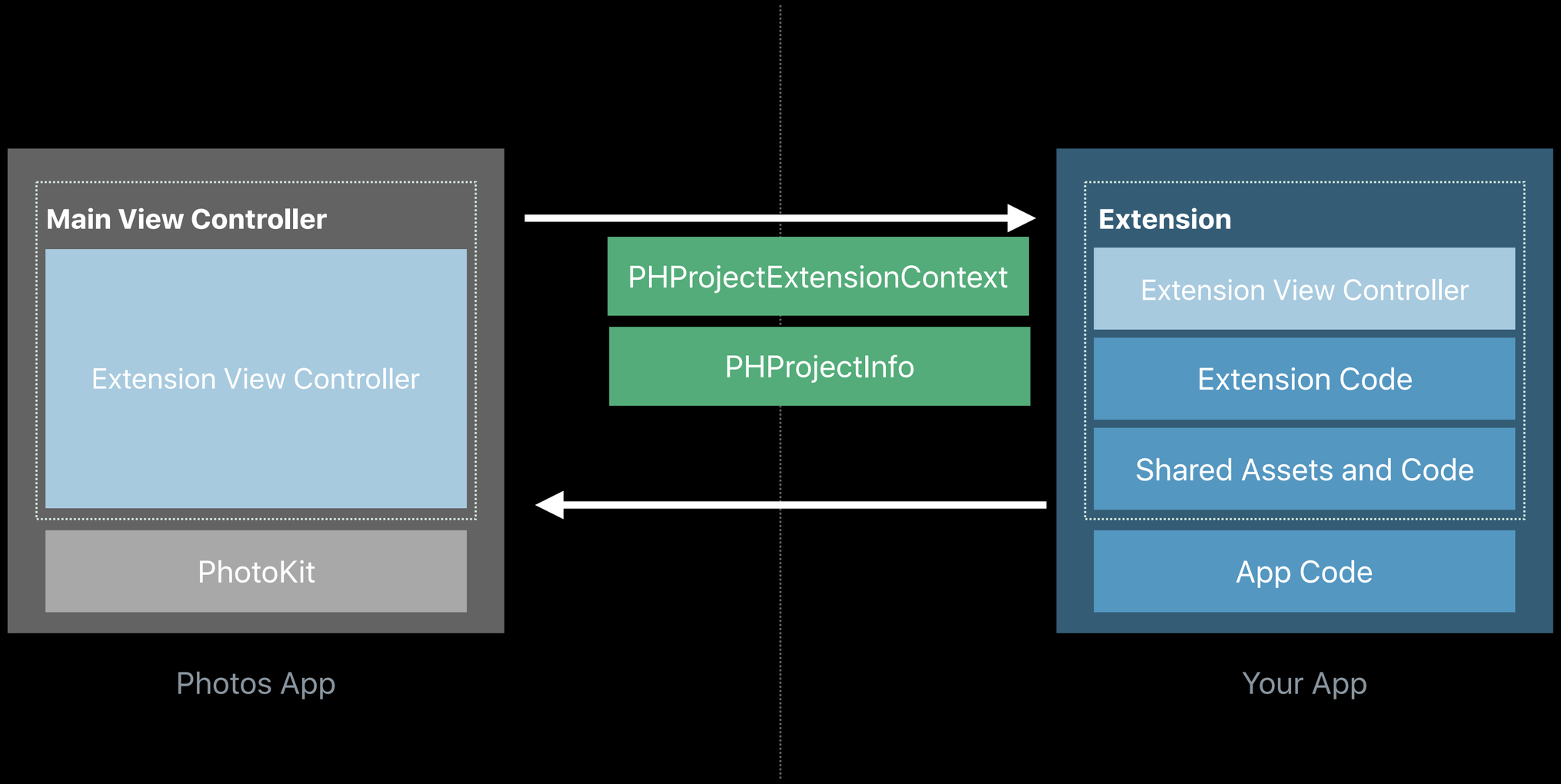
PHProjectExtensionController

Principal controller in your extension



PHProjectExtensionController

Principal controller in your extension



```
//  PHProjectExtensionController.h
//  PhotosUI

public protocol PHProjectExtensionController : NSObjectProtocol {

    optional public var supportedProjectTypes: [PHProjectTypeDescription] { get }

    // Called the first time a project is created
    public func beginProject(with extensionContext: PHProjectExtensionContext,
        projectInfo: PHProjectInfo, completion: @escaping (Error?) -> Void)

       // Called anytime the user returns to a project that was previously created
    public func resumeProject(with extensionContext: PHProjectExtensionContext,
        completion: @escaping (Error?) -> Void)

       // Called when a user is switching away from the project
    public func finishProject(completionHandler completion: @escaping () -> Void)

}
```

```
//  PHProjectExtensionController.h
//  PhotosUI

public protocol PHProjectExtensionController : NSObjectProtocol {

    optional public var supportedProjectTypes: [PHProjectTypeDescription] { get }

    // Called the first time a project is created
    public func beginProject(with extensionContext: PHProjectExtensionContext,
        projectInfo: PHProjectInfo, completion: @escaping (Error?) -> Void)

    // Called anytime the user returns to a project that was previously created
    public func resumeProject(with extensionContext: PHProjectExtensionContext,
        completion: @escaping (Error?) -> Void)

    // Called when a user is switching away from the project
    public func finishProject(completionHandler completion: @escaping () -> Void)

}
```

