

#WWDC18

Live Screen Broadcast with ReplayKit

Alexander Subbotin, ReplayKit Engineering

ReplayKit

Capture

- Screen visuals
- App audio
- Microphone input

Record and share

Broadcast live



ReplayKit

HD quality capture

Low latency

Low performance impact

Minimal power usage

Privacy safeguards

Agenda

Live broadcast overview

System broadcast picker

Developing broadcast extensions

Protecting content

Live Broadcast Overview

Live Broadcast

Broadcast live to third party broadcast services

Stream audio and visuals directly from device

Provide commentary with microphone and camera (iOS)

Content is secure and only accessible to the broadcast service

Live Broadcast

Usage examples

Stream gameplay to Mobcrush or YouTube

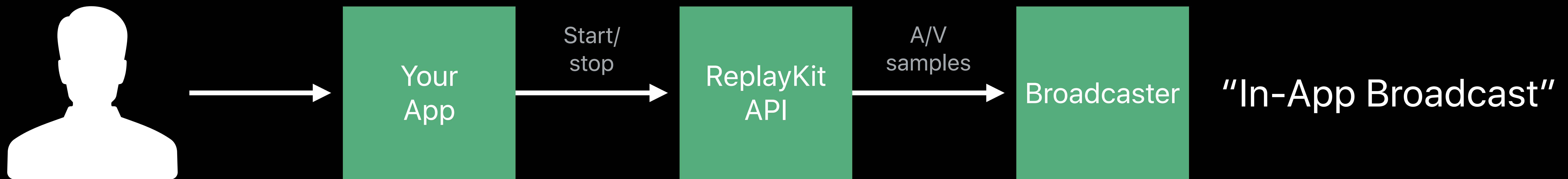
Mirror screen on a WebEx call

Work with customer support via TeamViewerQS

Stream a drawing app to Facebook

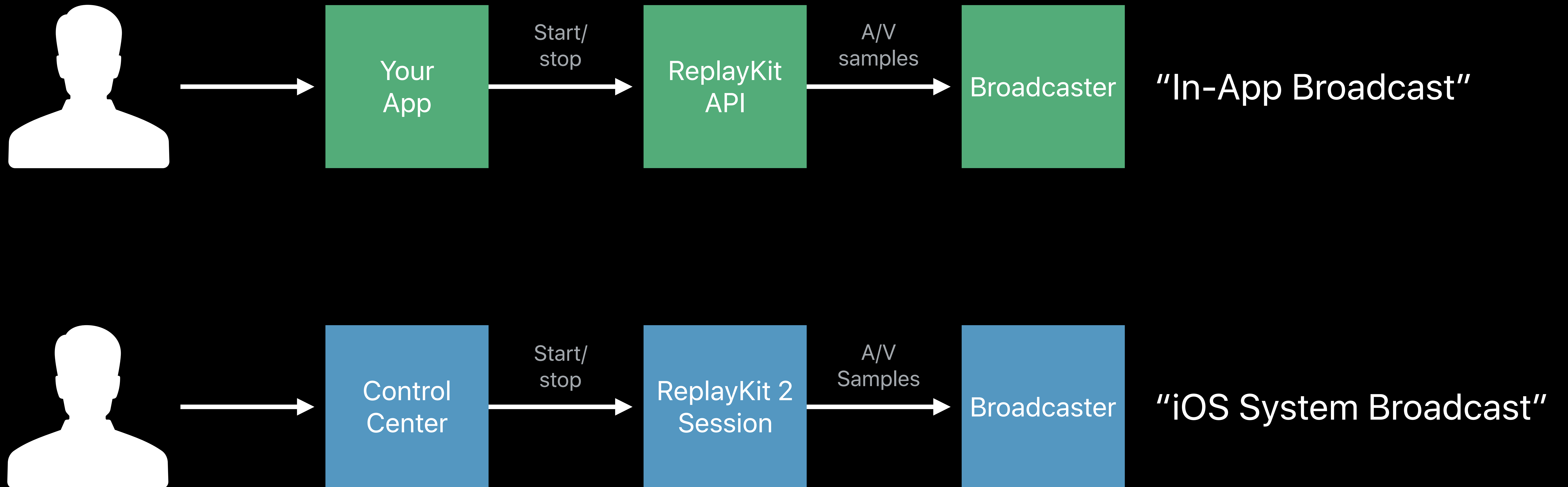
Live Broadcast

ReplayKit versus ReplayKit 2



Live Broadcast

ReplayKit versus ReplayKit 2



In-App Broadcast

Your app or game

- Provides the content—visuals and audio
- Starts and stops the broadcast

Broadcaster app

- Provides sign-in and upload extensions
- Streams content to their network

iOS System Broadcast

Broadcasts all onscreen activity (and sounds)

Start and stop from Control Center

Systemwide, continuous session

- Home screen
- Moving app to app

Built-in to iOS 11 and above



Let me earn your favor,
sweet Queen! Send me off to
slay a rare beast, and bring
you a trophy!



Albort the Swift

Ijeoma

5 years



1069





Everything on your screen, including notifications, will be recorded. Enable Do Not Disturb to prevent unexpected notifications.



Screen Recording

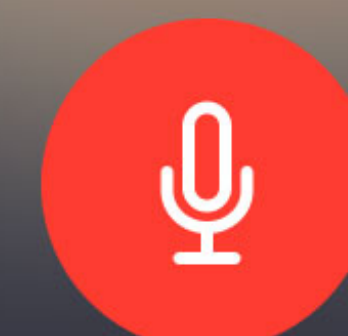
✓  Mobcrush

 Create

 Facebook



Start Broadcast



Microphone
Audio
On



Let me earn your favor,
sweet Queen! Send me off to
slay a rare beast, and bring
you a trophy!



Albort the Swift

Ijeoma

5 years

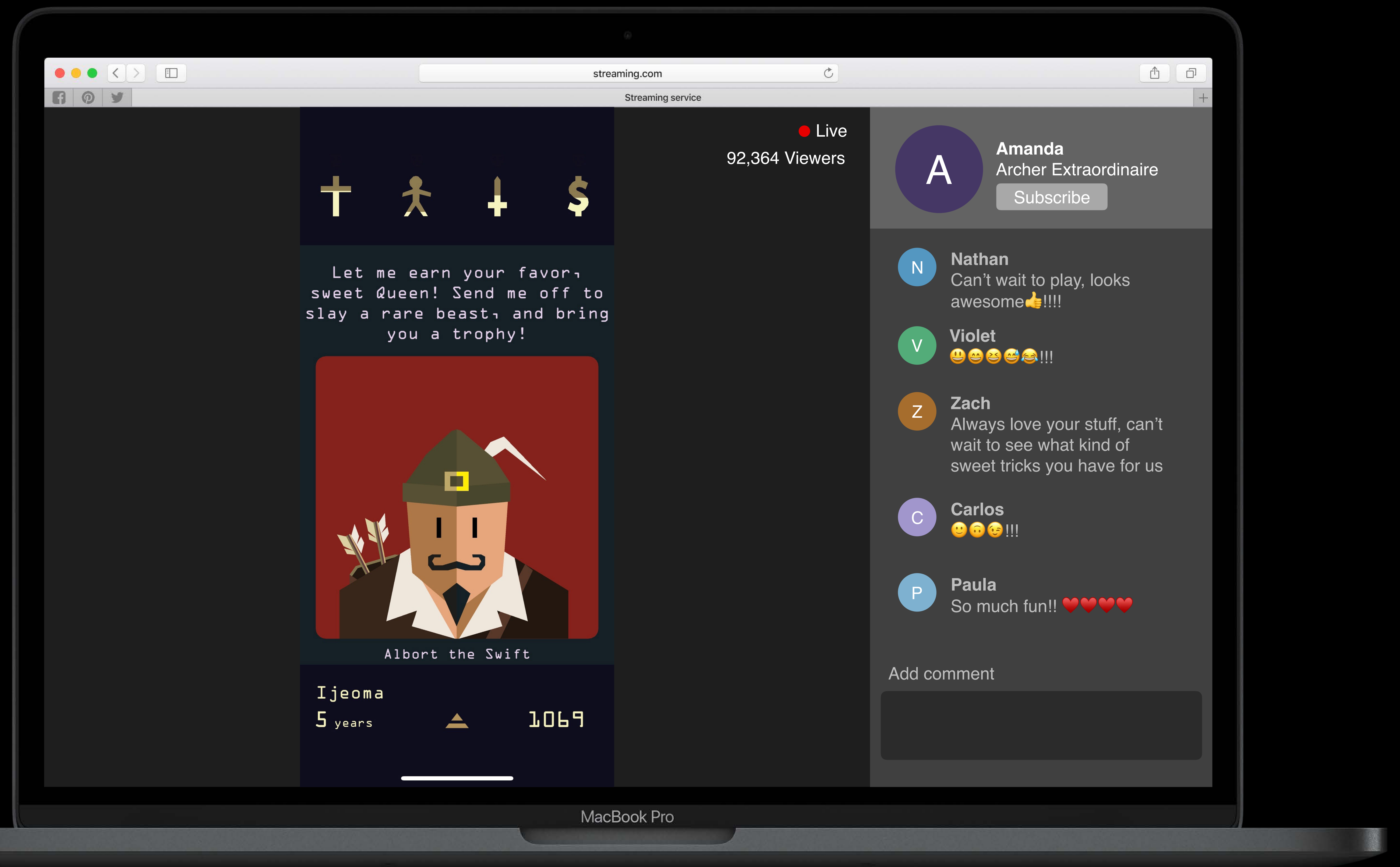
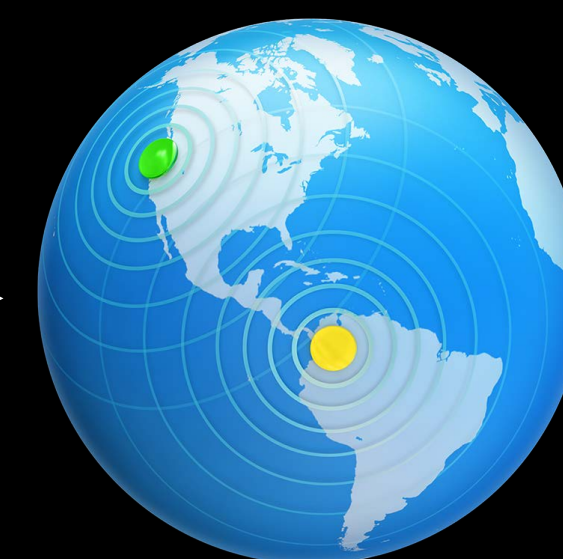


