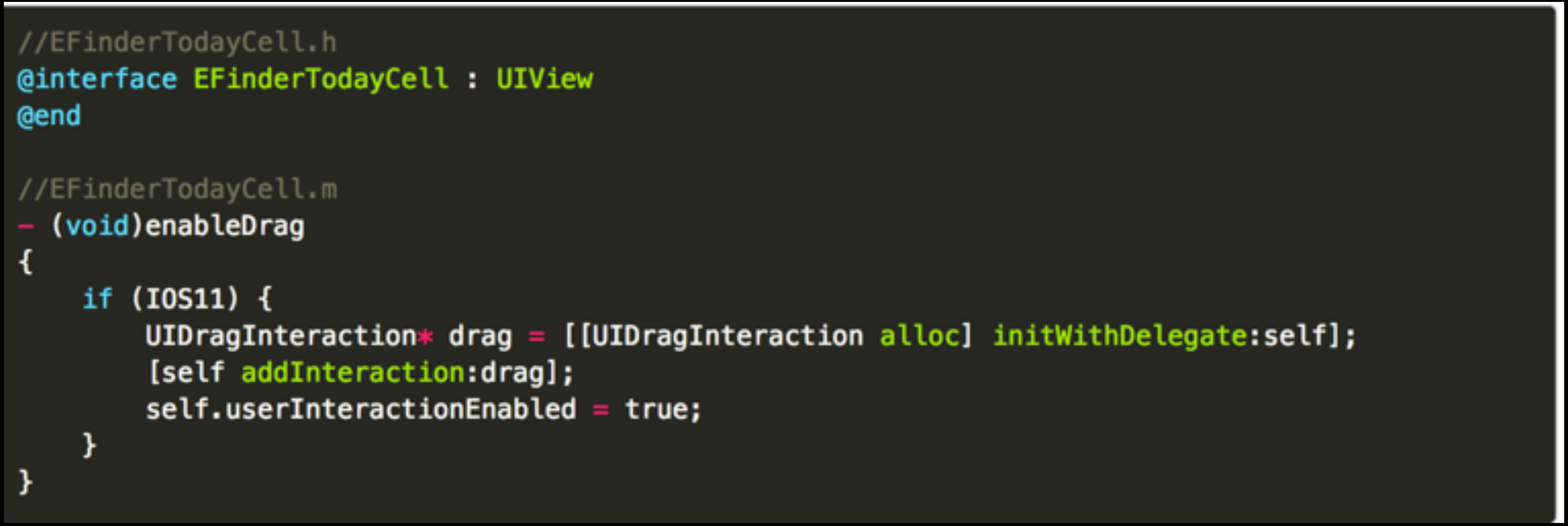
iOS 11新特性

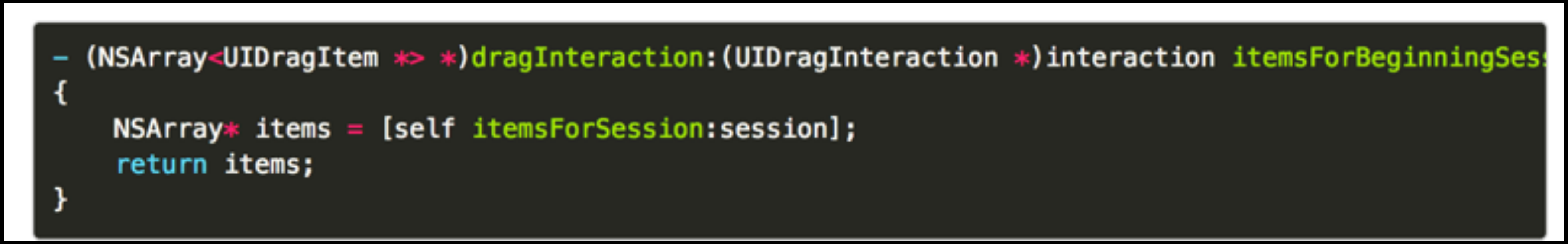
nate

- CocoaTouch
- Xcode9
- Core ML
- ARKit

Drag and Drop



```
//EFinderTodayCell.h
@interface EFinderTodayCell : UIView
@end

//EFinderTodayCell.m
- (void)enableDrag
{
    if (IOS11) {
        UIDragInteraction* drag = [[UIDragInteraction alloc] initWithDelegate:self];
        [self addInteraction:drag];
        self.userInteractionEnabled = true;
    }
}
```



```
- (NSArray<UIDragItem *> *)dragInteraction:(UIDragInteraction *)interaction itemsForBeginningSes
{
    NSArray* items = [self itemsForSession:session];
    return items;
}
```