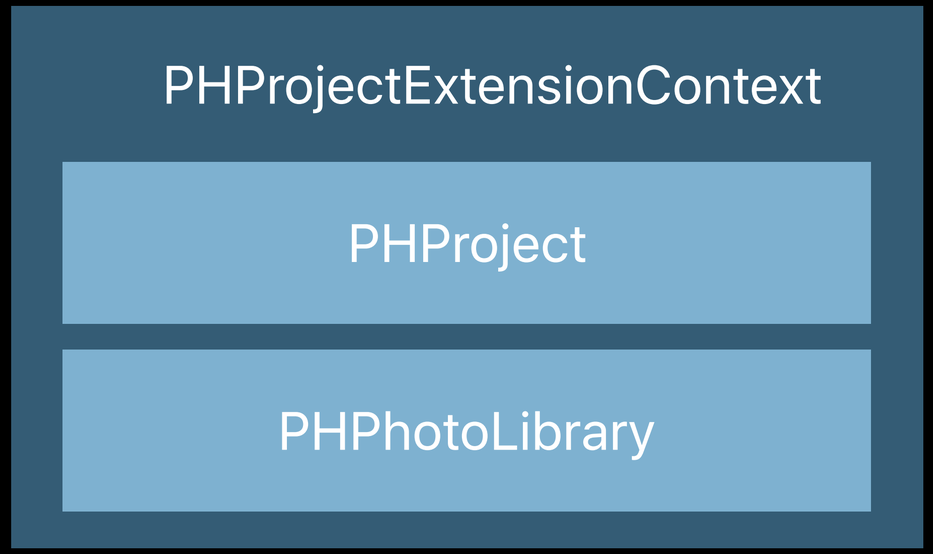
```
//  PHProjectExtensionController.h
//  PhotosUI

public protocol PHProjectExtensionController : NSObjectProtocol {

    optional public var supportedProjectTypes: [PHProjectTypeDescription] { get }

    // Called the first time a project is created
    public func beginProject(with extensionContext: PHProjectExtensionContext,
        projectInfo: PHProjectInfo, completion: @escaping (Error?) -> Void)

    // Called anytime the user returns to a project that was previously created
    public func resumeProject(with extensionContext: PHProjectExtensionContext,
        completion: @escaping (Error?) -> Void)

    // Called when a user is switching away from the project
    public func finishProject(completionHandler completion: @escaping () -> Void)

}
```

```
//  PHProjectExtensionController.h
//  PhotosUI

public protocol PHProjectExtensionController : NSObjectProtocol {

    optional public var supportedProjectTypes: [PHProjectTypeDescription] { get }

    // Called the first time a project is created
    public func beginProject(with extensionContext: PHProjectExtensionContext,
        projectInfo: PHProjectInfo, completion: @escaping (Error?) -> Void)

    // Called anytime the user returns to a project that was previously created
    public func resumeProject(with extensionContext: PHProjectExtensionContext,
        completion: @escaping (Error?) -> Void)

    // Called when a user is switching away from the project
    public func finishProject(completionHandler completion: @escaping () -> Void)

}
```

```
//  PHProjectExtensionController.h
//  PhotosUI

public protocol PHProjectExtensionController : NSObjectProtocol {

    optional public var supportedProjectTypes: [PHProjectTypeDescription] { get }

    // Called the first time a project is created
    public func beginProject(with extensionContext: PHProjectExtensionContext,
        projectInfo: PHProjectInfo, completion: @escaping (Error?) -> Void)

       // Called anytime the user returns to a project that was previously created
    public func resumeProject(with extensionContext: PHProjectExtensionContext,
        completion: @escaping (Error?) -> Void)

       // Called when a user is switching away from the project
    public func finishProject(completionHandler completion: @escaping () -> Void)

}
```

PHProjectExtensionContext

Your access to the project and photo library



PHProject

Intended for project state

Layout and asset references, not image data

Limited to 1 MB

```
//  PHProject.h
//  Photos

class PHProject : PHAssetCollection {
    var projectExtensionData: Data { get }
}
```

PHProject

Intended for project state

Layout and asset references, not image data

Limited to 1 MB

```
//  PHProject.h
//  Photos

class PHProject : PHAssetCollection {
    var projectExtensionData: Data { get }
}
```

PHProjectChangeRequest

Title and key asset, and data

Extension’s project data

```
do {
    let changeRequest = PHProjectChangeRequest(project: self.project)

   try self.library.performChangesAndWait {
        changeRequest.projectExtensionData =
            NSKeyedArchiver.archivedData(withRootObject: cloudIdentifiers)
    }

} catch {
    print("Failed to save project data: \(error.localizedDescription)")
}
```

PHProjectChangeRequest

Title and key asset, and data

Extension’s project data

```
do {
    let changeRequest = PHProjectChangeRequest(project: self.project)

   try self.library.performChangesAndWait {
        changeRequest.projectExtensionData =
            NSKeyedArchiver.archivedData(withRootObject: cloudIdentifiers)
    }

} catch {
    print("Failed to save project data: \(error.localizedDescription)")
}
```