1069

iOS System Broadcast



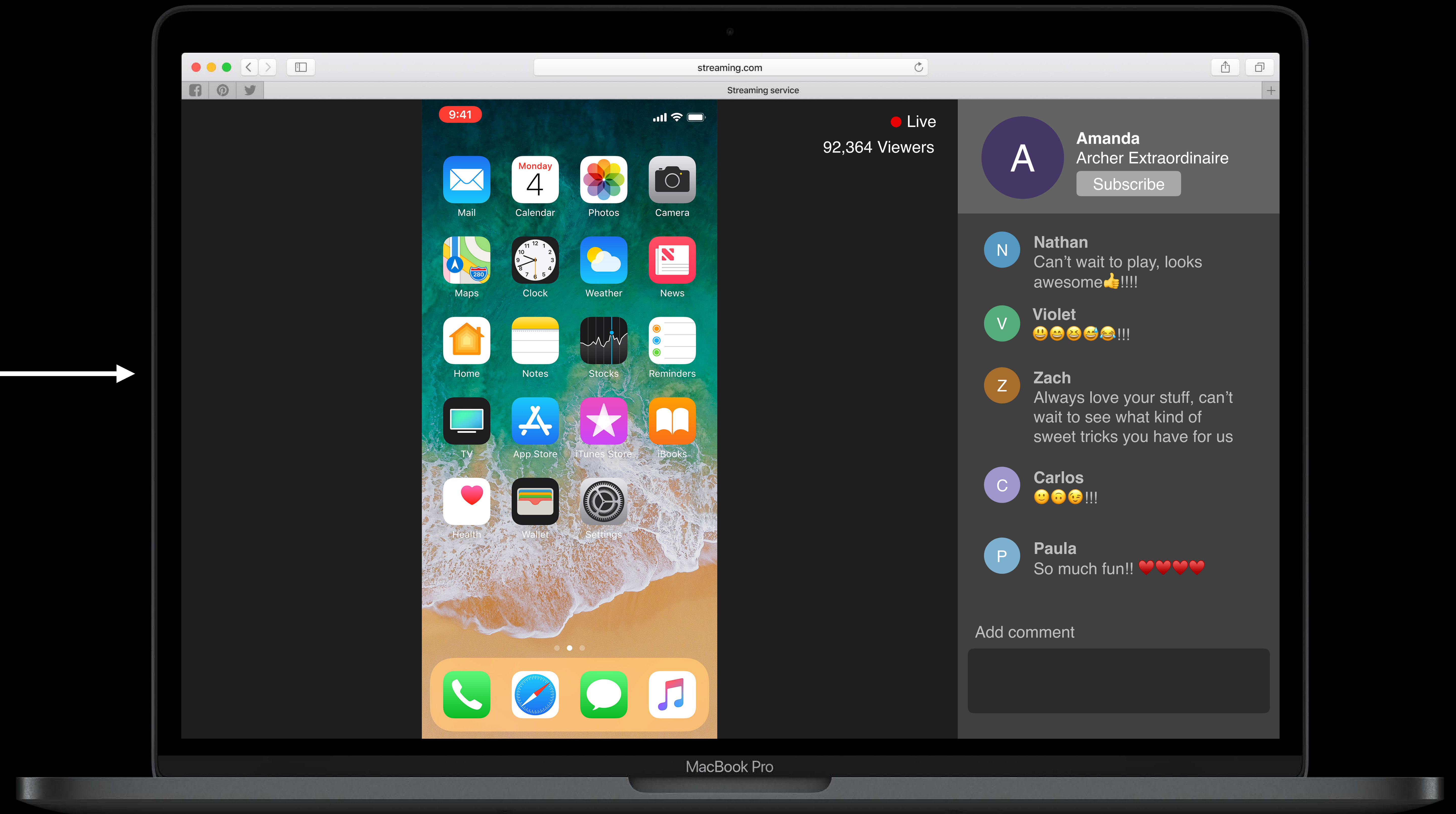
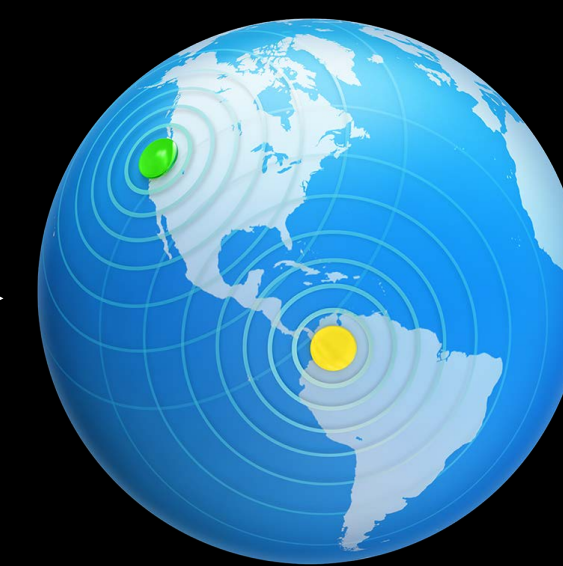
Broadcast application



iOS System Broadcast



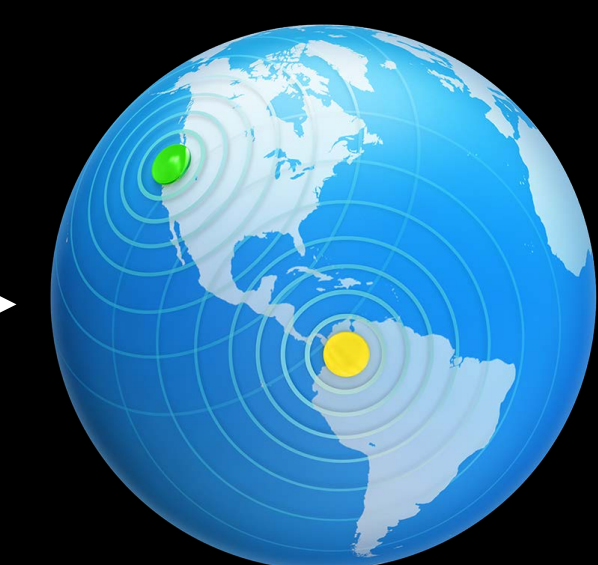
Broadcast application



iOS System Broadcast



Broadcast application



The laptop screen displays a live broadcast of the mobile app interface. The broadcast includes:

- Header:** "streaming.com" and "Streaming service".
- Live Status:** "Live" with a red dot and "92,364 Viewers".
- Comment Section:** A list of user comments:

| User | Comment |
|------------------------------|---|
| Amanda Archer Extraordinaire | Subscribe |
| Nathan | Can't wait to play, looks awesome👍!!!! |
| Violet | 👍👍👍👍👍!!! |
| Zach | Always love your stuff, can't wait to see what kind of sweet tricks you have for us |
| Carlos | 👍👍👍!!! |
| Paula | So much fun!! ❤️❤️❤️❤️ |

The broadcast also shows the same data charts as the mobile app, including the line chart, pie charts, and horizontal bar chart.

iOS System Broadcast



Broadcast application

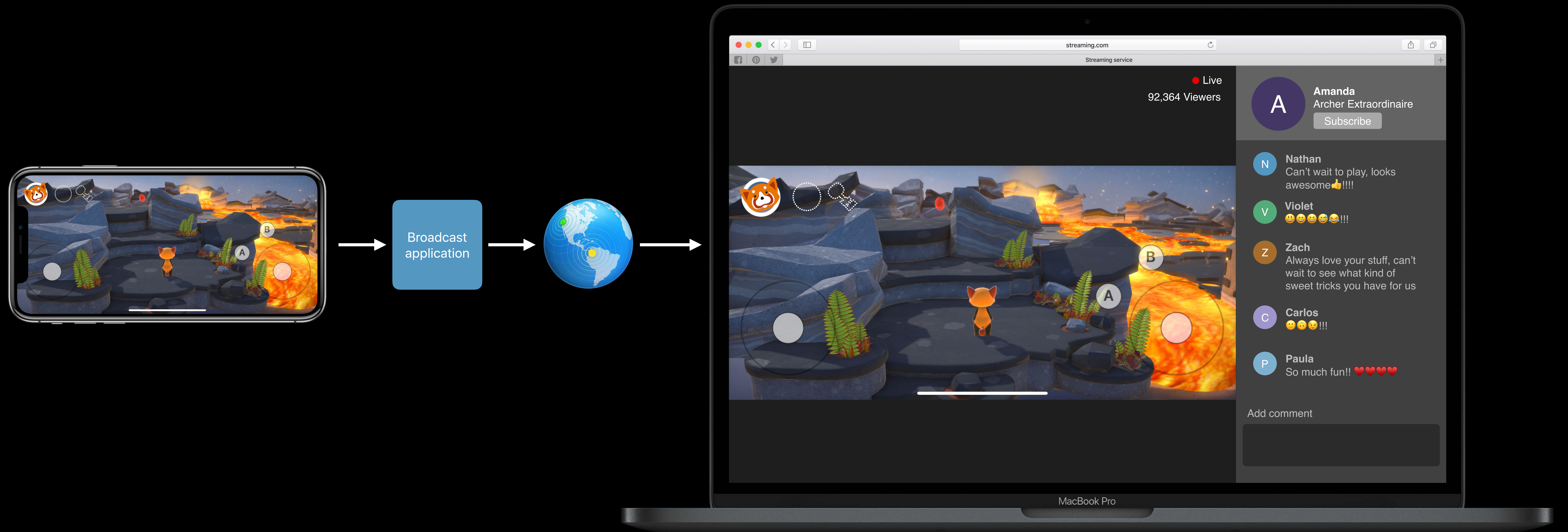


The MacBook Pro screen shows a live streaming interface. At the top right, it says 'Live' with a red dot and '92,364 Viewers'. The main content area displays the same dashboard as the iPhone. On the right side, there is a chat window with several messages:

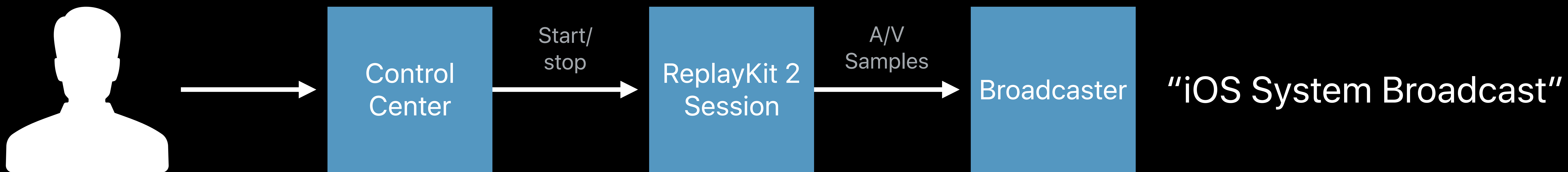
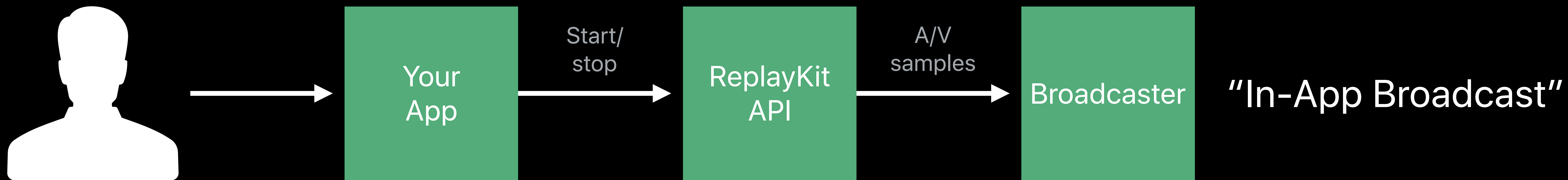
- Amanda Archer Extraordinaire** (Subscribe)
- Nathan**: Can't wait to play, looks awesome👍!!!!
- Violet**: 😄😄😄😄😄!!!!
- Zach**: Always love your stuff, can't wait to see what kind of sweet tricks you have for us
- Carlos**: 😄😄😄!!!!
- Paula**: So much fun!! ❤️❤️❤️❤️

At the bottom of the chat is an 'Add comment' input field. The browser address bar shows 'streaming.com' and 'Streaming service'. The time '9:41' is visible in the top left corner of the browser window.