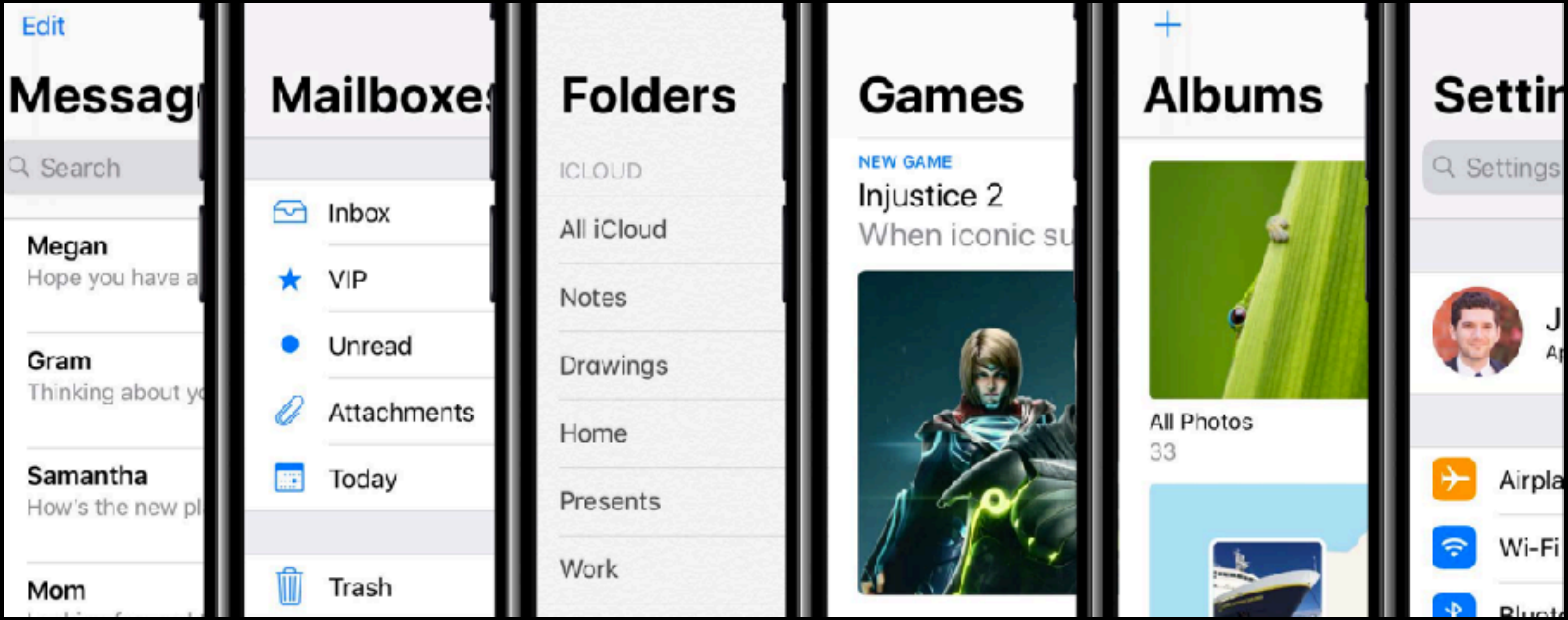
FileProviderUI

File Management

UIDocumentBrowserViewController

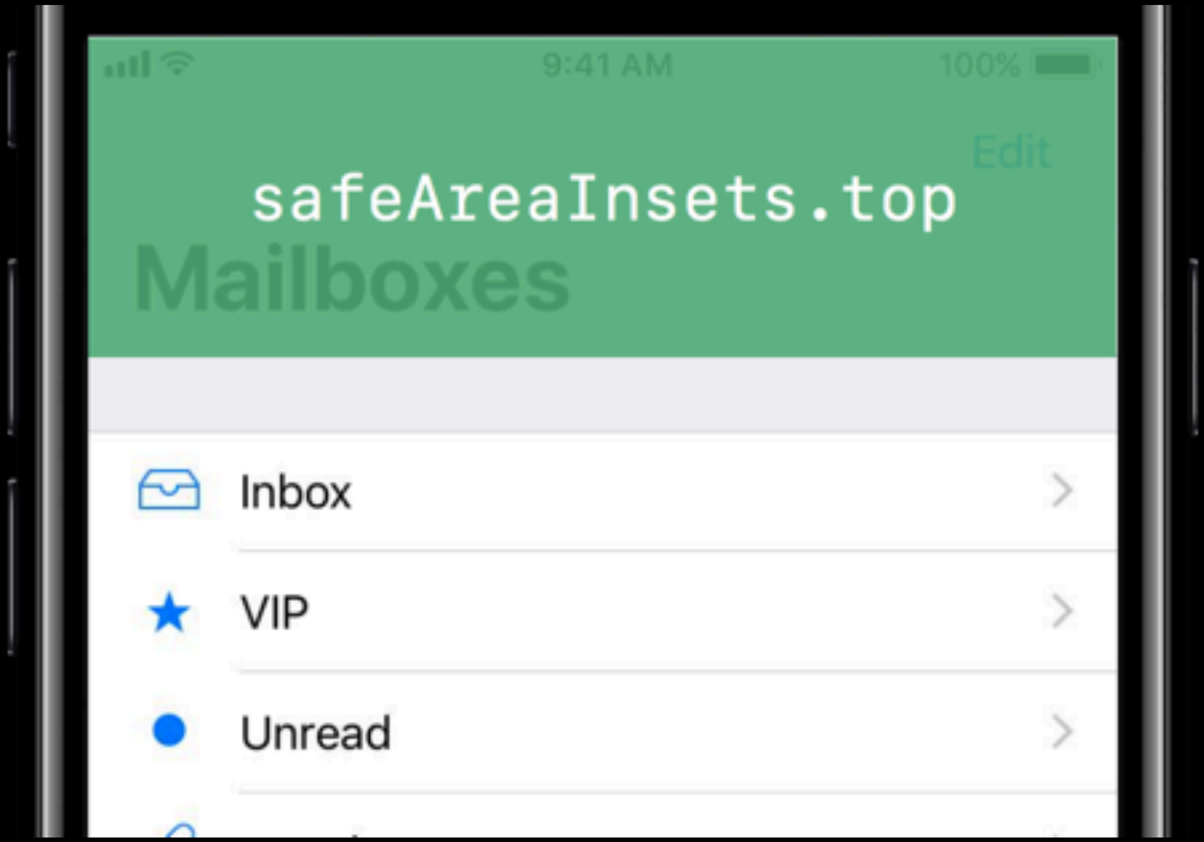
```
class UIDocumentBrowserViewController {
  init(forOpeningFilesWithContentTypes: [String]?)
  var delegate: UIDocumentBrowserViewControllerDelegate?
}
```

Navigation title



Enabling Large Titles

```
class UINavigationBar {
        var prefersLargeTitle: Bool
}
```