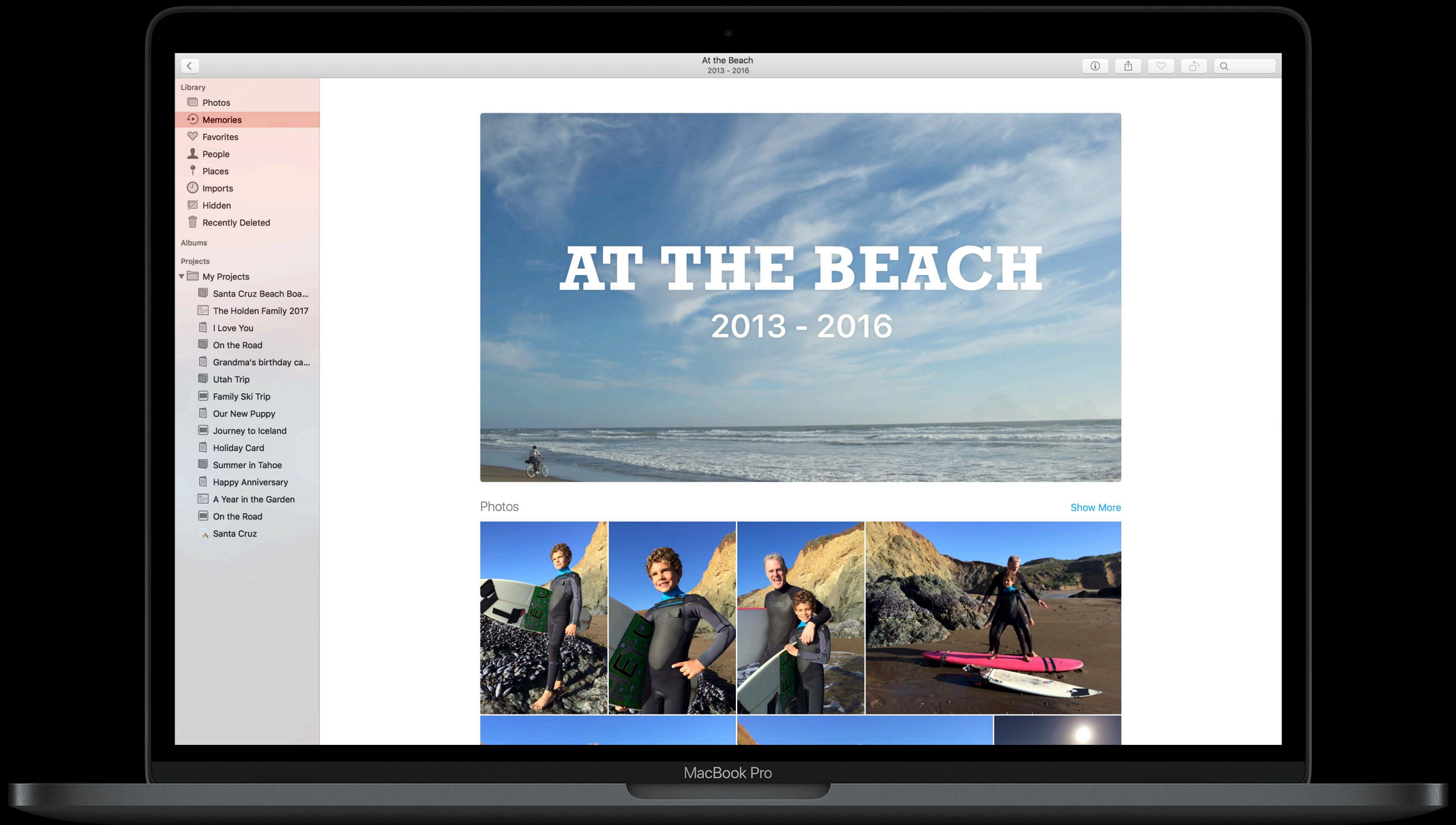
PHProjectChangeRequest

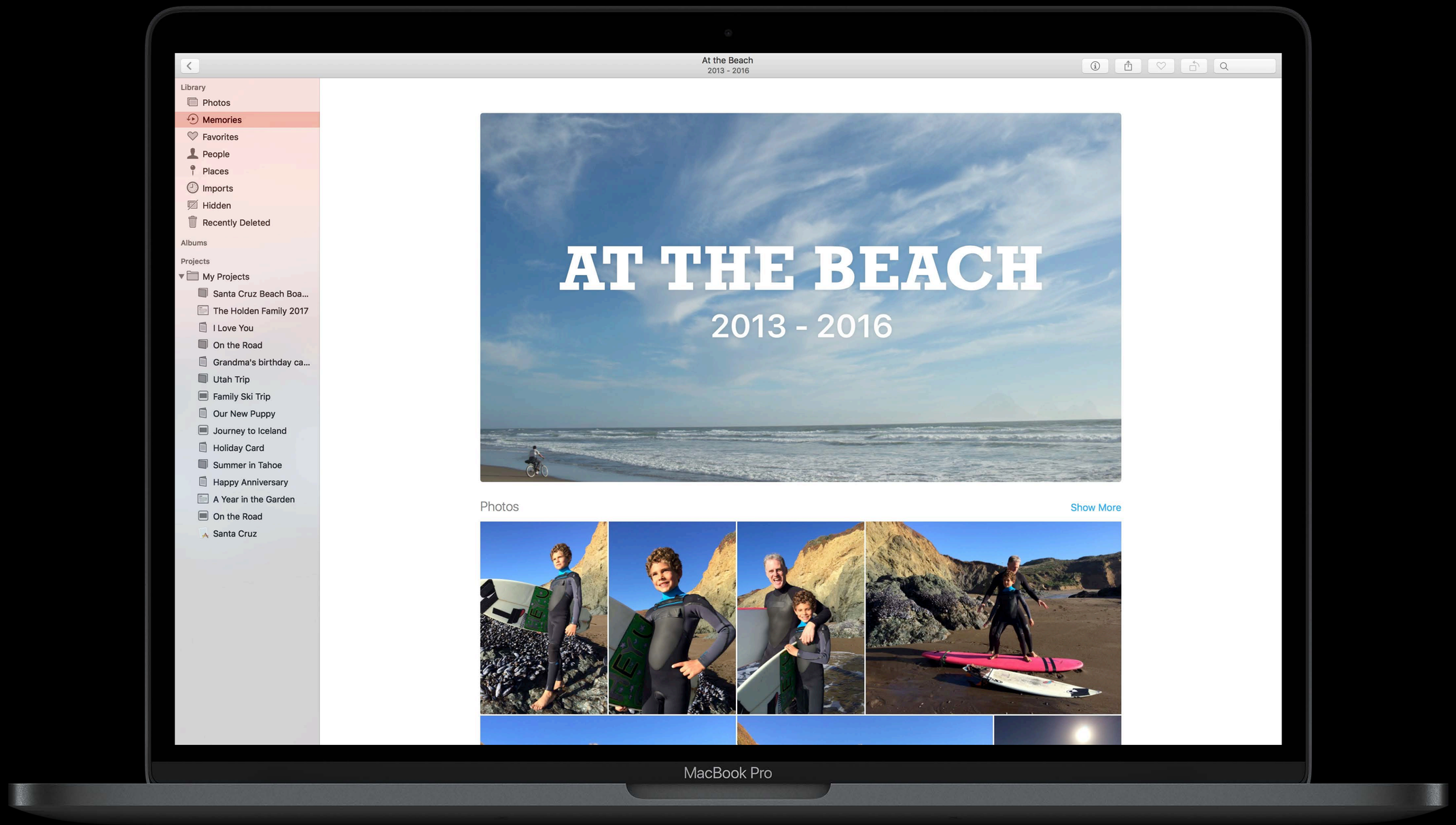
Title and key asset, and data

Extension’s project data

```
do {
    let changeRequest = PHProjectChangeRequest(project: self.project)

    try self.library.performChangesAndWait {
        changeRequest.projectExtensionData =
            NSKeyedArchiver.archivedData(withRootObject: cloudIdentifiers)
    }

} catch {
    print("Failed to save project data: \(error.localizedDescription)")
}
```

Where the magic begins


At the Beach
2013 - 2016
Library
Photos
Memories
Favorites
People
Places
Imports
Hidden
Recently Deleted
Albums
Projects
My Projects
Santa Cruz Beach Boa...
The Holden Family 2017
I Love You
On the Road
Grandma's birthday ca...
Utah Trip
Family Ski Trip
Our New Puppy
Journey to Iceland
Holiday Card
Summer in Tahoe
Happy Anniversary
A Year in the Garden
On the Road
Santa Cruz
AT THE BEACH
2013 - 2016
Photos
Show More
MacBook Pro


At the Beach
2013 - 2016
Library
Photos
Memories
Favorites
People
Places
Imports
Hidden
Recently Deleted
Albums
Projects
My Projects
Santa Cruz Beach Boa...
The Holden Family 2017
I Love You
On the Road
Grandma's birthday ca...
Utah Trip
Family Ski Trip
Our New Puppy
Journey to Iceland
Holiday Card
Summer in Tahoe
Happy Anniversary
A Year in the Garden
On the Road
Santa Cruz
AT THE BEACH
2013 - 2016
Photos
Show More
MacBook Pro

PHProjectInfo

All the context you need at project creation

PHProjectInfo

All the context you need at project creation

```
// PHProjectInfo
var sections: [PHProjectSection]
```