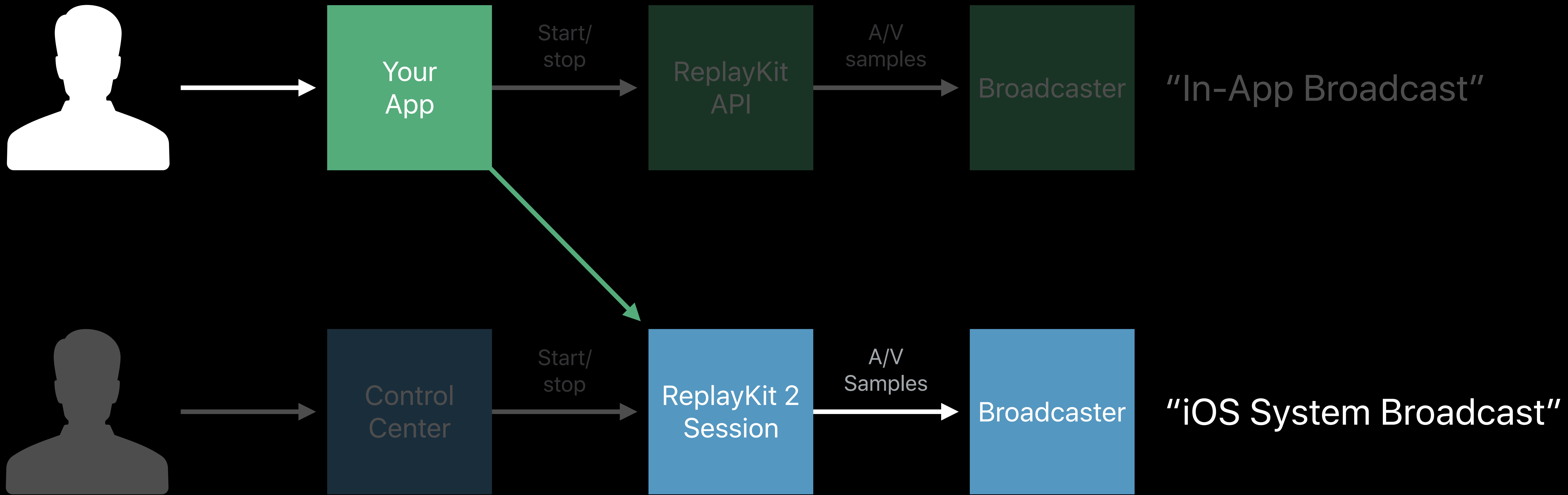
iOS System Broadcast



System Broadcast Picker



NEW



Broadcast Picker



NEW

Your app can initiate an iOS system broadcast session

Simple one-button UI

No compromises for privacy

Secure architecture

New in iOS 12



Fox 2



Fox 2



Everything on your screen, including notifications, will be recorded. Enable Do Not Disturb to prevent unexpected notifications.



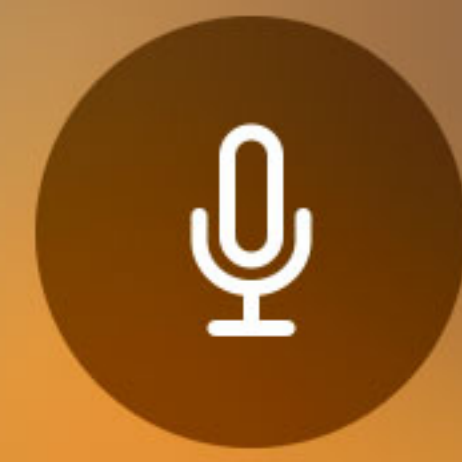
Screen Broadcast

 Facebook

✓  Create



Start Broadcast



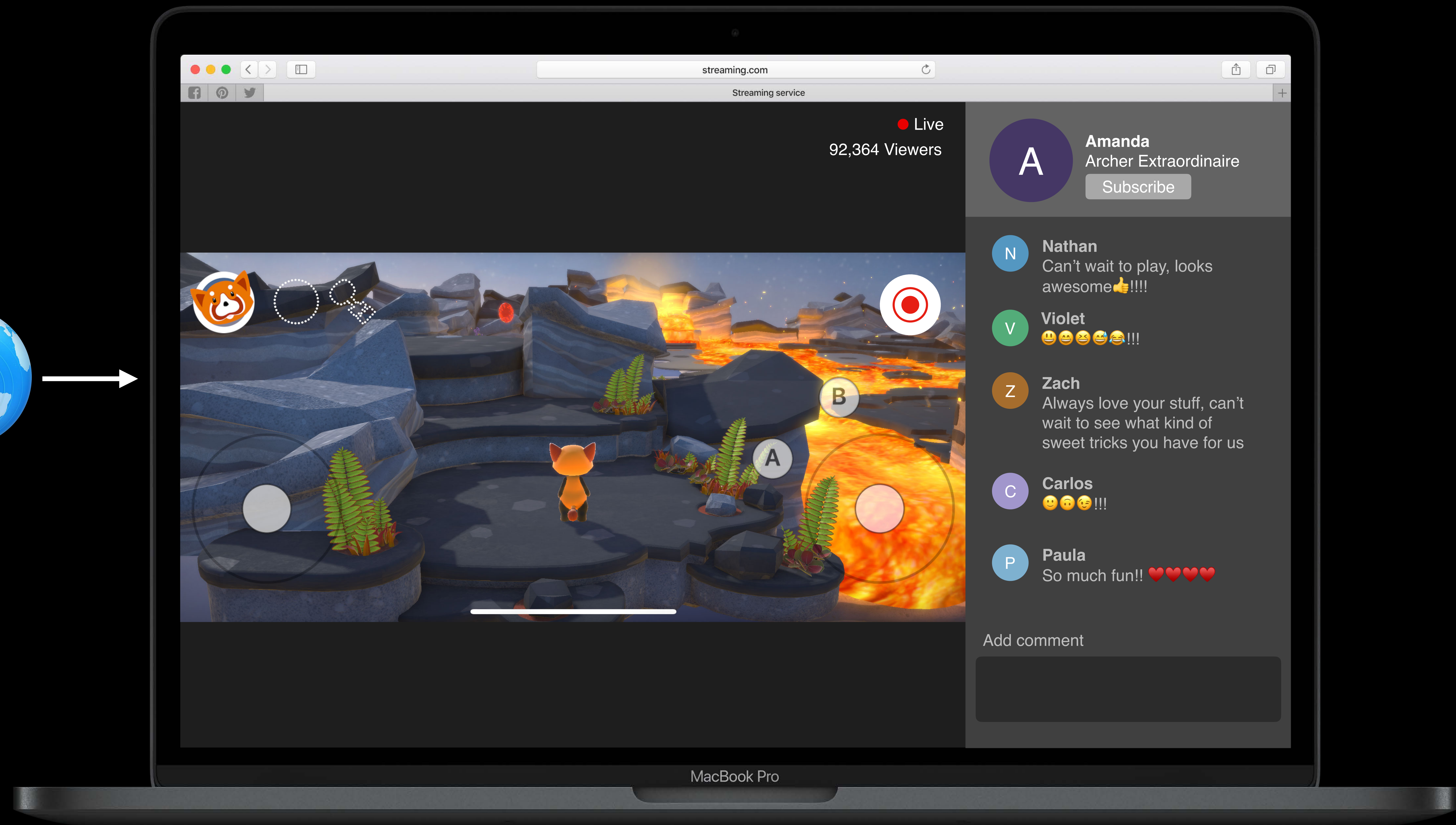
Microphone
Audio
Off



Fox 2



Broadcast application



```
// RPSystemBroadcastPickerView
```


```
class RPSystemBroadcastPickerView: UIView {  
    open var preferredExtension: String  
}
```

fox2 iOS > Generic iOS Device

fox2 > fox2 iOS > Main....board > Main....Base > Game...cene > Game...troller > View > System Broadcast Picker View

fox2

- README.md
- fox2 Shared
- fox2 iOS
 - AAPLButtonOverlay.h
 - AAPLButtonOverlay.m
 - AAPLControlOverlay.h
 - AAPLControlOverlay.m
 - AAPLGameViewController.h
 - AAPLGameViewController.m
 - AAPLPadOverlay.h
 - AAPLPadOverlay.m
 - AppDelegate.h
 - AppDelegate.m
 - Main.storyboard
 - LaunchScreen.storyboard
 - Info.plist
 - Supporting Files
 - fox2 tvOS
 - fox2 macOS
 - Products
 - Frameworks



View as: iPhone X (wC hC) 100%

Device Orientation

Vary for Traits

Custom Class

- Class: RPSystemBroadcastPickerView
- Module: None
- Inherit Module From Target

Identity

Restoration ID

User Defined Runtime Attributes

| Key Path | Type | Value |
|----------|------|-------|
|----------|------|-------|

Document

- Label: Xcode Specific Label
- Object ID: uCY-pf-gtD
- Lock: Inherited - (Nothing)
- Notes: No Font
- Comment For Localizer

Accessibility

- Accessibility: Enabled
- Label: Label
- Hint: Hint
- Identifier: Identifier

Image View - Displays a single image, or an animation described by an array of images.

Collection View - Displays data in a collection of cells.

Collection View Cell - Defines the attributes and behavior of cells in a collection view.

fox2 iOS > Generic iOS Device

fox2 > fox2 iOS > Main....board > Main....Base > Game...cene > Game...troller > View > System Broadcast Picker View

fox2

- README.md
- fox2 Shared
- fox2 iOS
 - AAPLButtonOverlay.h
 - AAPLButtonOverlay.m
 - AAPLControlOverlay.h
 - AAPLControlOverlay.m
 - AAPLGameViewController.h
 - AAPLGameViewController.m
 - AAPLPadOverlay.h
 - AAPLPadOverlay.m
 - AppDelegate.h
 - AppDelegate.m
 - Main.storyboard
 - LaunchScreen.storyboard
 - Info.plist
 - Supporting Files
- fox2 tvOS
- fox2 macOS
- Products
- Frameworks

View as: iPhone X (wC hC) 100%

Device Orientation

Vary for Traits

Filter

Custom Class

Class: RPSystemBroadcastPickerView

Module: None

Inherit Module From Target

Identity

Restoration ID

User Defined Runtime Attributes

| Key Path | Type | Value |
|----------|------|-------|
|----------|------|-------|

Document

Label: Xcode Specific Label

Object ID: uCY-pf-gtD

Lock: Inherited - (Nothing)

Notes: No Font

Comment For Localizer

Accessibility

Accessibility: Enabled

Label: Label

Hint: Hint

Identifier: Identifier

Image View - Displays a single image, or an animation described by an array of images.

Collection View - Displays data in a collection of cells.

Collection View Cell - Defines the attributes and behavior of cells in a collection view.

```
import ReplayKit.broadcast

class ViewController: UIViewController {
    var broadcastPicker: RPSystemBroadcastPickerView?

    override func viewDidLoad() {
        super.viewDidLoad()
        broadcastPicker = RPSystemBroadcastPickerView(frame: kPickerFrame)
        view.addSubview(broadcastPicker)
    }
}
```

Everything on your screen, including notifications, will be recorded. Enable Do Not Disturb to prevent unexpected notifications.