`safeAreaInsets.top`

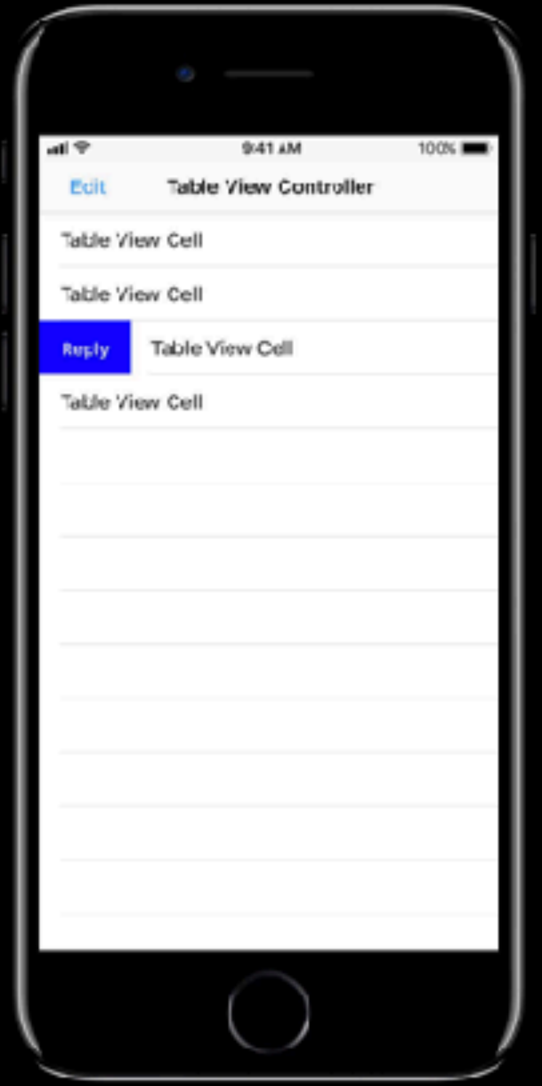
Swipe Actions

New look-and-feel automatically for all table views

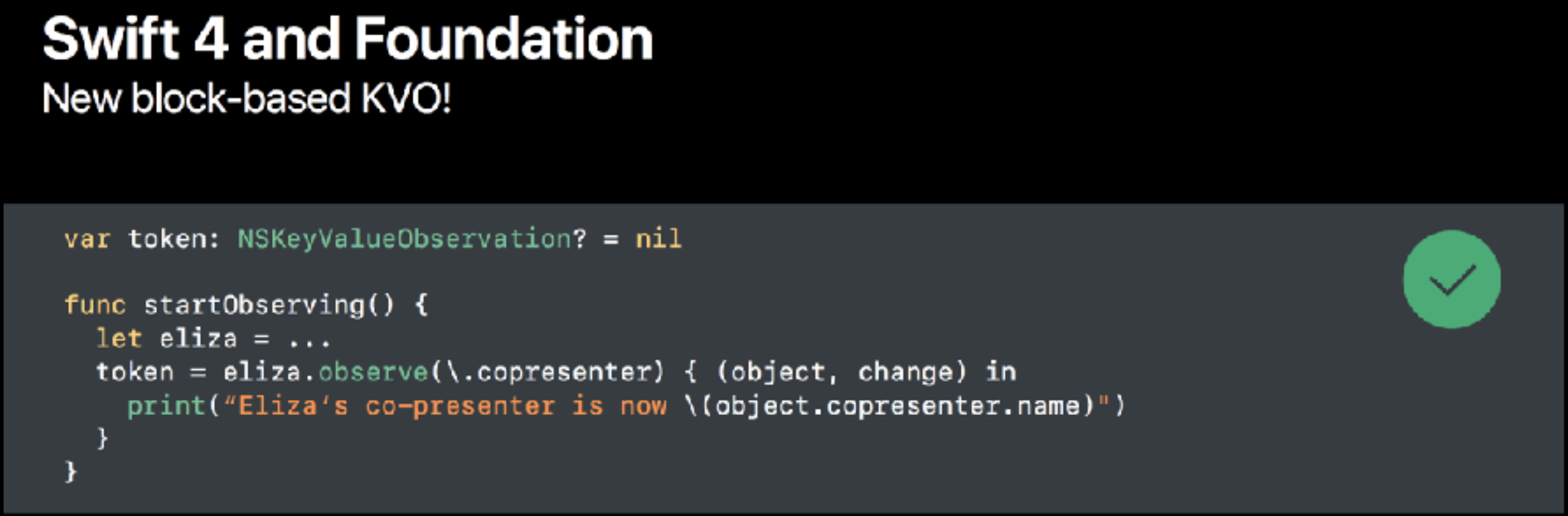
Supports full swipe-to-delete for iOS 11-linked apps

New features with API adoption

- Images
- Leading and trailing actions
- Completion handler and cancellation



KVO block

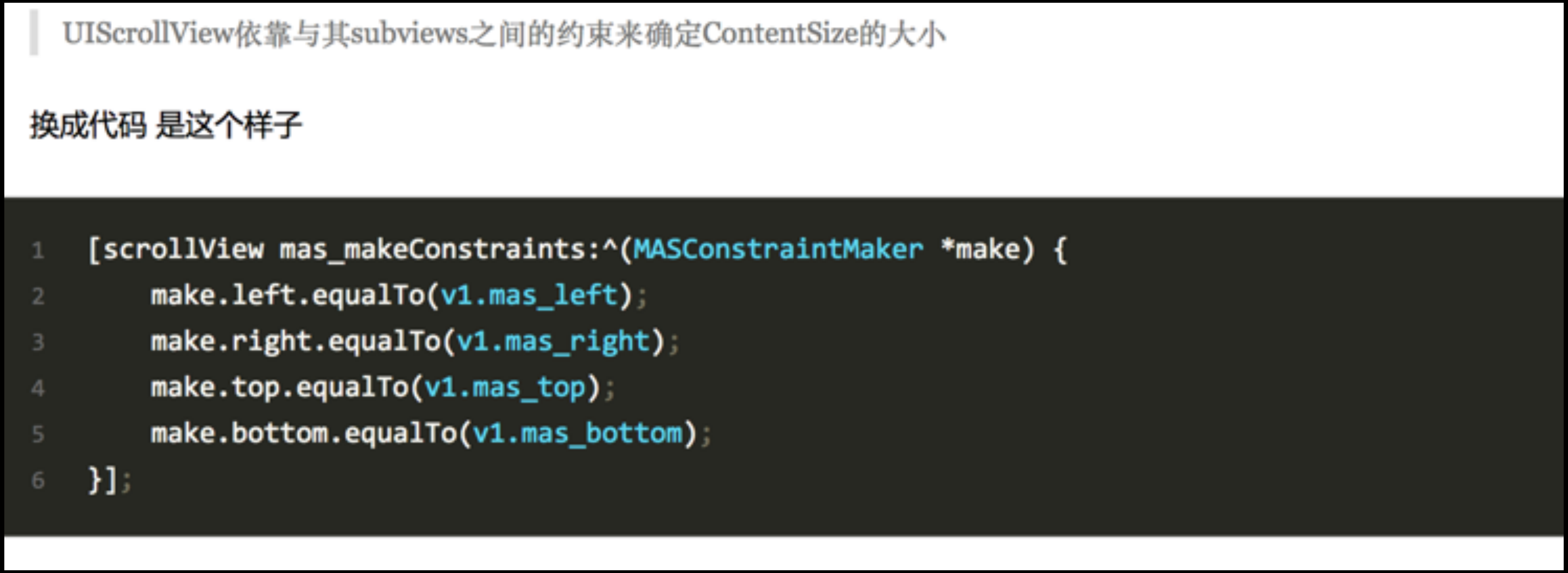


Swift 4 and Foundation

New block-based KVO!

```swift
var token: NSKeyValueObservation? = nil

func startObserving() {
  let eliza = ...
  token = eliza.observe(\.copresenter) { (object, change) in
    print(“Eliza‘s co-presenter is now \(object.copresenter.name)")
  }
}
```

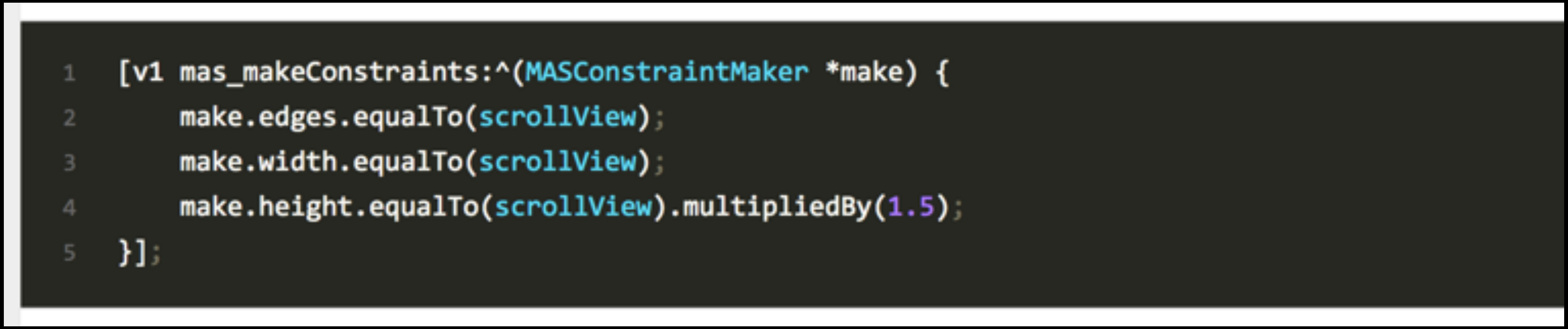
scrollview 下的autolayout



UIScrollView依靠与其subviews之间的约束来确定ContentSize的大小

换成代码 是这个样子

```
[scrollView mas_makeConstraints:^(MASConstraintMaker *make) {
    make.left.equalTo(v1.mas_left);
    make.right.equalTo(v1.mas_right);
    make.top.equalTo(v1.mas_top);
    make.bottom.equalTo(v1.mas_bottom);
}];
```



```
[v1 mas_makeConstraints:^(MASConstraintMaker *make) {
    make.edges.equalTo(scrollView);
    make.width.equalTo(scrollView);
    make.height.equalTo(scrollView).multipliedBy(1.5);
}];
```

