System Frameworks Session

#WWDC16

What's New in HomeKit

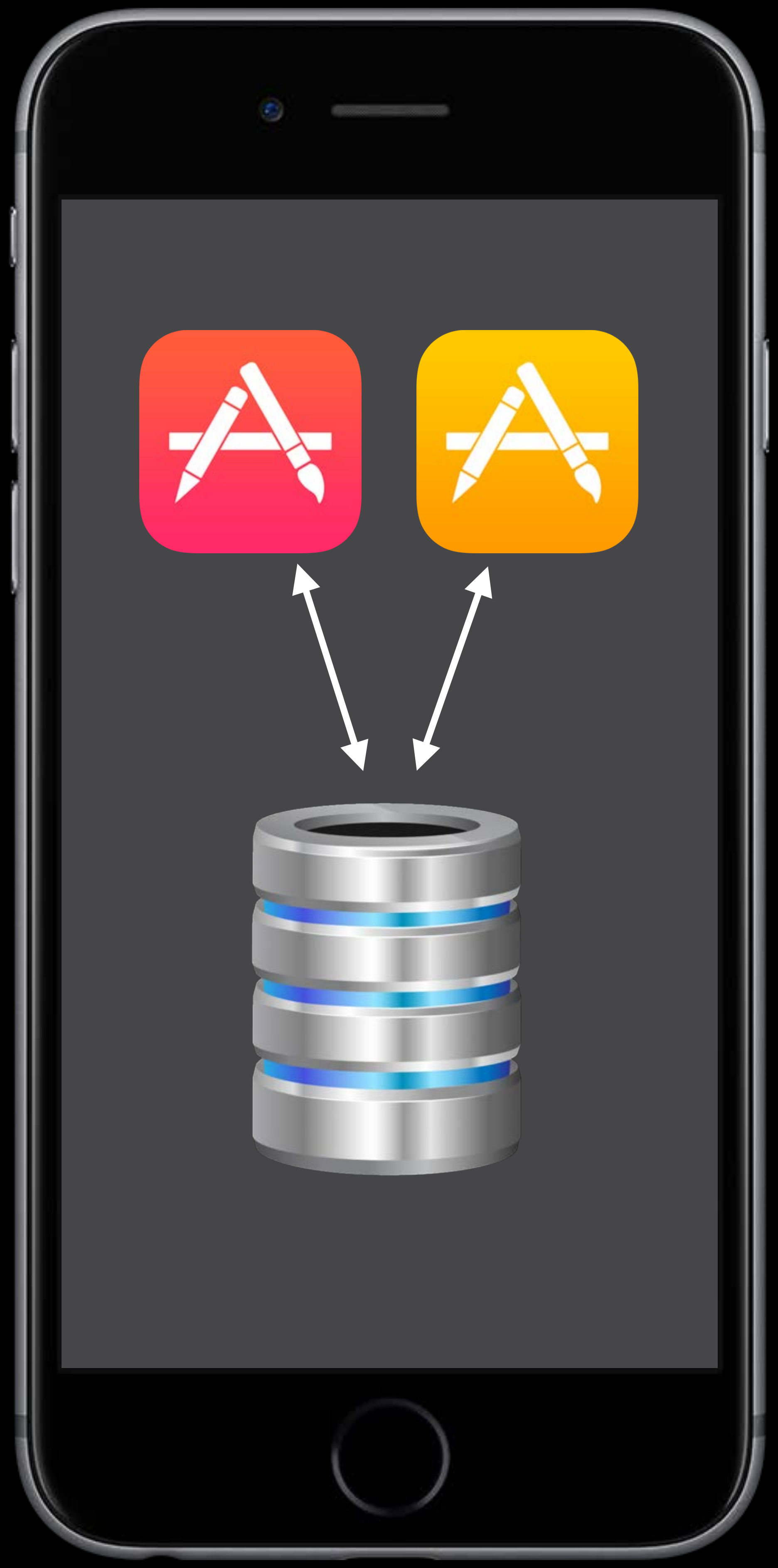