Screen Broadcast



TeamViewer

Start Broadcast



Microphone
Audio
Off

preferredExtension

Pairs broadcast picker to a particular extension

Assign bundle identifier of your extension

Initialize before view is presented

```
import ReplayKit.broadcast

class ViewController: UIViewController {
    var broadcastPicker: RPSystemBroadcastPickerView?

    override func viewDidLoad() {
        super.viewDidLoad()
        broadcastPicker = RPSystemBroadcastPickerView(frame: kPickerFrame)
        broadcastPicker.preferredExtension = "com.your-app.broadcast.extension"

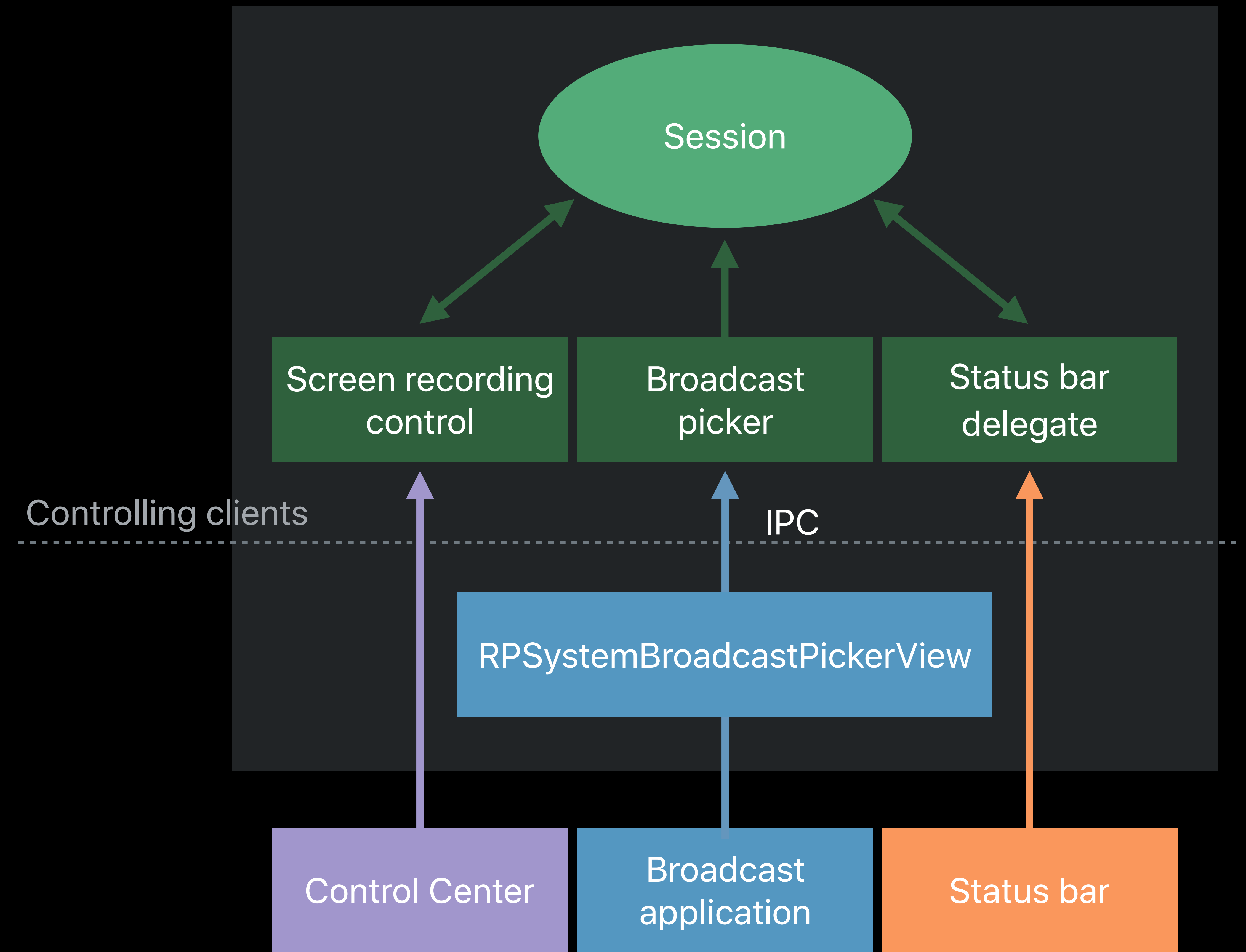
        view.addSubview(broadcastPicker)
    }
}
```

RPSystemBroadcastPickerView

Brings up system broadcast picker

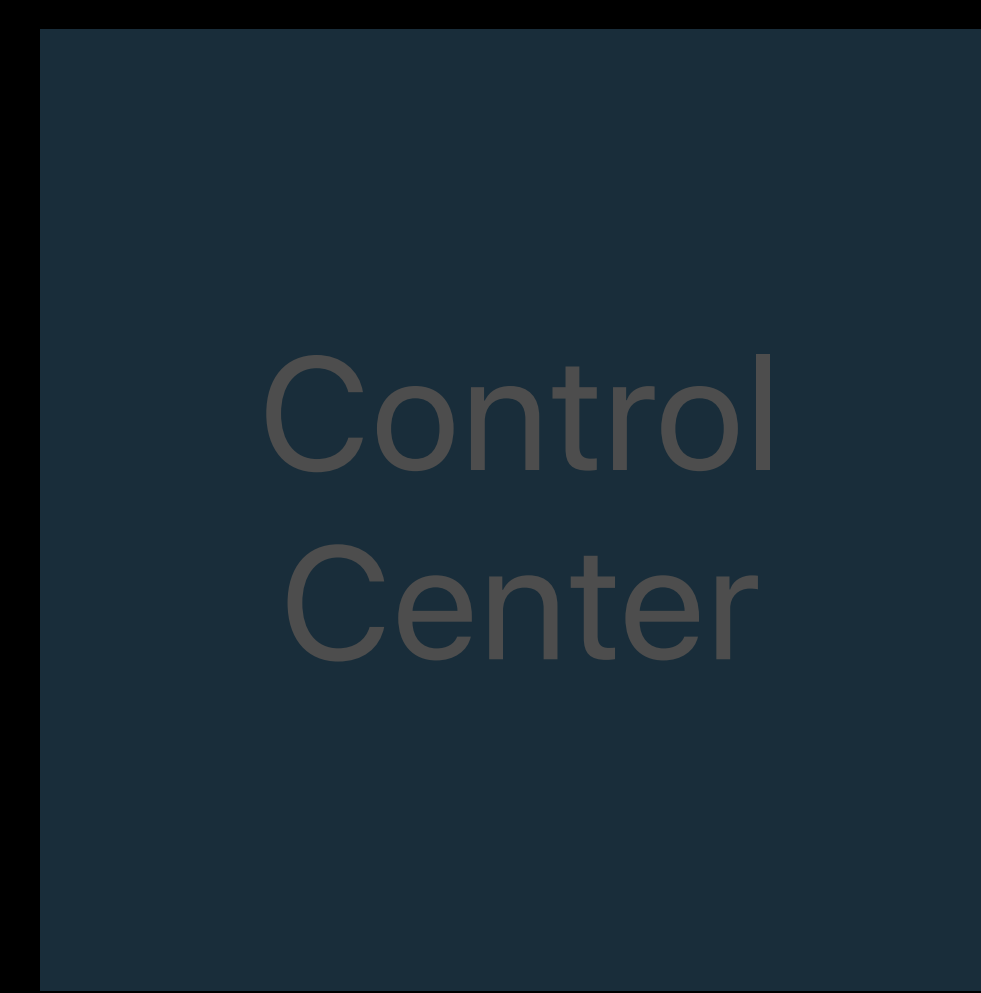
Allows to configure broadcast picker

Doesn't own any state of the session



Developing Broadcast Extensions

For ReplayKit 2



Start/
stop



A/V
samples



"iOS System Broadcast"