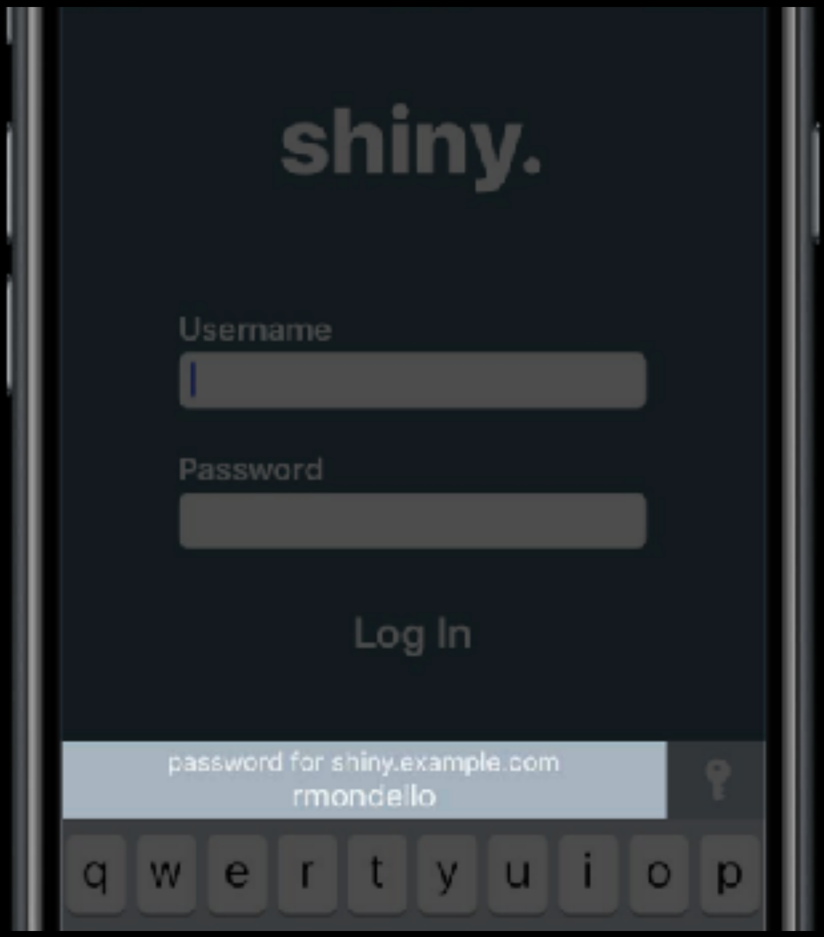
Auto Layout and Scroll View

Frame vs. content

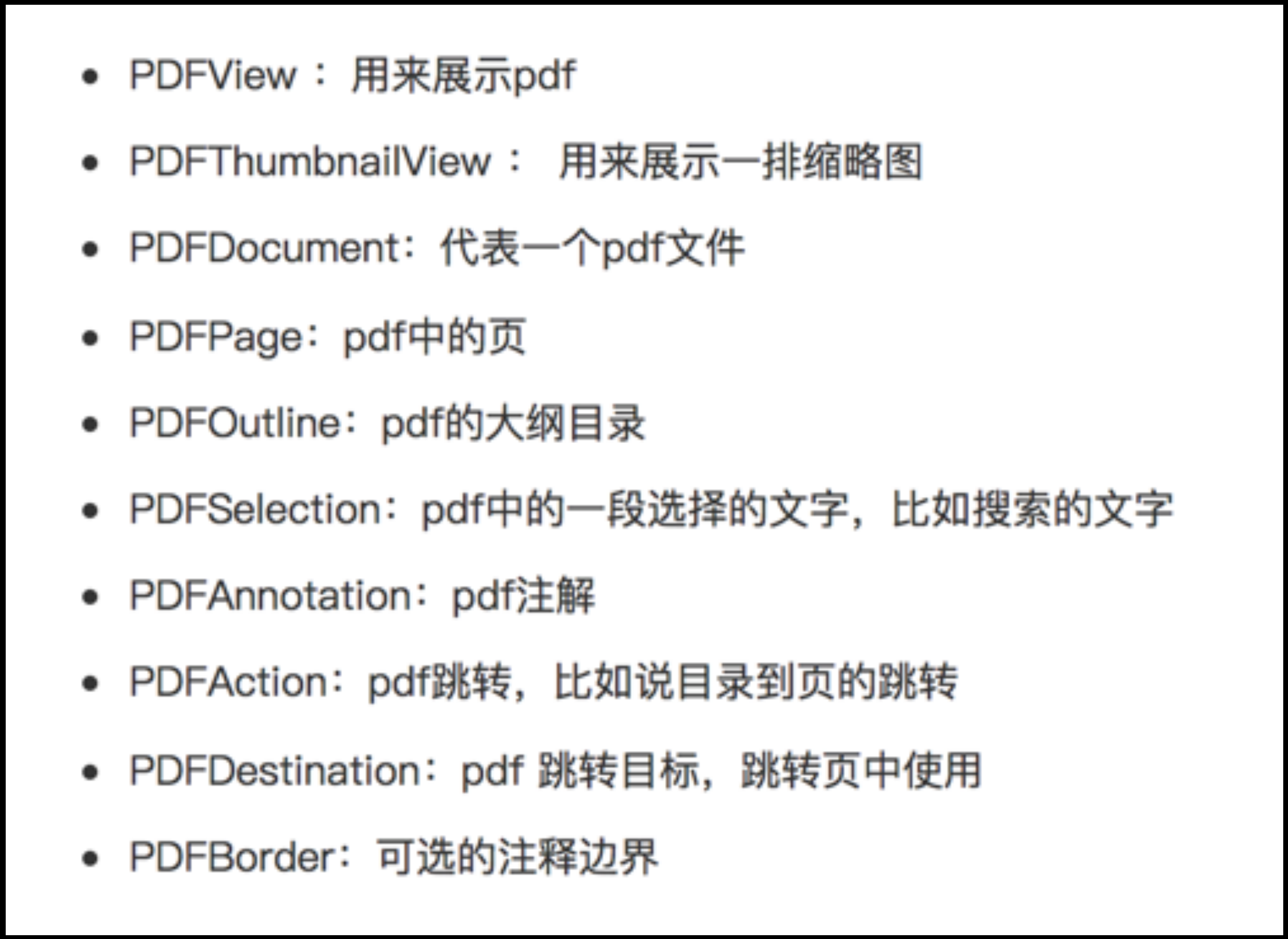
```
class UIScrollView {

  var contentLayoutGuide: UILayoutGuide { get }

  var frameLayoutGuide: UILayoutGuide { get }

}
```

```
imageView.centerXAnchor.constraint(equalTo: scrollView.contentLayoutGuide.centerXAnchor)
imageView.centerYAnchor.constraint(equalTo: scrollView.contentLayoutGuide.centerYAnchor)
```