PHProjectInfo

All the context you need at project creation

```
// PHProjectInfo
var sections: [PHProjectSection]
```

```
// PHProjectSection
var sectionType: PHProjectSection.SectionType
var sectionContents: [PHProjectSectionContent]
```

PHProjectInfo

All the context you need at project creation

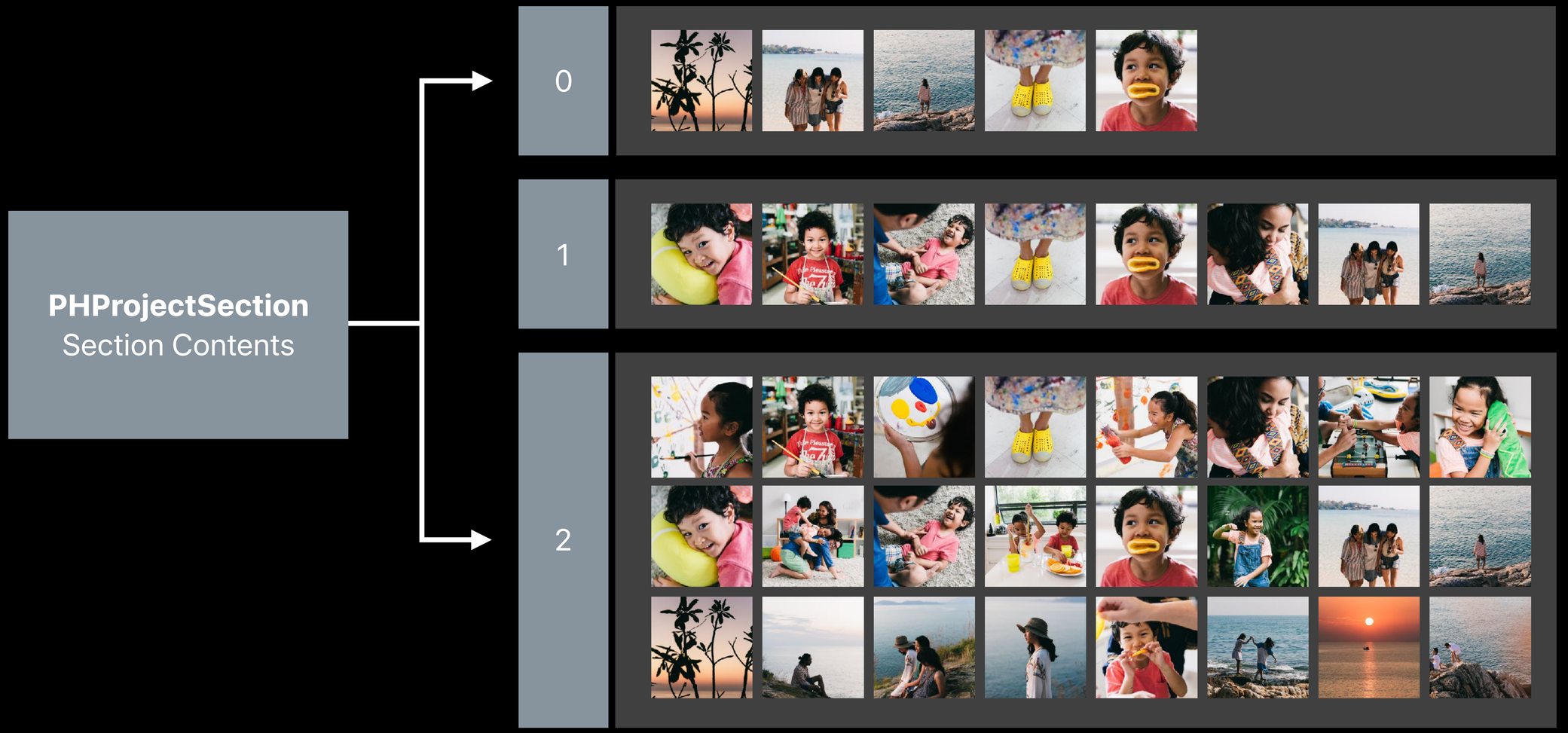
```
// PHProjectInfo
var sections: [PHProjectSection]
```

```
// PHProjectSection
var sectionType: PHProjectSection.SectionType
var sectionContents: [PHProjectSectionContent]
```

```
// PHProjectSectionContent
var elements: [PHProjectElement]
var numberOfColumns: Int
var aspectRatio: Double

var cloudAssetIdentifiers: [PHCloudIdentifier]
```

PHProjectSectionContent

Multiple content objects represent different curation levels



```
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

```
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

PHCloudIdentifier

Asset identifiers in PHProjectInfo are PHCloudIdentifiers

Convert to local identifiers to fetch PHAssets

Always persist cloud identifiers in saved project data

```
// get content cloud identifiers
cloudIdentifiers += dataDict.value(forKey: "contentIdentifiers") as! [PHCloudIdentifier]

// convert to local identifiers
let localIdentifiers = self.library.localIdentifiers(for: cloudIdentifiers)
```

PHCloudIdentifier

Asset identifiers in PHProjectInfo are PHCloudIdentifiers

Convert to local identifiers to fetch PHAssets

Always persist cloud identifiers in saved project data

```
// get content cloud identifiers
cloudIdentifiers += dataDict.value(forKey: "contentIdentifiers") as! [PHCloudIdentifier]
```