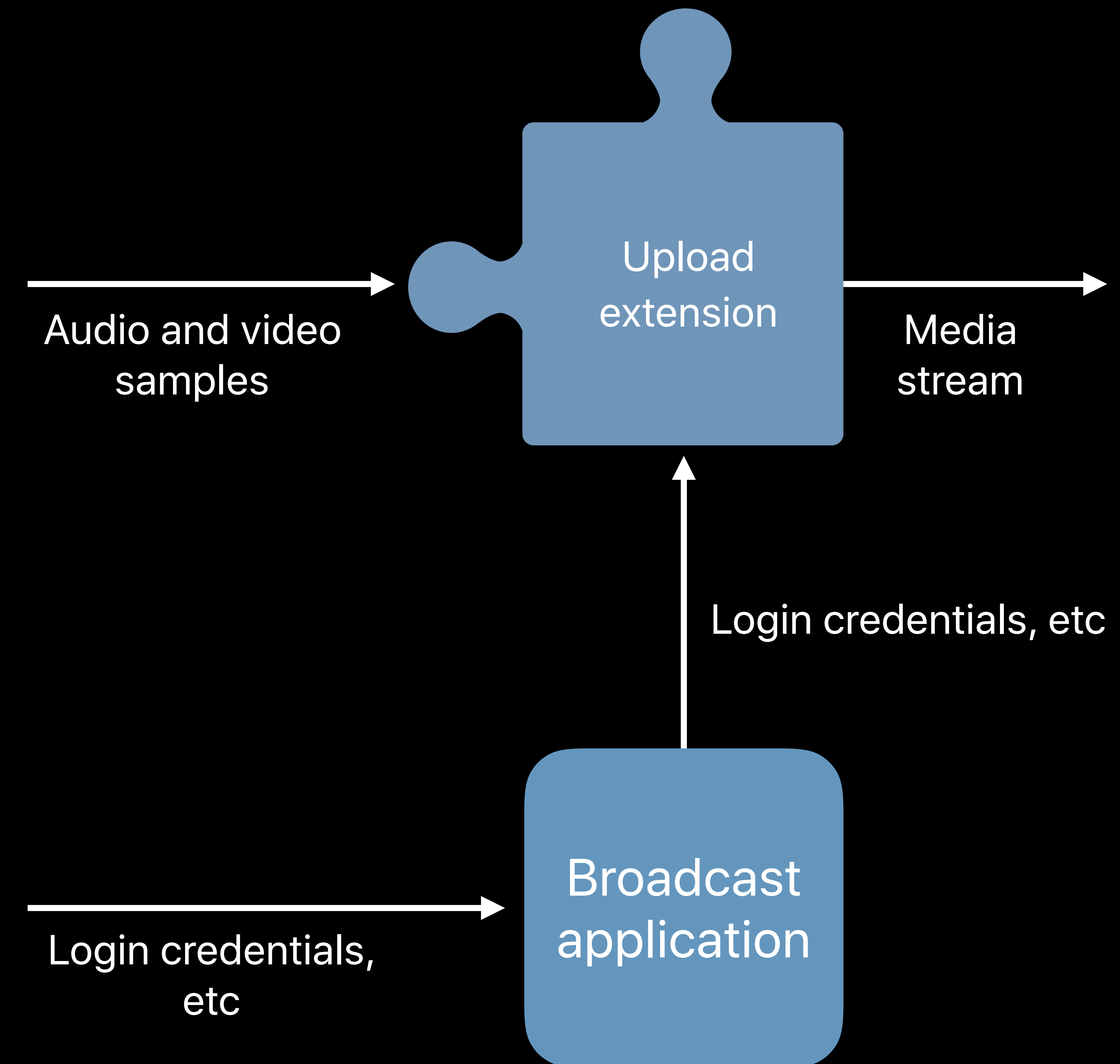
Broadcast App and Extension

Broadcast Application

- Account sign-in, broadcast title

Broadcast Upload Extension

- Encode samples, upload to service



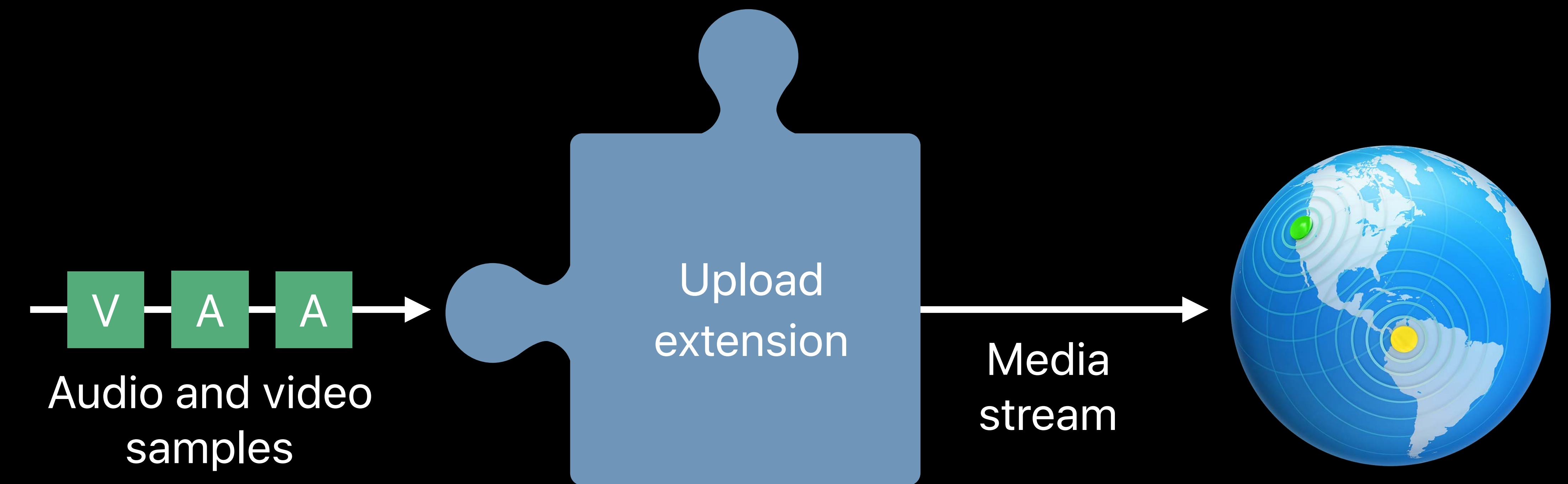
Broadcast Upload Extension

Receives audio and video samples

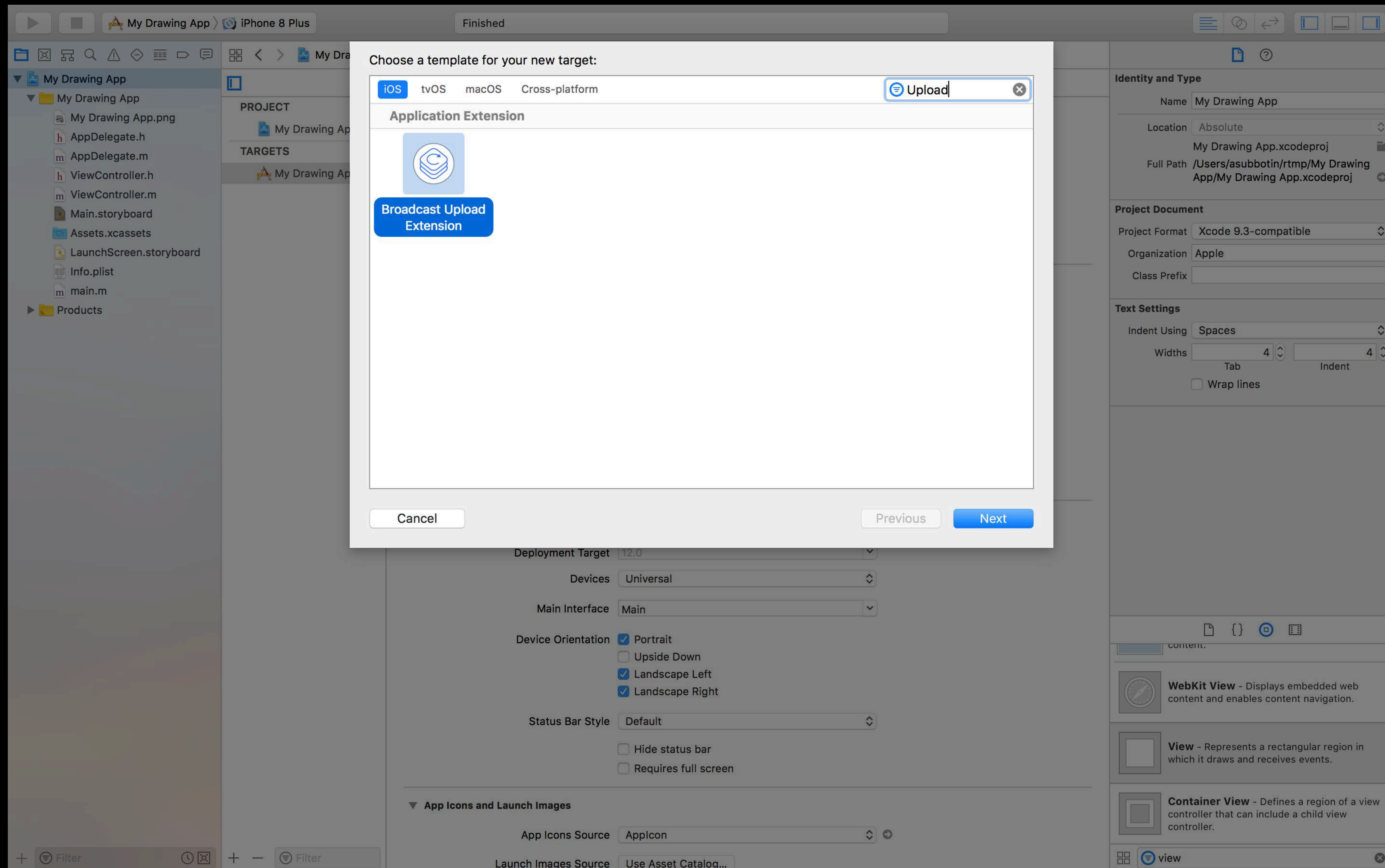
Encodes and uploads video stream

Handles device orientation changes

Annotates broadcast with app information

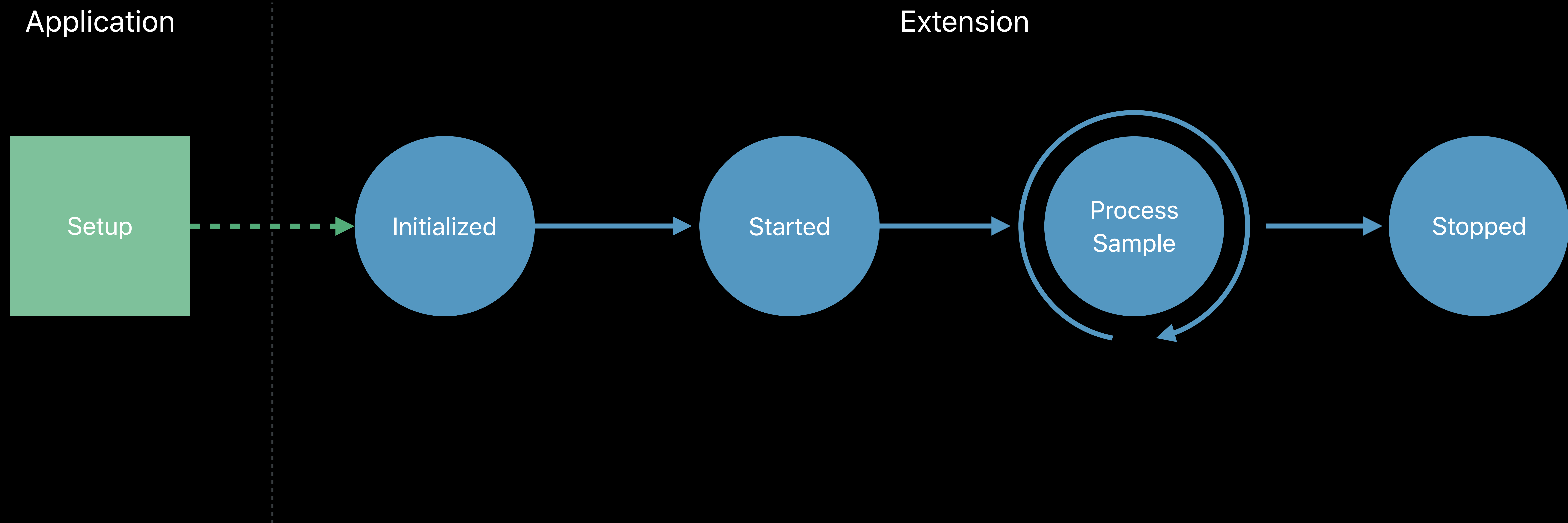


Broadcast Extension Template



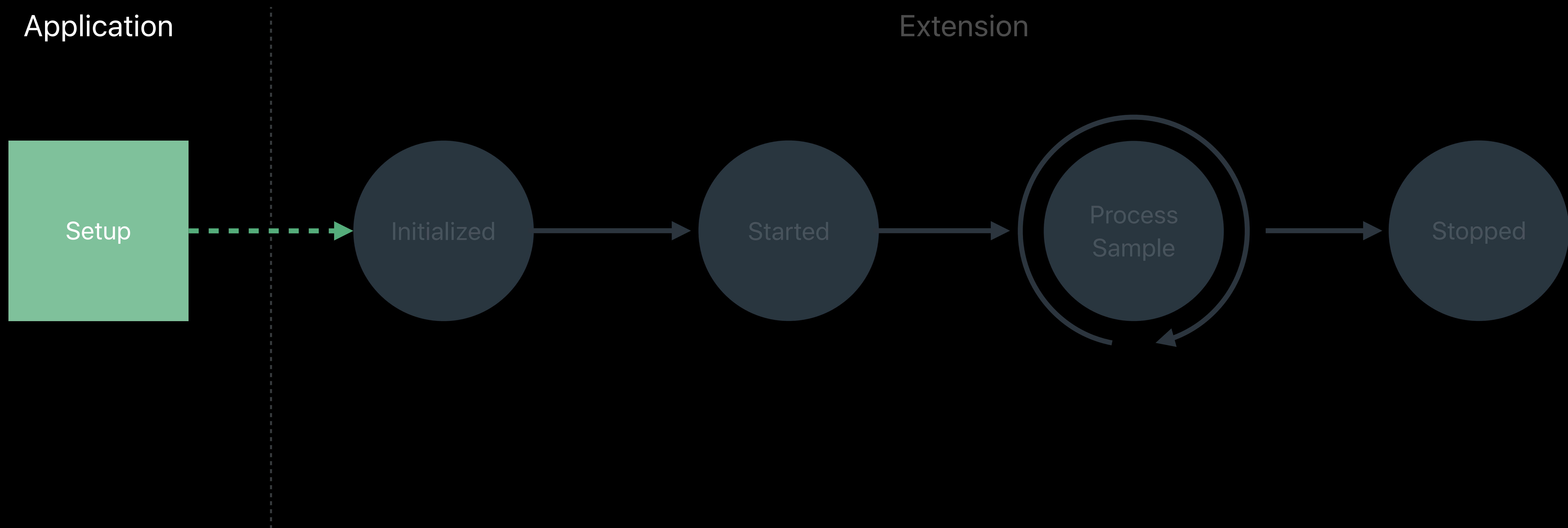
```
// SampleHandler created by Xcode templates for Upload Extension
class SampleHandler: RPBroadcastSampleHandler {
    // User has requested to start the broadcast
    override func broadcastStarted(withSetupInfo setupInfo: [String : NSObject]?)
    // User has requested to finish the broadcast
    override func broadcastFinished()
    // Handle the sample buffer here
    override func processSampleBuffer(_ sampleBuffer: CMSampleBuffer,
                                      with sampleBufferType: RPSampleBufferType)
    // Use details of application to annotate the broadcast
    override func broadcastAnnotated(withApplicationInfo info: [String : NSObject])
}
```