iCloud 密码自动填充app

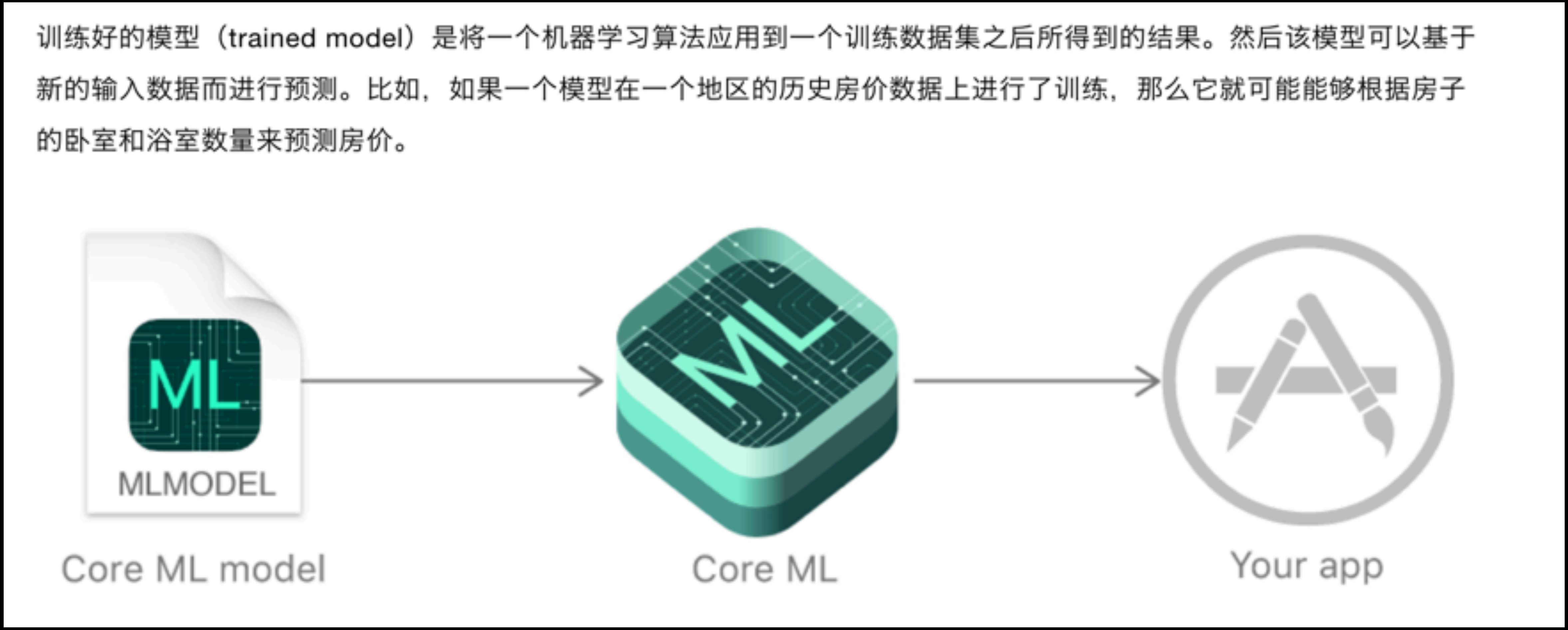


PDFKit



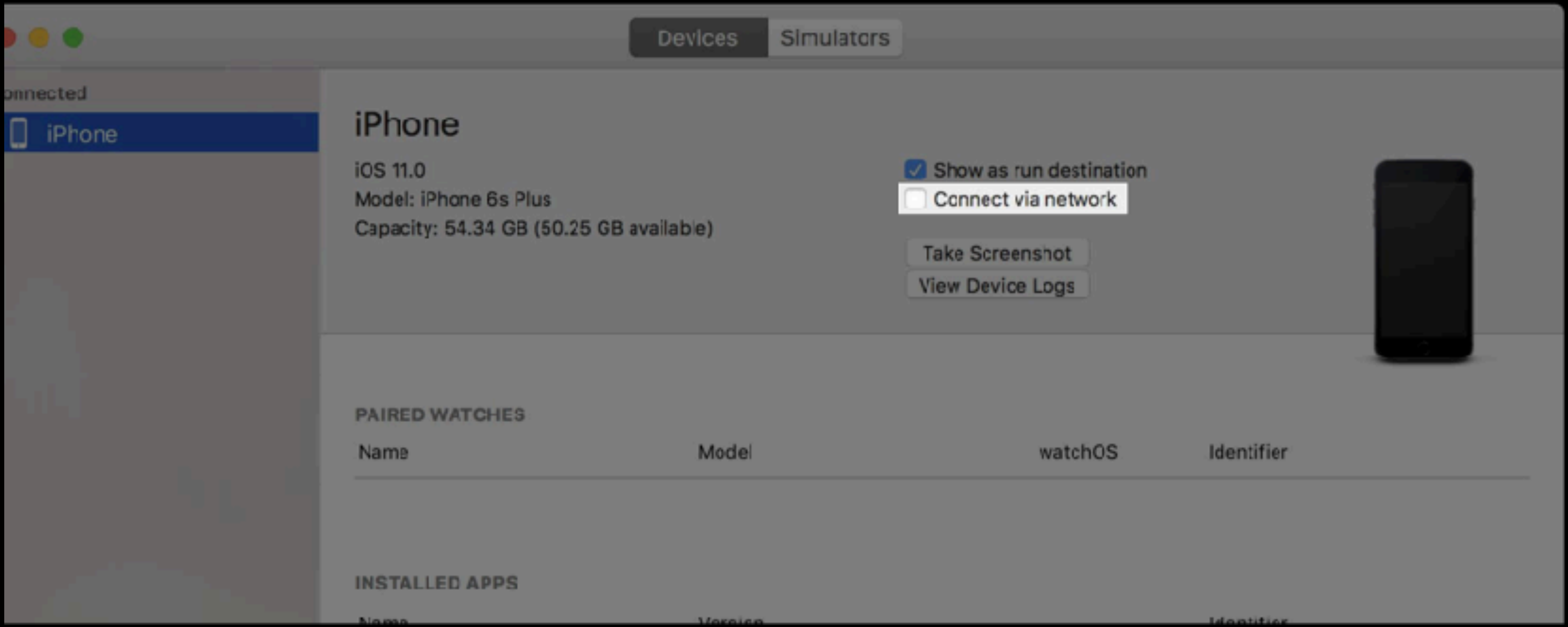
- PDFView： 用来展示pdf
- PDFThumbnailView： 用来展示一排缩略图
- PDFDocument：代表一个pdf文件
- PDFPage：pdf中的页
- PDFOutline：pdf的大纲目录
- PDFSelection：pdf中的一段选择的文字，比如搜索的文字
- PDFAnnotation：pdf注解
- PDFAction：pdf跳转，比如说目录到页的跳转
- PDFDestination：pdf 跳转目标，跳转页中使用
- PDFBorder：可选的注释边界

Core ML



训练好的模型（trained model）是将一个机器学习算法应用到一个训练数据集之后所得到的结果。然后该模型可以基于新的输入数据而进行预测。比如，如果一个模型在一个地区的历史房价数据上进行了训练，那么它就可能能够根据房子的卧室和浴室数量来预测房价。