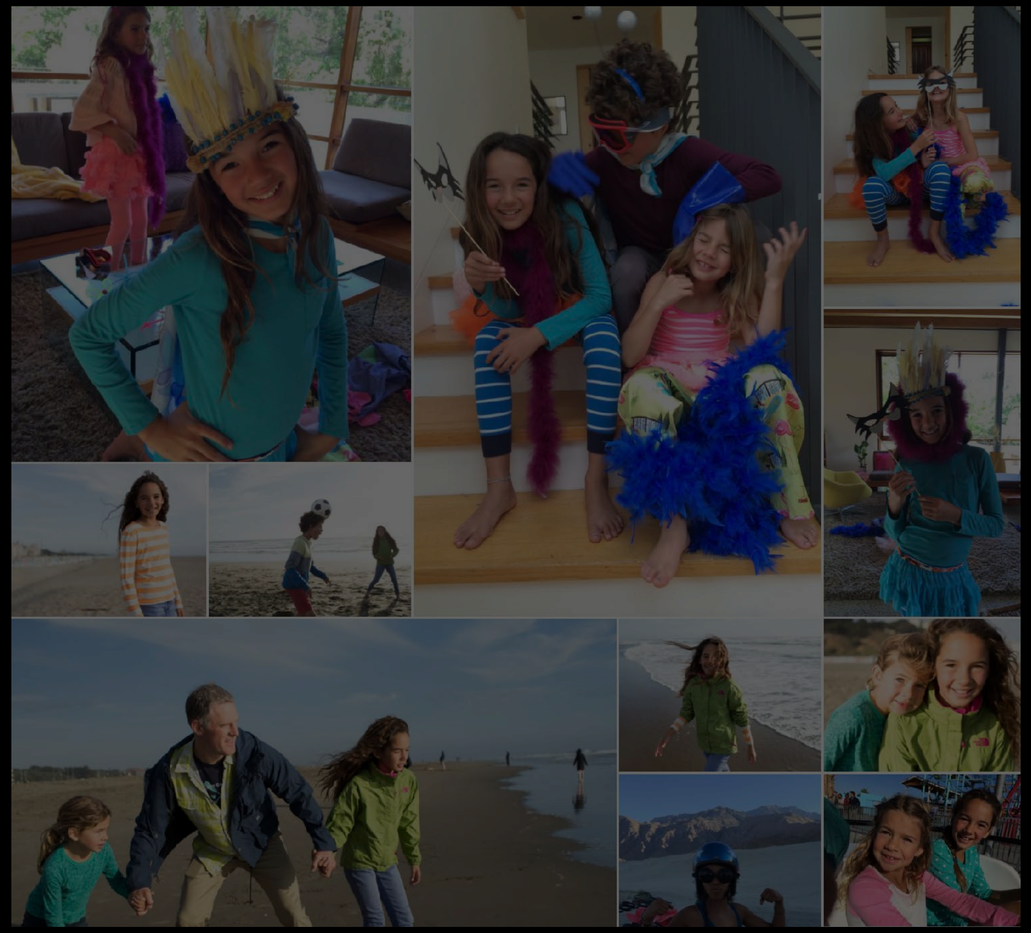
```
// convert to local identifiers
let localIdentifiers = self.library.localIdentifiers(for: cloudIdentifiers)
```

```
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

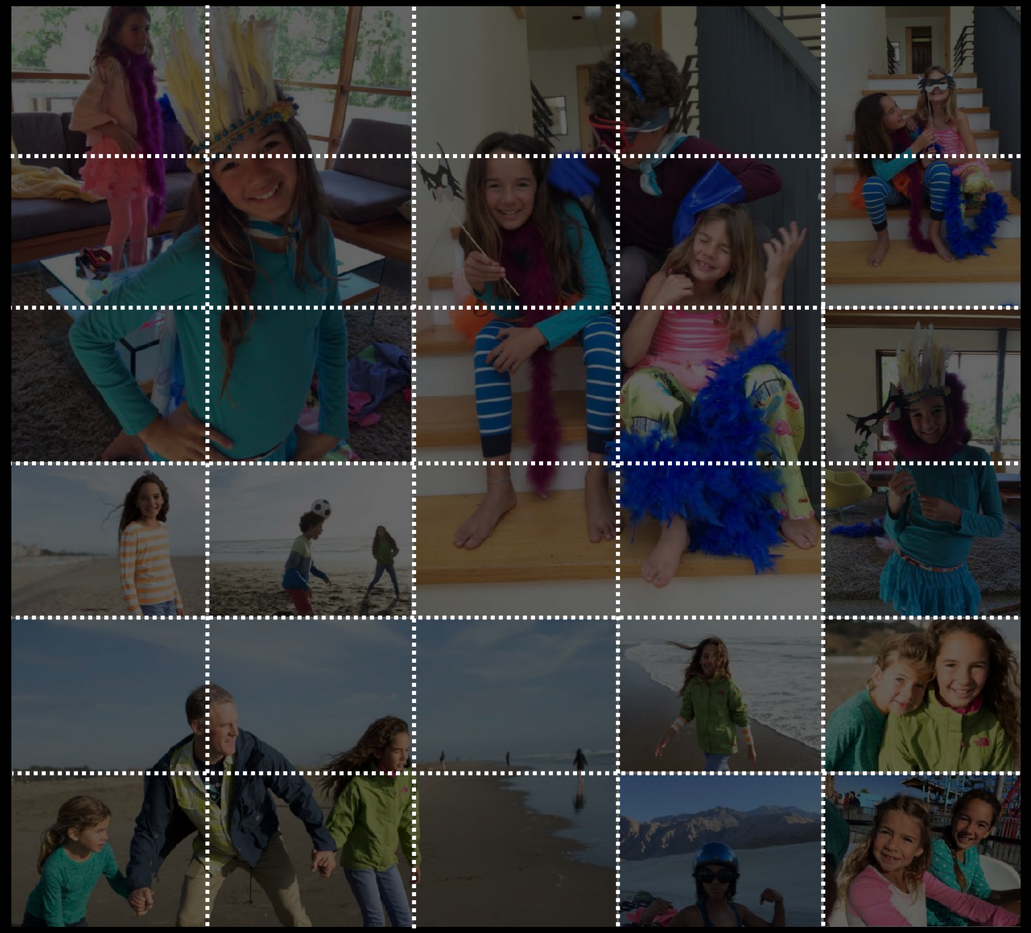
```
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