Session 710

Dennis Mathews HomeKit Engineering

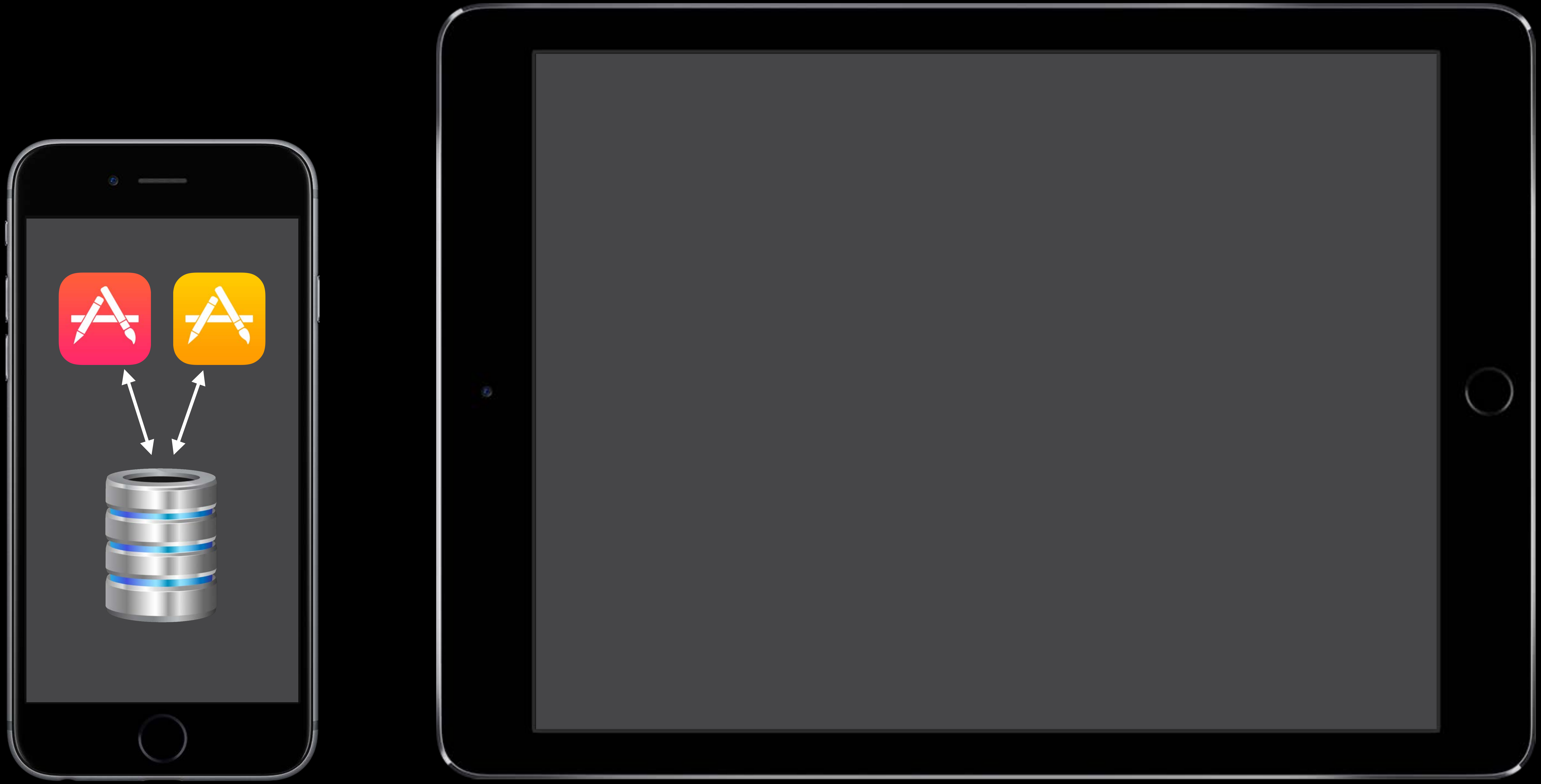
Overview