无线debug

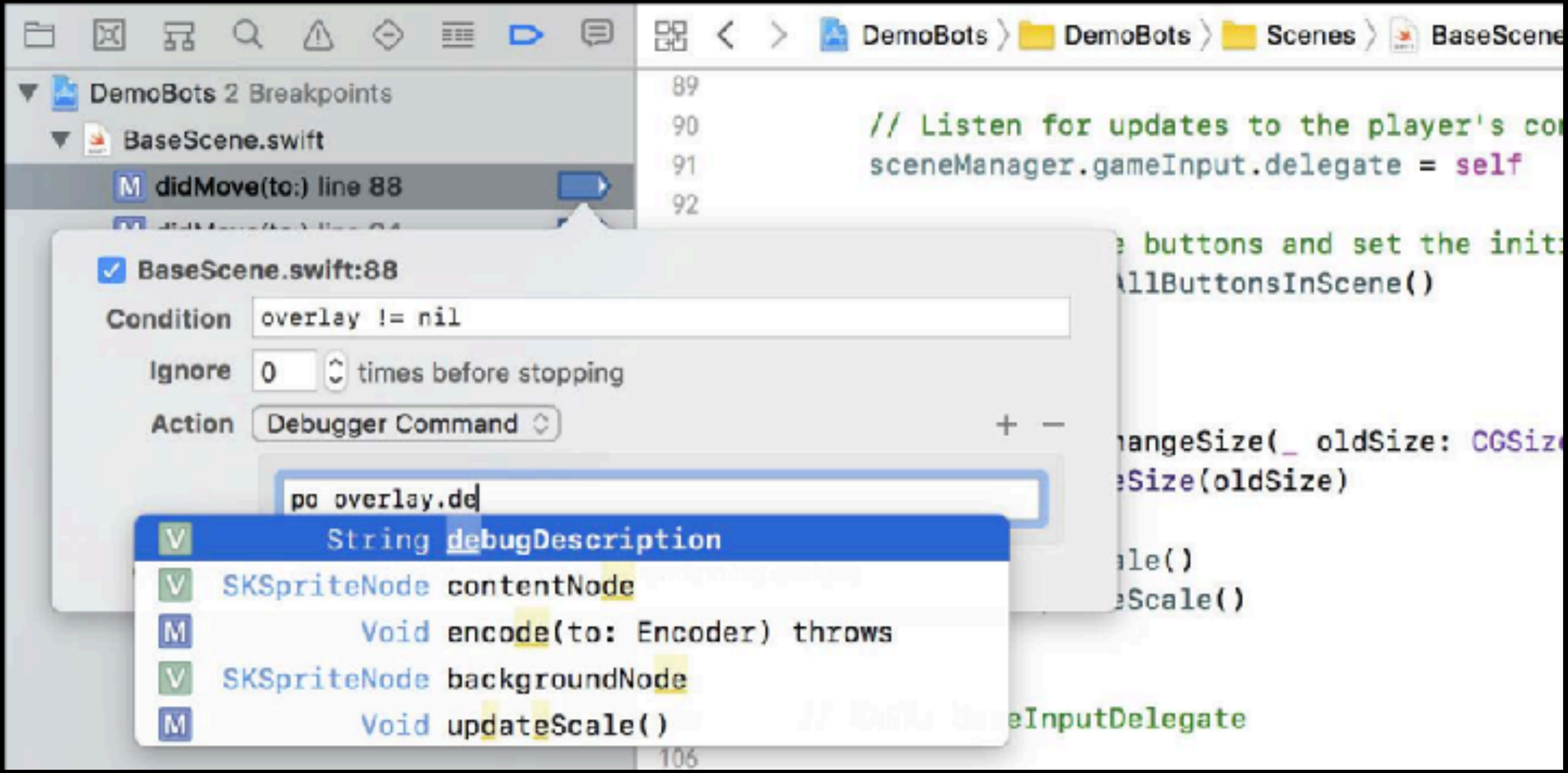


iPhone

iOS 11.0
Model: iPhone 6s Plus
Capacity: 54.34 GB (50.25 GB available)

[✓] Show as run destination
[ ] Connect via network

断点条件代码补全



DemoBots 2 Breakpoints
BaseScene.swift
didMove(to:) line 88

BaseScene.swift:88

Condition `overlay != nil`

Ignore 0 times before stopping

Action Debugger Command

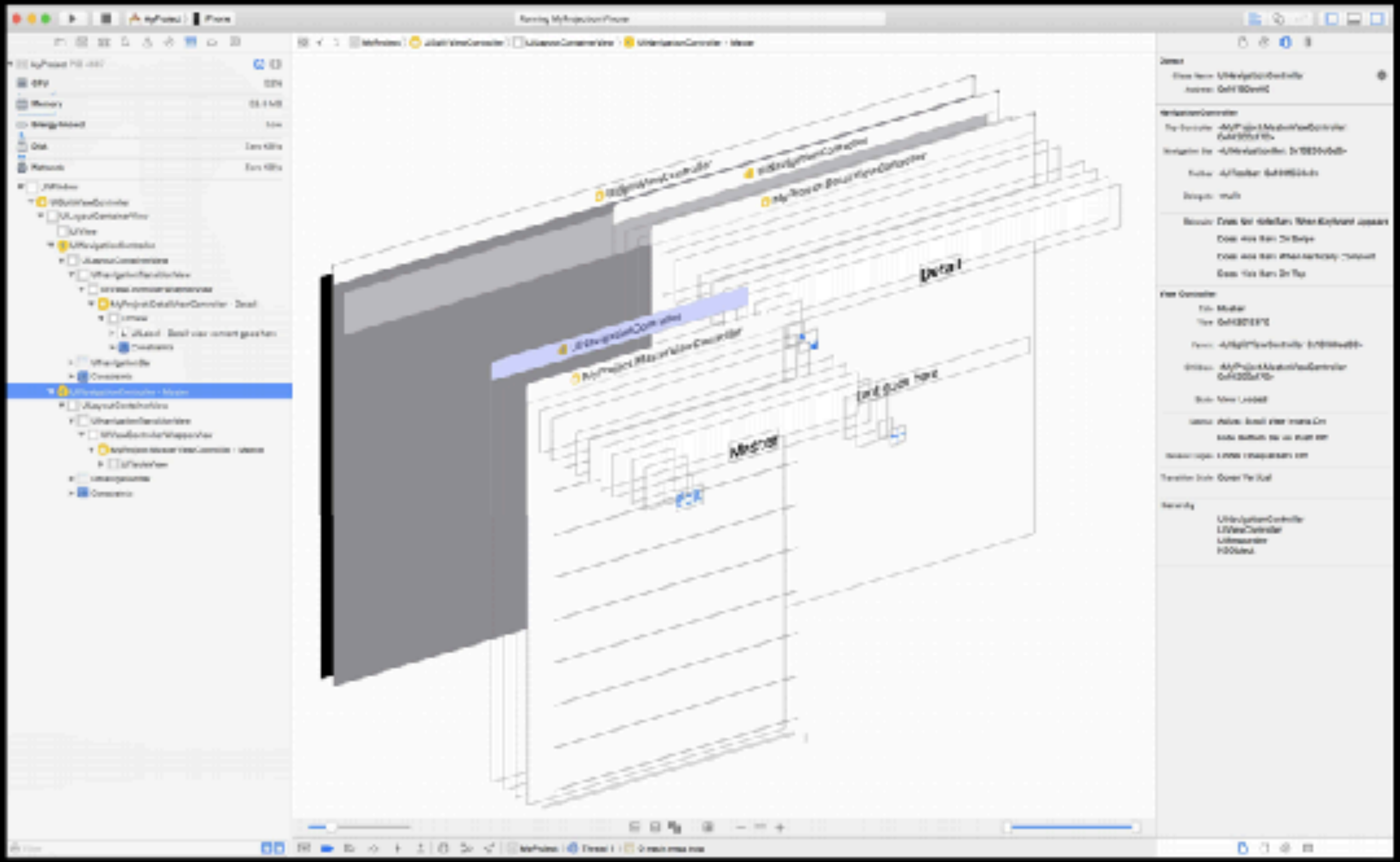
```
po overlay.de
```

```
String debugDescription
SKSpriteNode contentNode
Void encode(to: Encoder) throws
SKSpriteNode backgroundNode
Void updateScale()
```