Grid Coordinates

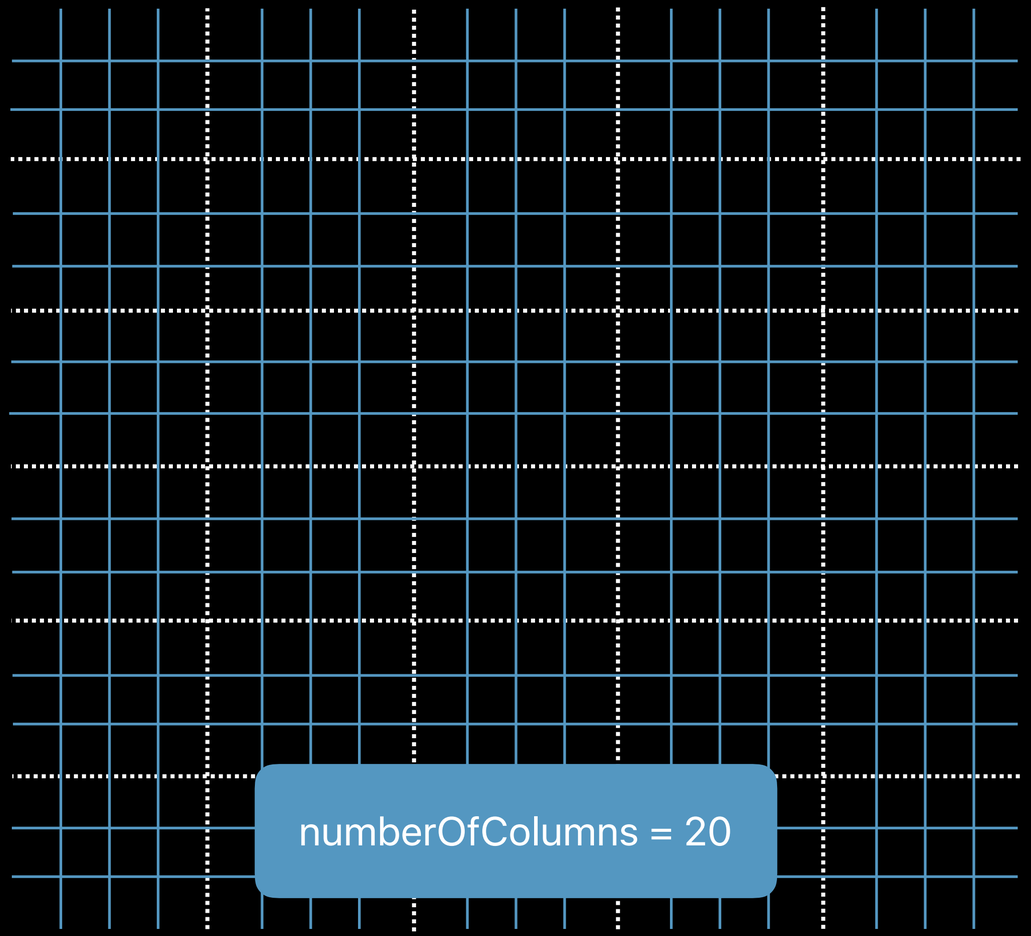
Grid Coordinates



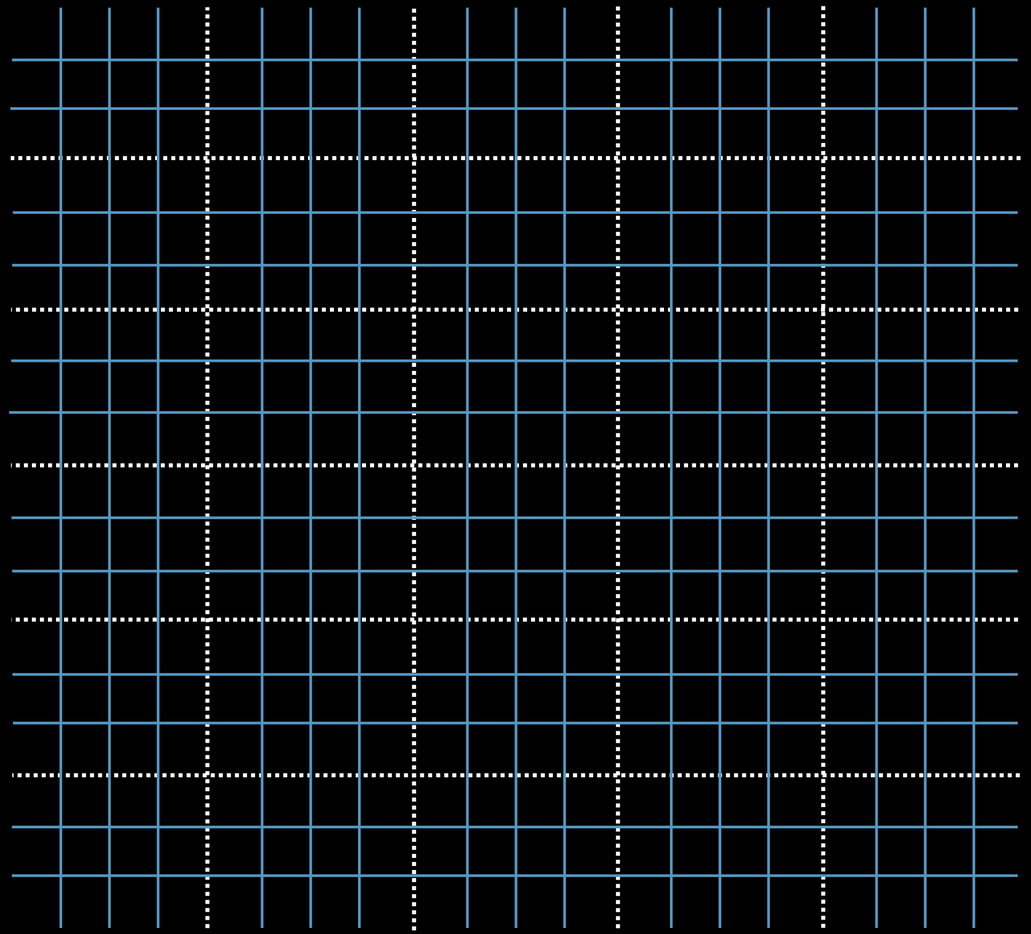
Grid Coordinates



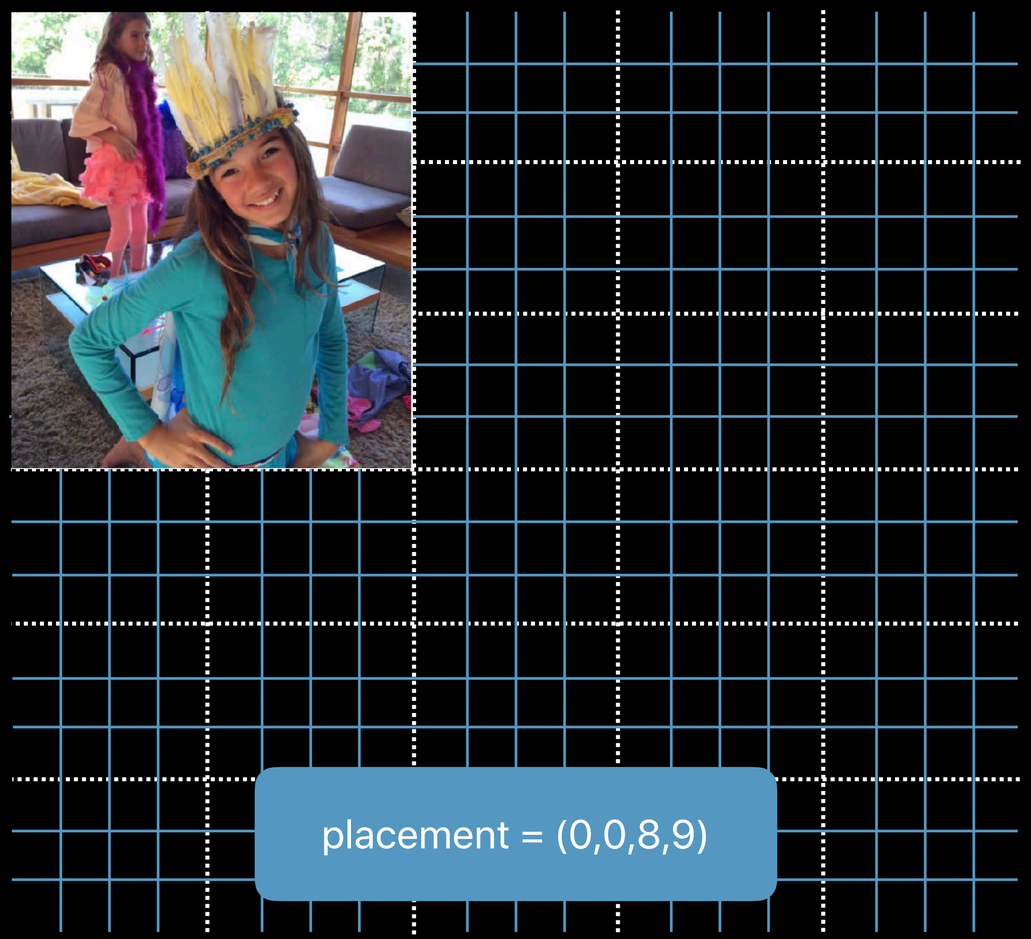
Grid Coordinates



Grid Coordinates



Grid Coordinates



```
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

```swift
// PHProjectSectionContent

class PHProjectSectionContent : NSObject, NSSecureCoding {
    var elements: [PHProjectElement] { get }
    var numberOfColumns: Int { get }
    var aspectRatio: Double { get }
    var cloudAssetIdentifiers: [PHCloudIdentifier] { get }
}
```

```
// PHProjectElement