Home Database

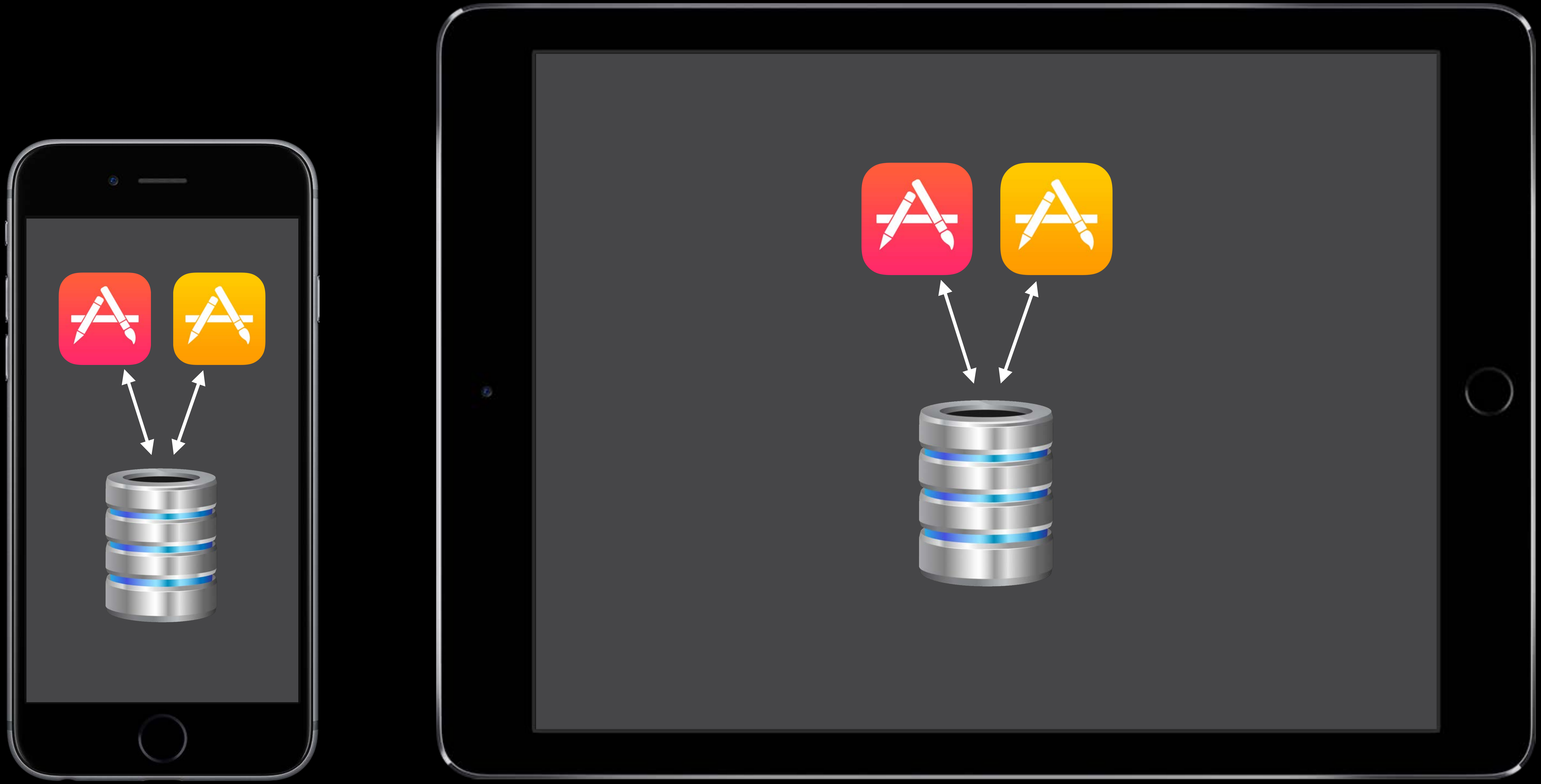
Home Database



Home Database



Home Database