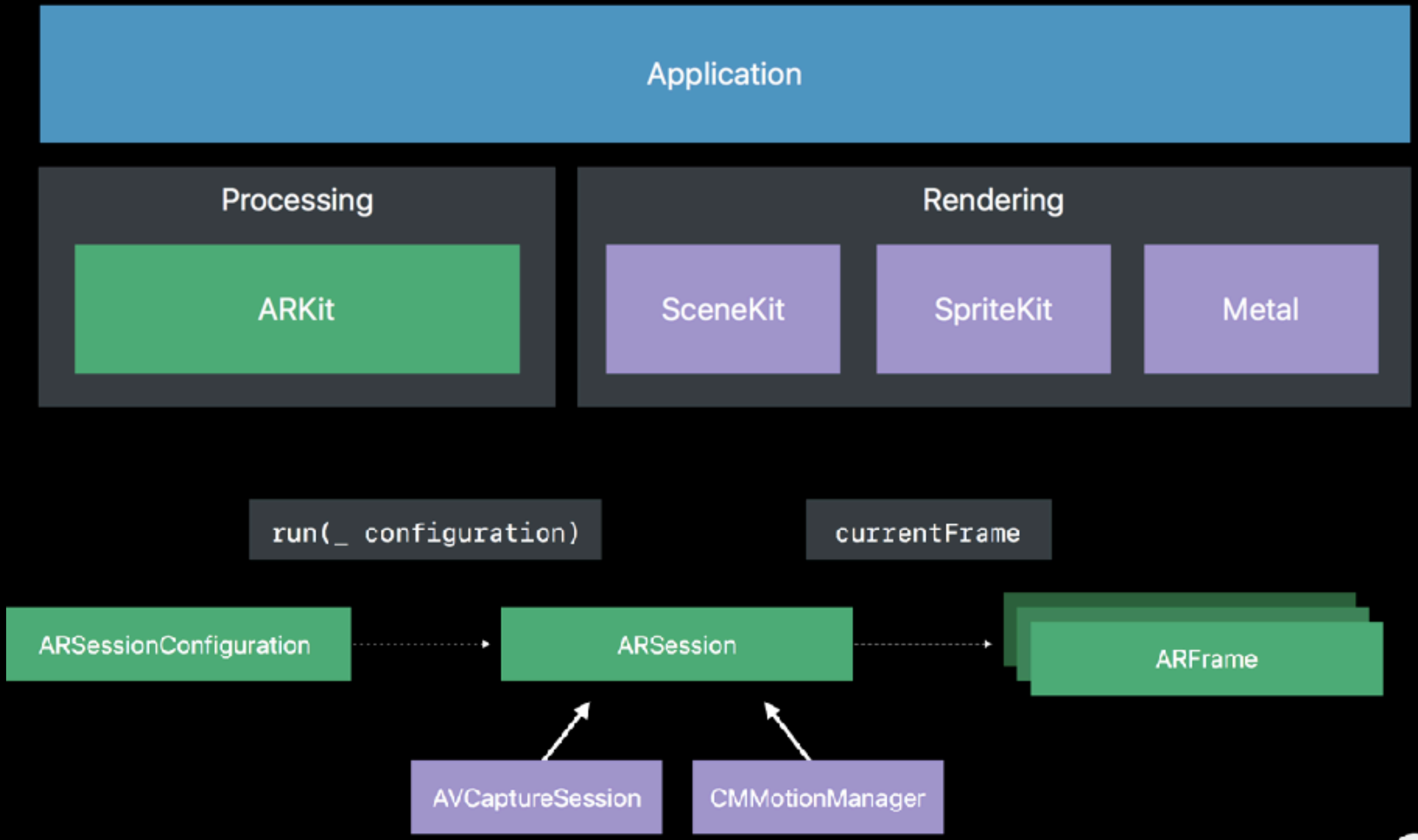
```
// Listen for updates to the player's co
sceneManager.gameInput.delegate = self
```