class PHProjectElement : NSObject, NSSecureCoding {

    // Relative significance in range from 0.0 to 1.0. Default is 0.5.
    var weight: Double { get }


    // Positioning in grid space coordinates for element if honoring layout grid
    var placement: CGRect { get }

}
```

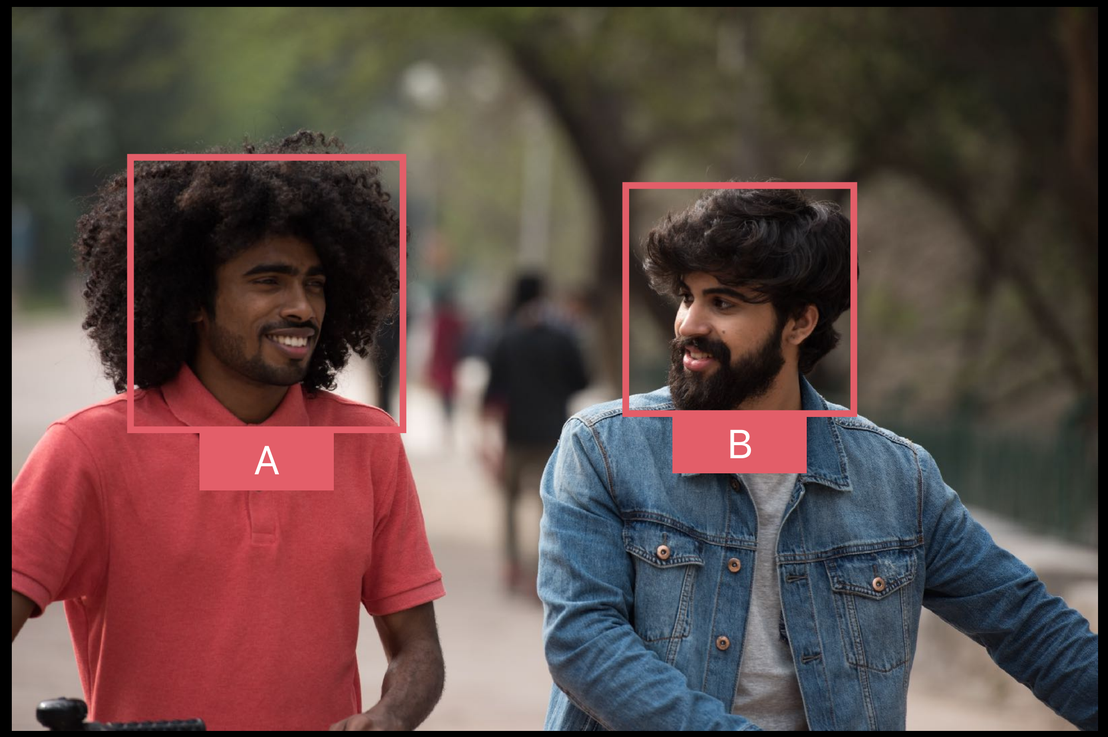
```
// PHProjectElement

class PHProjectElement : NSObject, NSSecureCoding {

    // Relative significance in range from 0.0 to 1.0. Default is 0.5.
    var weight: Double { get }


    // Positioning in grid space coordinates for element if honoring layout grid
    var placement: CGRect { get }