Home Sharing

Home Sharing

Sharing

Home Sharing

Sharing

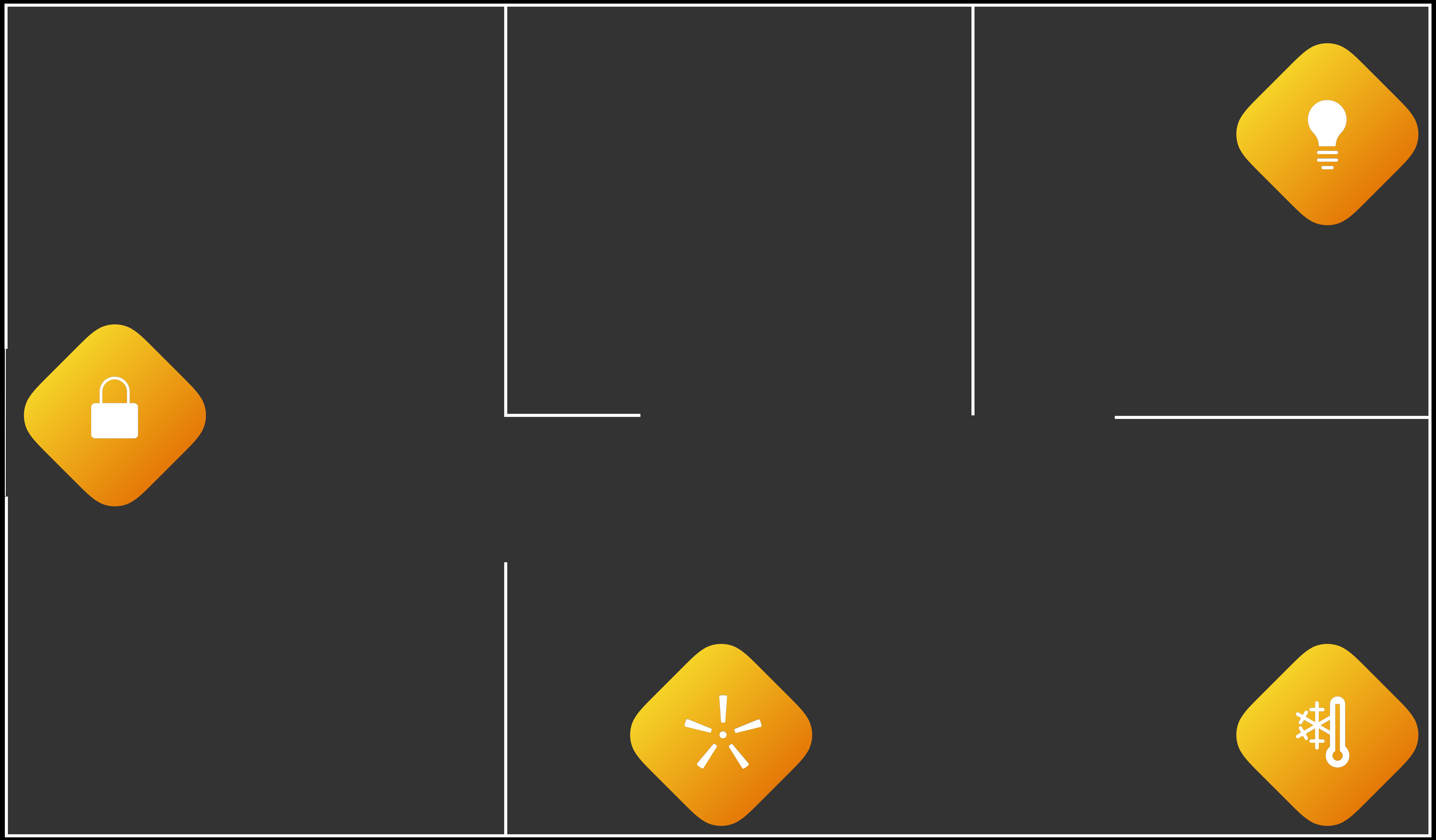