Broadcast Lifecycle

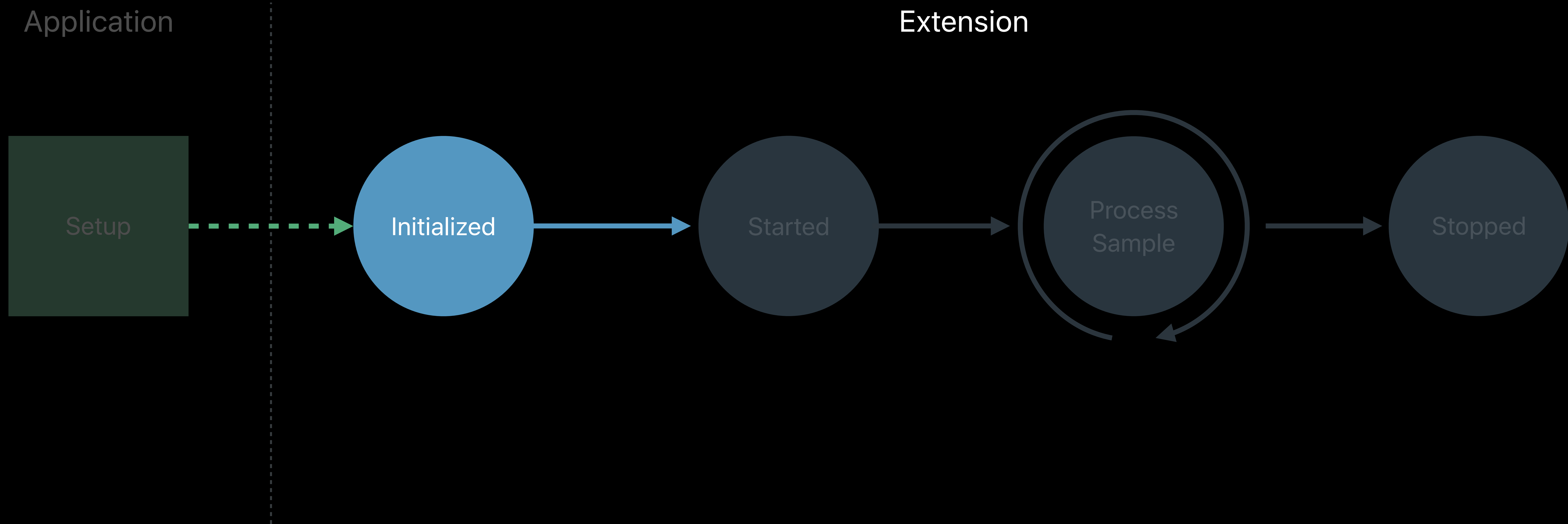


Handling Sign-In and Broadcast Setup

Broadcast application

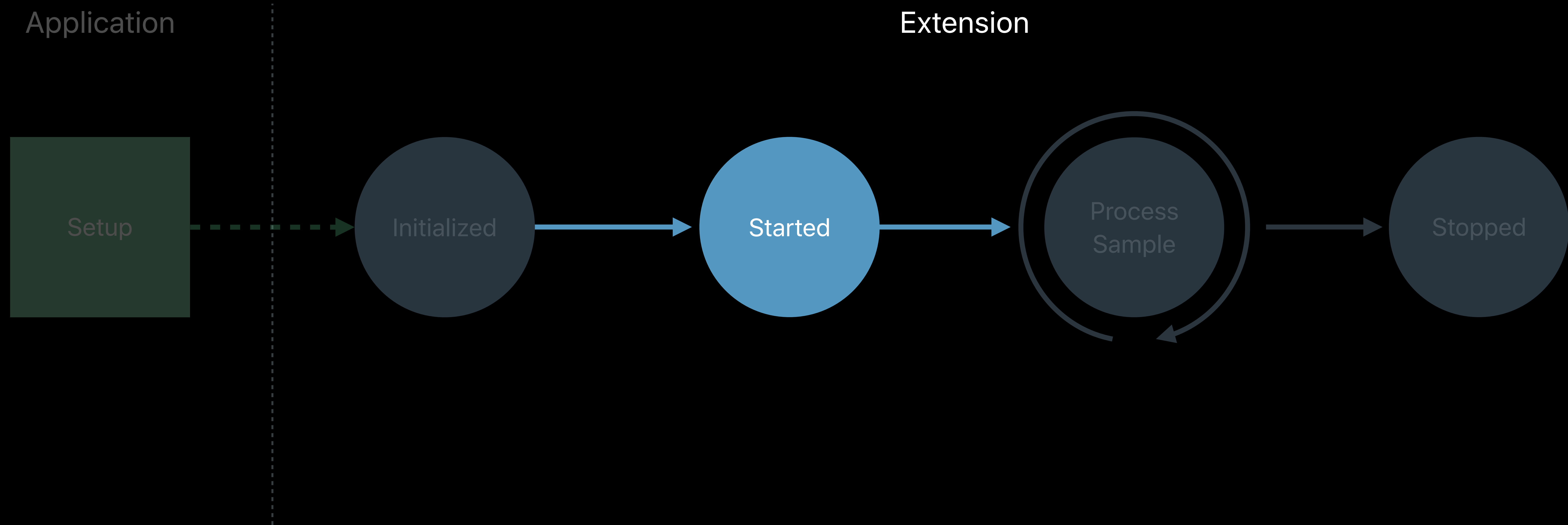


Initialization



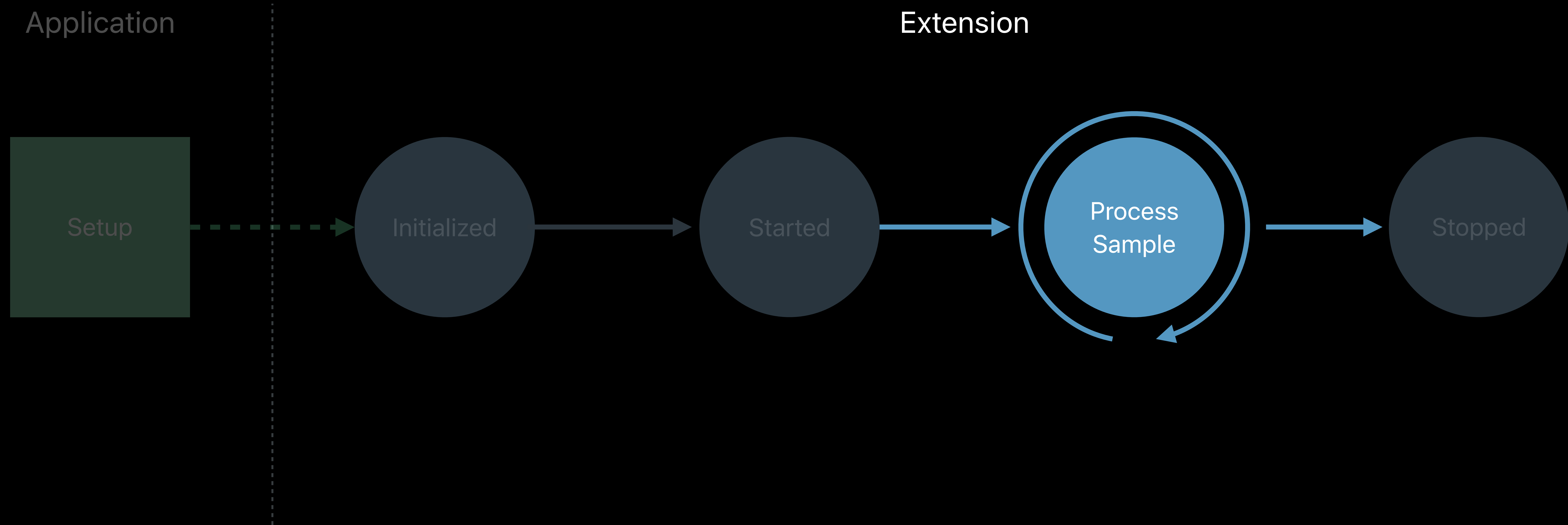
```
// Override init to read login credentials from shared keychain
class SampleHandler : RPBroadcastSampleHandler {
    override fun init() {
        super.init()
        session = BroadcastSession.instance
        var credentials = KeychainAccess.getLoginCredentials()
        session.authenticate(credentials)
    }
}
```