}
```

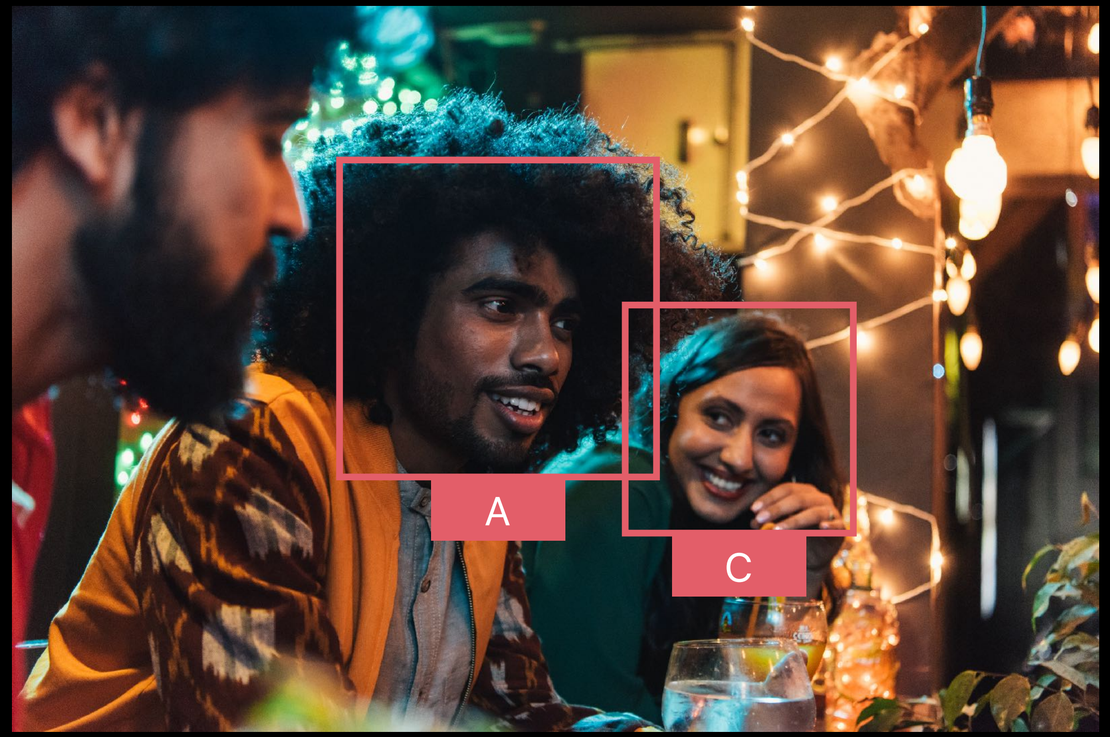
```
// PHProjectElement

class PHProjectElement : NSObject, NSSecureCoding {

    // Relative significance in range from 0.0 to 1.0. Default is 0.5.
    var weight: Double { get }


    // Positioning in grid space coordinates for element if honoring layout grid
    var placement: CGRect { get }

}
```

```
// PHProjectAssetElement

class PHProjectAssetElement : PHProjectElement, NSSecureCoding {

    // convert to local identifier to fetch PHAsset
    var cloudAssetIdentifier: PHCloudIdentifier { get }
    var annotation: String { get }
    var cropRect: CGRect { get }
    var regionsOfInterest: [PHProjectRegionOfInterest] { get }

}
```

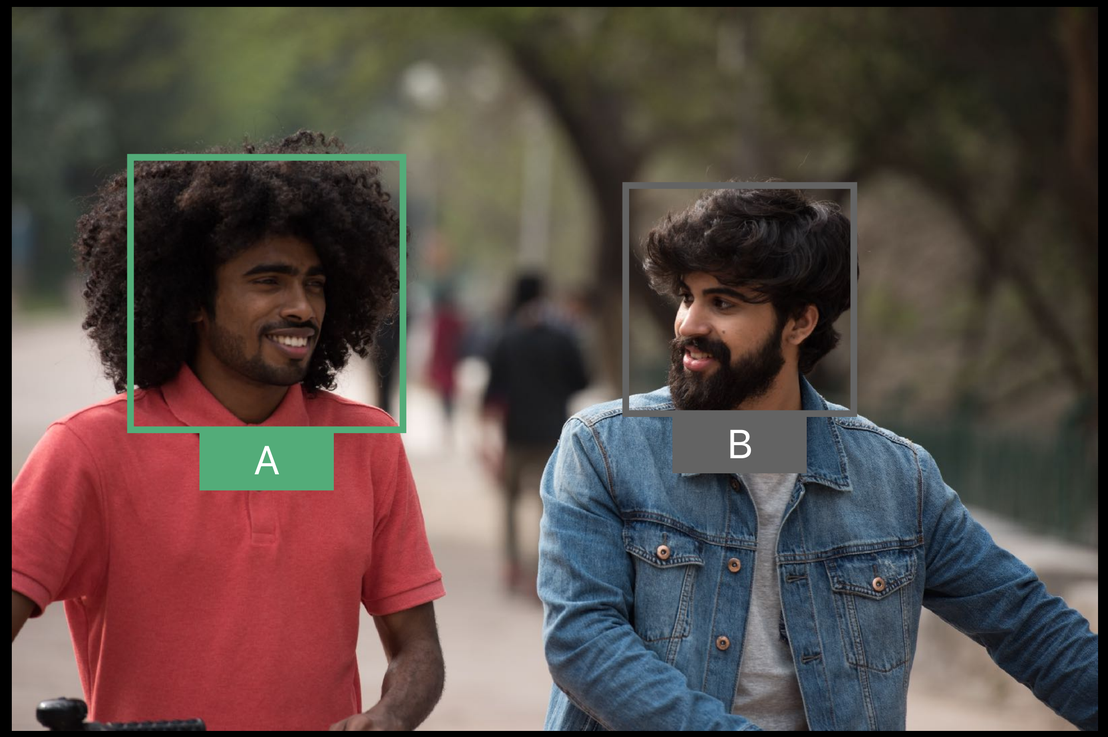
```
// PHProjectAssetElement

class PHProjectAssetElement : PHProjectElement, NSSecureCoding {

    // convert to local identifier to fetch PHAsset
    var cloudAssetIdentifier: PHCloudIdentifier { get }
    var annotation: String { get }
    var cropRect: CGRect { get }
    var regionsOfInterest: [PHProjectRegionOfInterest] { get }

}
```

PHProjectRegionOfInterest





PHProjectRegionOfInterest



Demo

Let’s build our own


Photos - Picture Show
Done
MacBook Pro

Summary

Take advantage of frictionless photo picker

Use new media presentation types

Test with large libraries

Create photo services with Photos Project Extensions

More Information

https://developer.apple.com/wwdc17/505

Related Sessions

Vision Framework: Building on Core ML	Hall 2	Wednesday 3:10PM
Capturing Depth in iPhone Photography	Executive Ballroom	Wednesday 5:10PM
Photography Get Together	Technology Lab J	Wednesday 6:30PM
Imaging Editing with Depth	Grand Ballroom A	Thursday 11:00AM
Advances in Core Image: Filters, Metal, Vision, and More	Executive Ballroom	Thursday 1:50PM
Working with HEIF and HEVC	Hall 2	Friday 11:00AM

Related Sessions

Vision Framework: Building on Core ML	Hall 2	Wednesday 3:10PM
Capturing Depth in iPhone Photography	Executive Ballroom	Wednesday 5:10PM
Photography Get Together	Technology Lab J	Wednesday 6:30PM
Imaging Editing with Depth	Grand Ballroom A	Thursday 11:00AM
Advances in Core Image: Filters, Metal, Vision, and More	Executive Ballroom	Thursday 1:50PM
Working with HEIF and HEVC	Hall 2	Friday 11:00AM

Labs

Photos Extensions and PhotoKit Lab	Technology Lab A	Thur 9:00AM–12:00PM
Photos Editing and Core Image Lab	Technology Lab F	Thur 3:10PM–6:00PM
Photos Depth and Capture Lab	Technology Lab A	Thur 3:10PM–6:00PM
HEIF and HEVC Lab	Technology Lab F	Fri 12:00PM–1:50PM
Vision Lab	Technology Lab A	Fri 1:50PM–4:00PM
Photos Depth and Capture Lab	Technology Lab F	Fri 1:50PM–4:00PM


WWDC17