- Owner invites users

Home Sharing

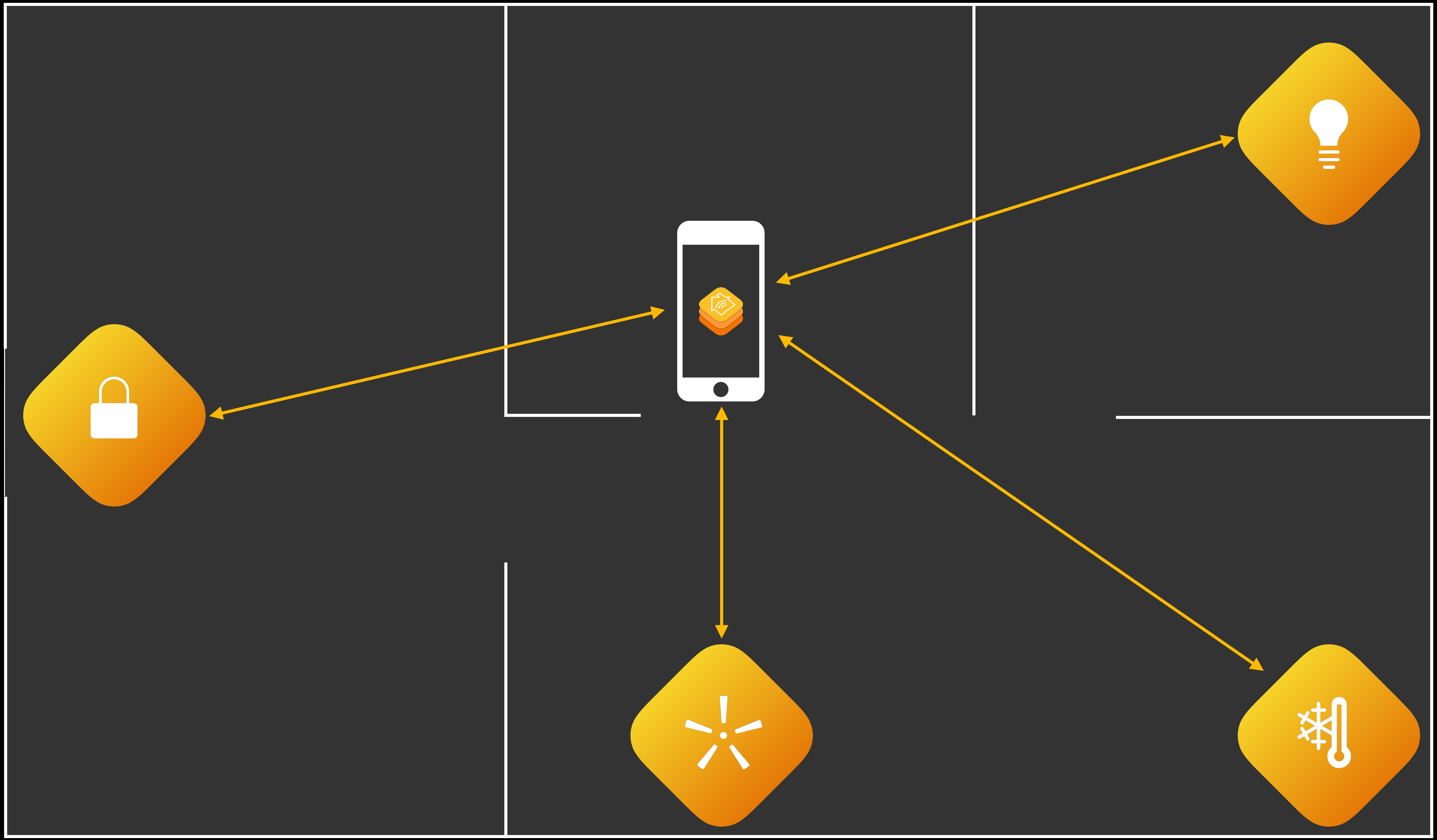
Sharing

- Owner invites users