VC debug

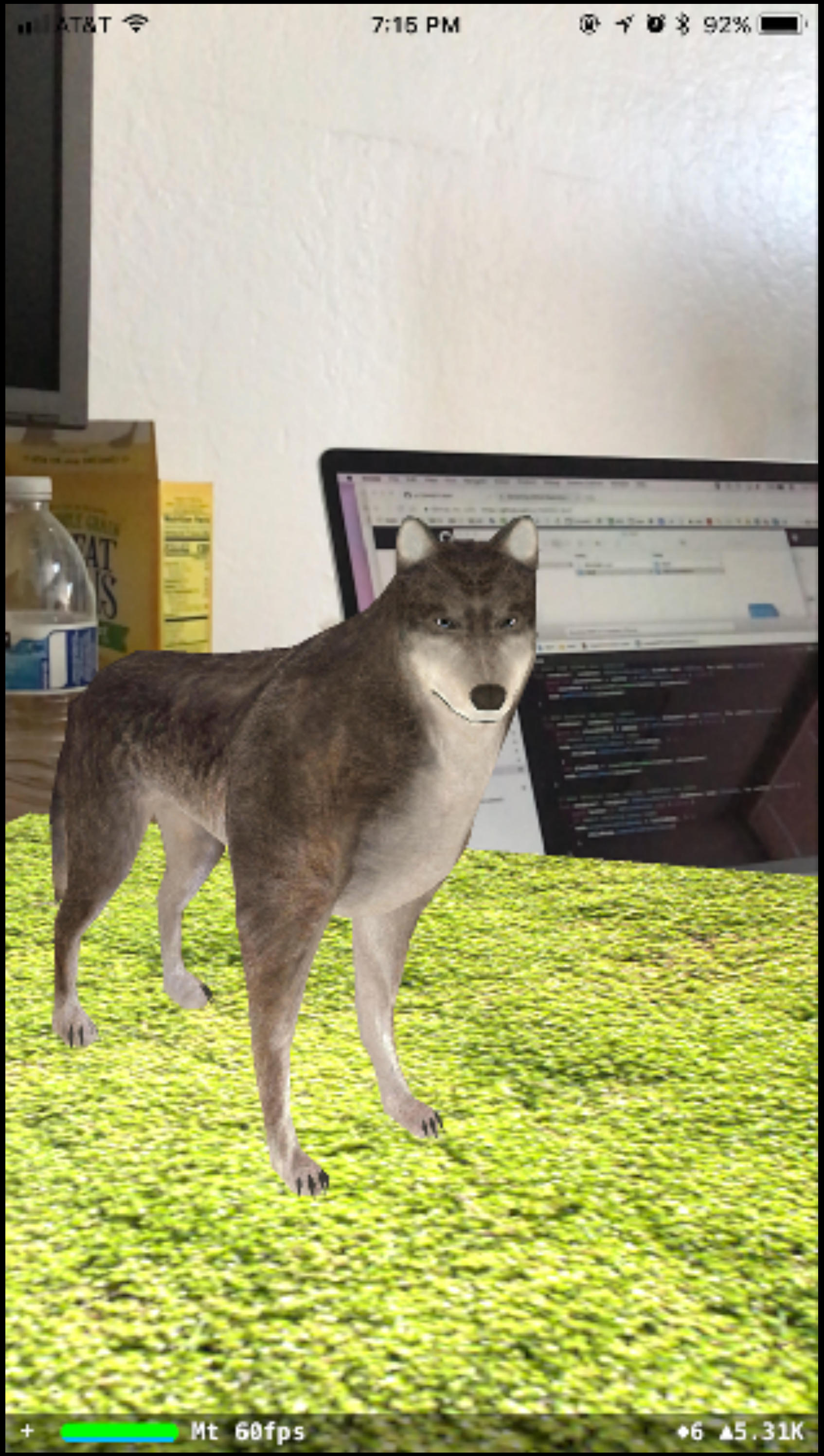
iew Controller Debugging

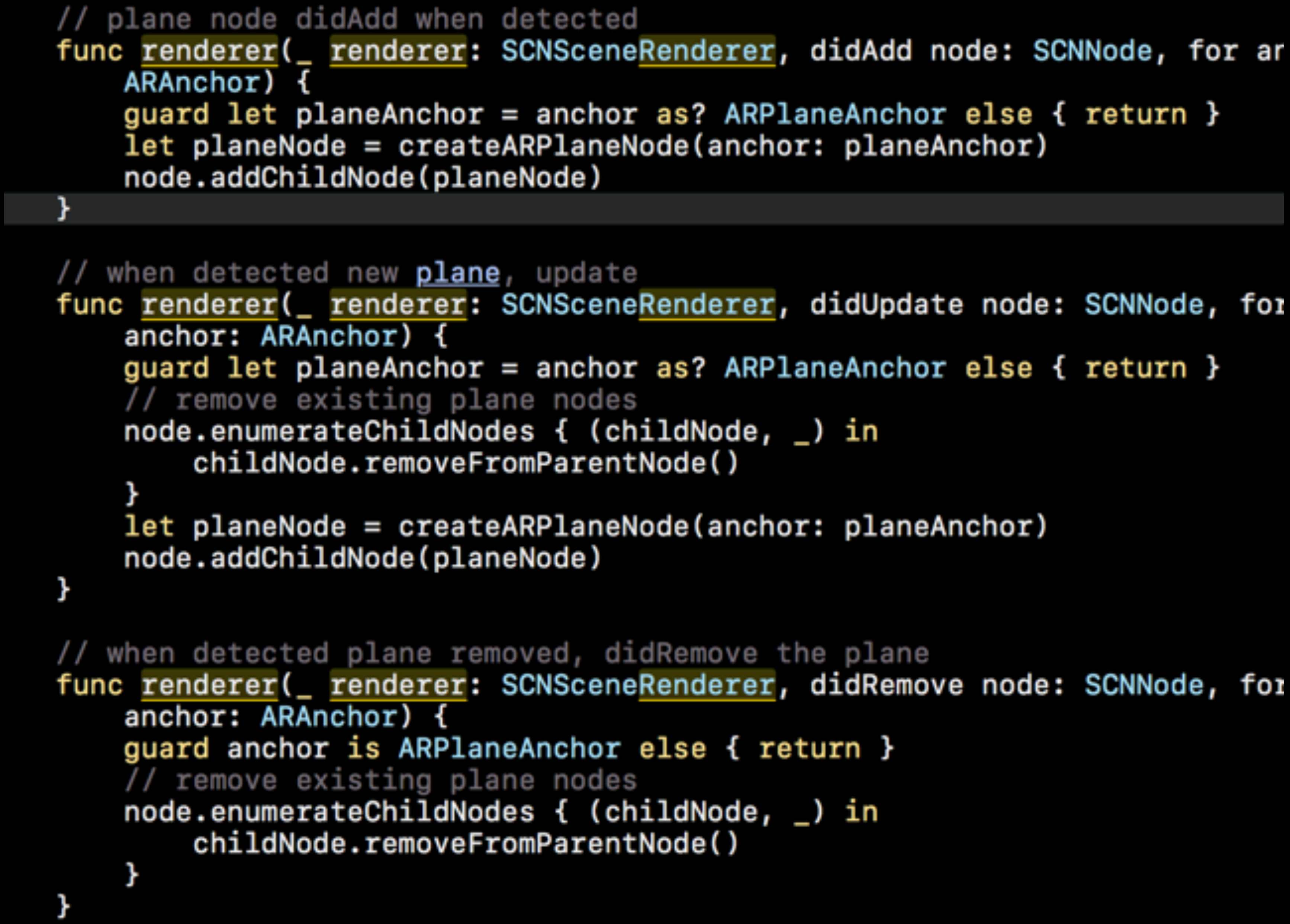


ARKit



Application
Processing
ARKit
Rendering
SceneKit
SpriteKit
Metal
run(_ configuration)
currentFrame
ARSessionConfiguration
ARSession
ARFrame
AVCaptureSession
CMMotionManager


AT&T
7:15 PM
92%
Mt 60fps
♦6 ▲5.31K



```
// plane node didAdd when detected
func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNode, for ar
    ARAnchor) {
    guard let planeAnchor = anchor as? ARPlaneAnchor else { return }
    let planeNode = createARPlaneNode(anchor: planeAnchor)
    node.addChildNode(planeNode)
}

// when detected new plane, update
func renderer(_ renderer: SCNSceneRenderer, didUpdate node: SCNNode, for
    anchor: ARAnchor) {
    guard let planeAnchor = anchor as? ARPlaneAnchor else { return }
    // remove existing plane nodes
    node.enumerateChildNodes { (childNode, _) in
        childNode.removeFromParentNode()
    }
    let planeNode = createARPlaneNode(anchor: planeAnchor)
    node.addChildNode(planeNode)
}

// when detected plane removed, didRemove the plane
func renderer(_ renderer: SCNSceneRenderer, didRemove node: SCNNode, for
    anchor: ARAnchor) {
    guard anchor is ARPlaneAnchor else { return }
    // remove existing plane nodes
    node.enumerateChildNodes { (childNode, _) in
        childNode.removeFromParentNode()
    }
}
```

```
    /** create and return ARPlaneNode */
    func createARPlaneNode(anchor: ARPlaneAnchor) -> SCNNode {
        let pos = SCNVector3Make(anchor.transform.columns.3.x, anchor.transform.
            columns.3.y, anchor.transform.columns.3.z)
//        print("New surface detected at \(pos)")

        // Create the geometry and its materials
        let plane = SCNPlane(width: CGFloat(anchor.extent.x), height: CGFloat
            (anchor.extent.z))
        let grassImage = UIImage(named: "grass")
        let grassMaterial = SCNMaterial()
        grassMaterial.diffuse.contents = grassImage
        grassMaterial.isDoubleSided = true
        plane.materials = [grassMaterial]
        // Create a plane node with the plane geometry
        let planeNode = SCNNode(geometry: plane)
        planeNode.position = pos
        planeNode.transform = SCNMatrix4MakeRotation(-Float.pi / 2, 1, 0, 0)

        // add the wolf to pos of the plane node
        if wolfNode == nil {
            if let wolfScene = SCNScene(named: "art.scnassets/wolf.dae") {
                wolfNode = wolfScene.rootNode.childNode(withName: "wolf",
                    recursively: true)
                wolfNode.position = pos
                sceneView.scene.rootNode.addChildNode(wolfNode!)
            }
        }
        return planeNode
    }
```

“技术本来就应该是尖端的。正如伊拉恩·加内特所说，编程语言的所谓“业界最佳实践”，实际上不会让你变成最佳，只会让你变得很平常。”

–Paul Graham

谢谢