Handling broadcastStarted



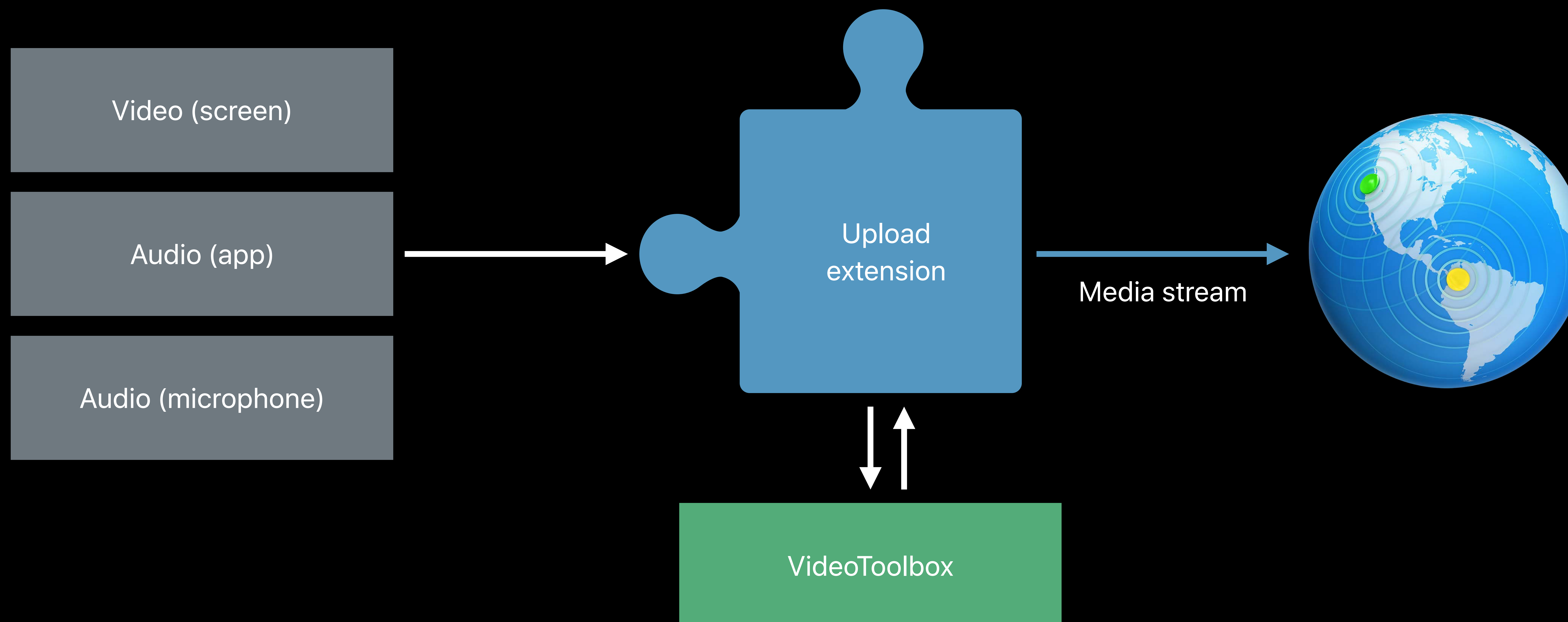
```
// Override broadcastStarted to prepare to receive media samples
override func broadcastStarted(withSetupInfo setupInfo: [String : NSObject]?) {
    // Verify user is logged in
    if (session.userLoggedIn()) {
        session.createMediaEngine()
    }
}
```

Processing Media Samples



Broadcast Upload Extension

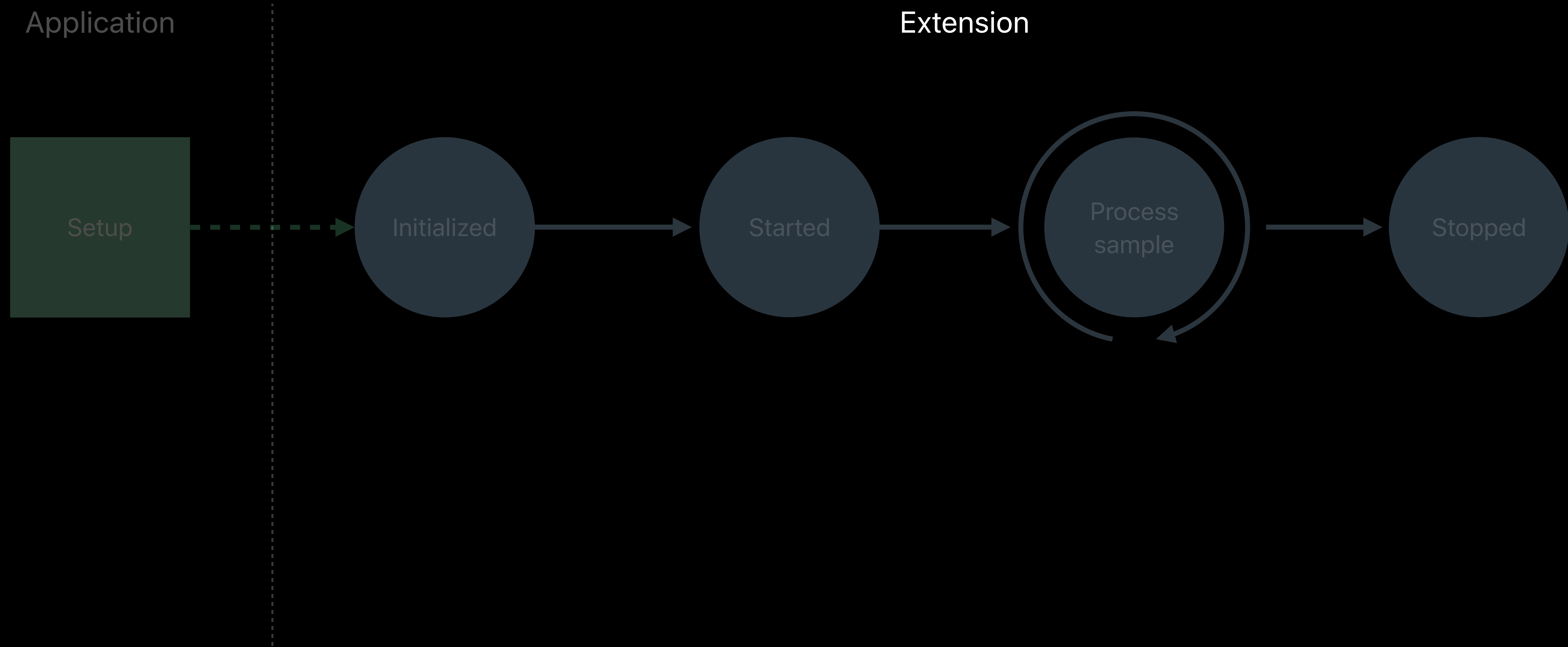
processSampleBuffer



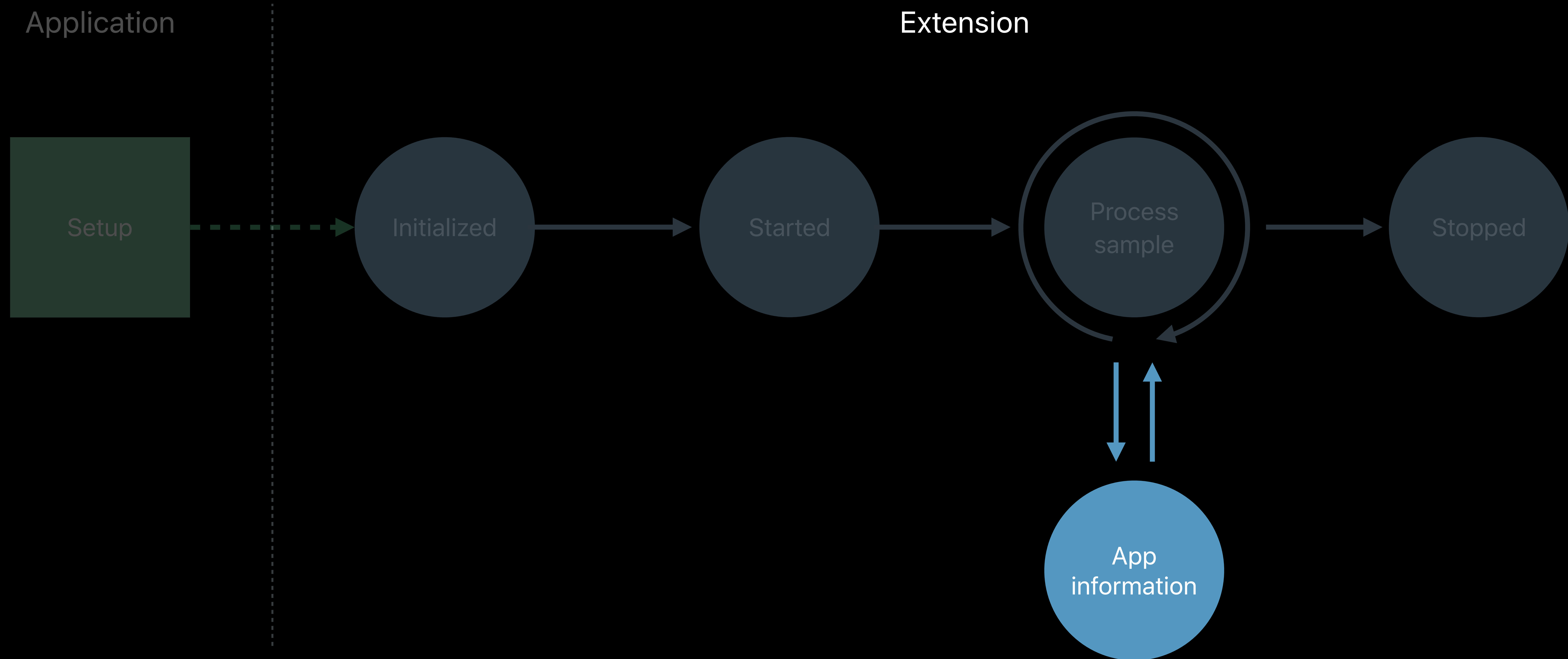
```
// Both audio and video samples are handled by processSampleBuffer routine
override func processSampleBuffer(_ sampleBuffer: CMSampleBuffer,
                                  with sampleBufferType: RPSampleBufferType) {
    switch sampleBufferType {
    case RPSampleBufferType.video:
        var imageBuffer:CVImageBuffer = CMSampleBufferGetImageBuffer(sampleBuffer)!
        var pts = CMSampleBufferGetPresentationTimeStamp(sampleBuffer) as CMTime
        VTCompressionSessionEncodeFrame(session, imageBuffer, pts,
                                         kCMTimeInvalid, nil, nil, nil)

        break
    case RPSampleBufferType.audioApp:
        // Handle audio sample buffer for app audio
        break
    case RPSampleBufferType.audioMic:
        // Handle audio sample buffer for mic audio
        break
    }
}
```

Handling Application Information



Handling Application Information





Search

Q Angry

Users

Games



Angry Birds 2



Angry Flappies



Angry Birds Go!



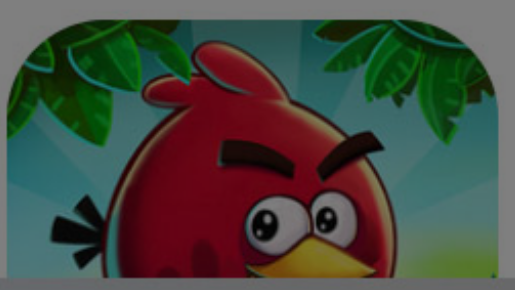
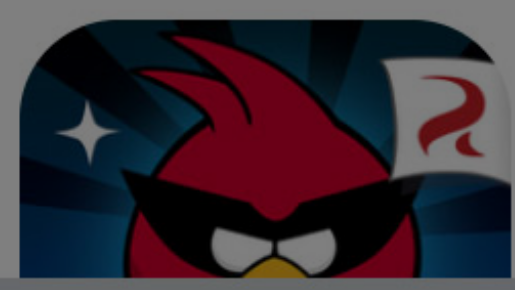
Angry Birds Epic RPG



Angry Birds Free



Angry Birds Star Wars II

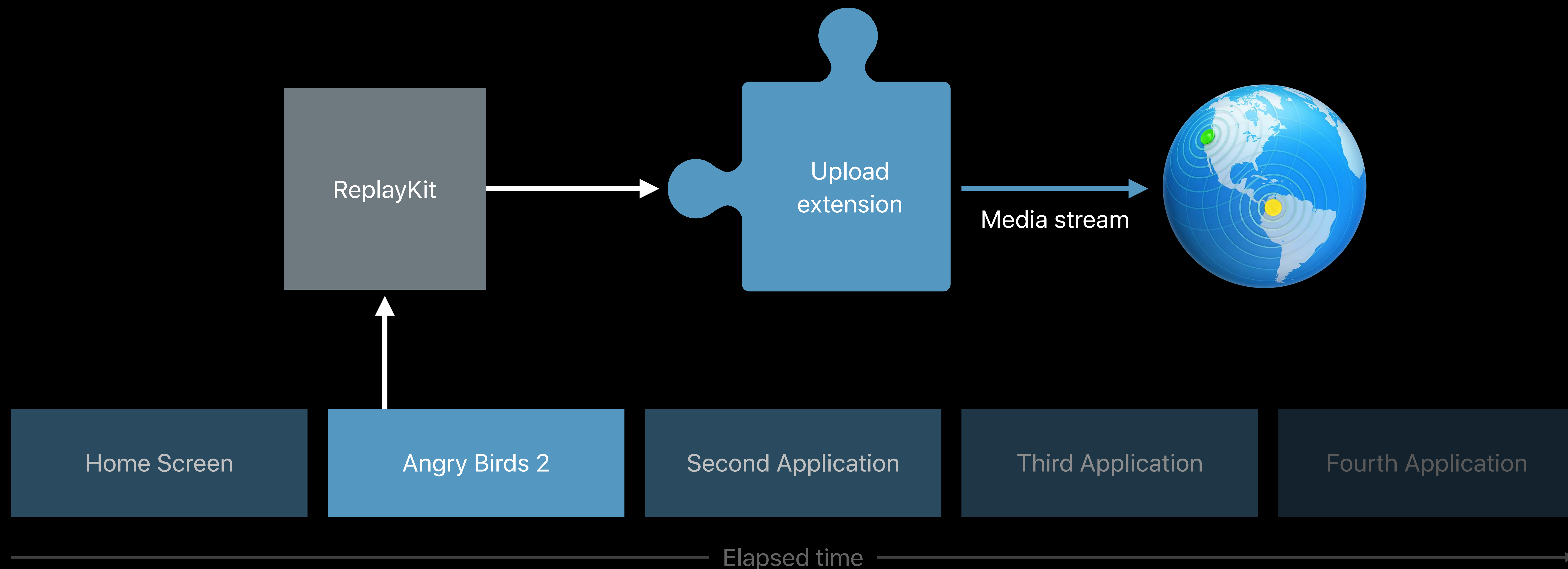


q w e r t y u i o p
a s d f g h j k l
↑ z x c v b n m ↵
123 space Search



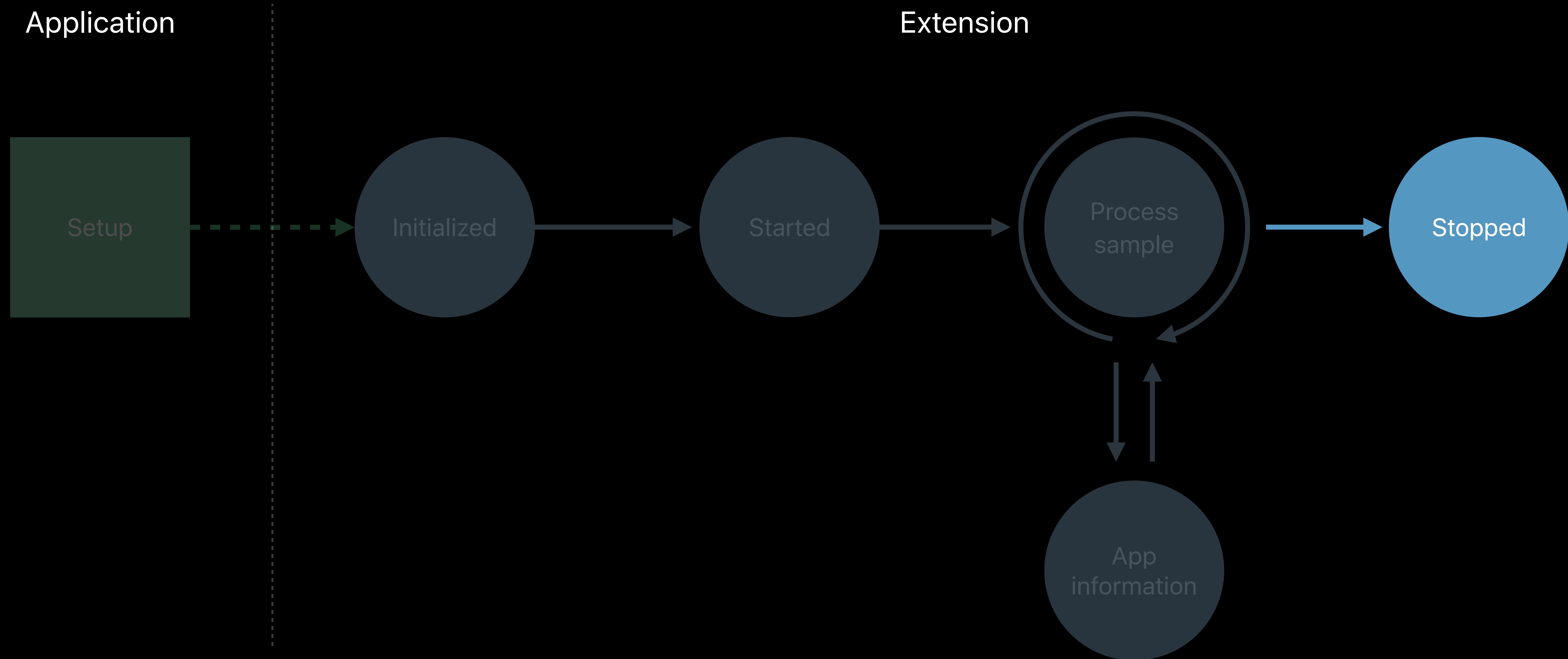
Broadcast Upload Extension

`broadcastAnnotatedWithApplicationInfo`

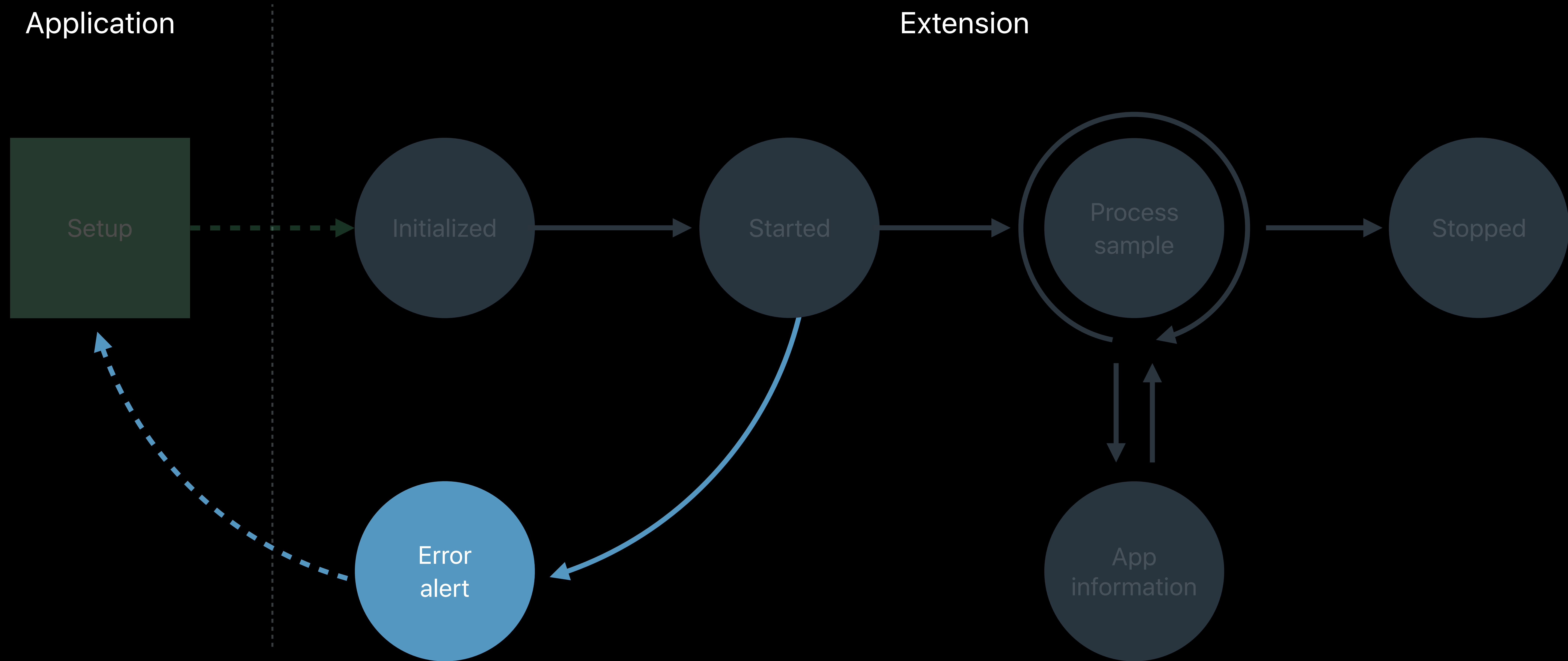


```
// Use application details to help users find your broadcast
override fun broadcastAnnotated(withApplicationInfo applicationInfo: [AnyHashable : Any]) {
    var bundleIdentifier = applicationInfo[RPApplicationInfoBundleIdentifierKey]
    if (bundleIdentifier != nil) {
        session.addMetadataWithApplicationInfo(bundleIdentifier)
    }
}
```

Handling broadcastFinished



Handling Sign-In





Screen Broadcasting

Live Broadcast to Mobcrush has stopped due to: Not Logged In

OK

[Go to Application](#)

Mobcrush



PLAY

Livestream your gameplay everywhere
at once.



Sign up

Sign In

By signing up, you agree to our
[Terms & Privacy Policy](#)

```
// Override broadcastStarted to prepare to receive media samples
override func broadcastStarted(withSetupInfo setupInfo: [String : NSObject]?) {
    // Verify user is logged in and there's network connectivity
    if (session.userLoggedIn()) {
        session.createMediaEngine()
    } else {
        let userInfo = [NSLocalizedFailureReasonErrorKey : "Not Logged In"]
        let error = NSError(domain: "RPBroadcastErrorDomain", code: 401, userInfo: userInfo)
        finishBroadcastWithError(error)
    }
}
```

Developing Broadcast Extensions

Summary

Implement sign-in and setup UI in your application

Finish the broadcast and send user to the app if something is missing

Encode and upload video stream to your service

Use information about app on the screen to help users find your broadcast

Protecting Content

Protecting Content

UIScreen.isCaptured

Prevent capturing of audio and video content of your app

Stop media playback or displaying sensitive content

- Check value of `UIScreen.captured`
- Register for `UINavigationControllerDidChangeNotification`
- Check `UIScreen.screens.count` to allow screen mirroring

```
// Protecting content of your application from being captured
```

```
import UIKit
```

```
class func handleScreenCapturedChange() {
```

```
    let isScreenMirroring = UIScreen.screens.count > 1
```

```
    if (UIScreen.isCaptured && !isScreenMirroring) {
```

```
        // stop audio playback and remove sensitive content from the screen
```

```
    }
```

```
}
```

Summary

Live screen broadcast

System broadcast picker

Developing broadcast extensions

Protecting content

More Information

<https://developer.apple.com/wwdc18/601>

 **WWDC18**