- User accepts invitation

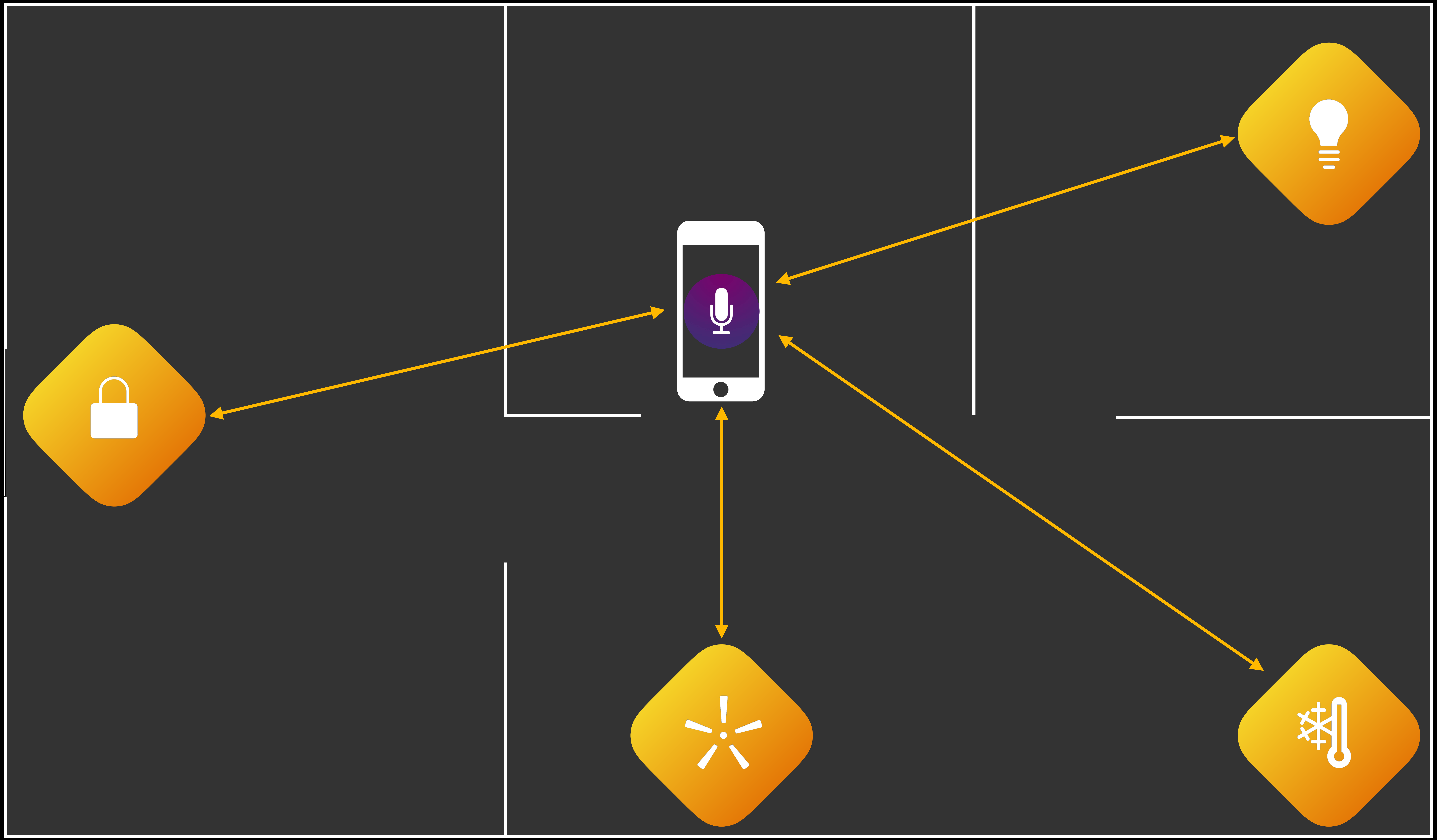
Common Protocol



Common Protocol

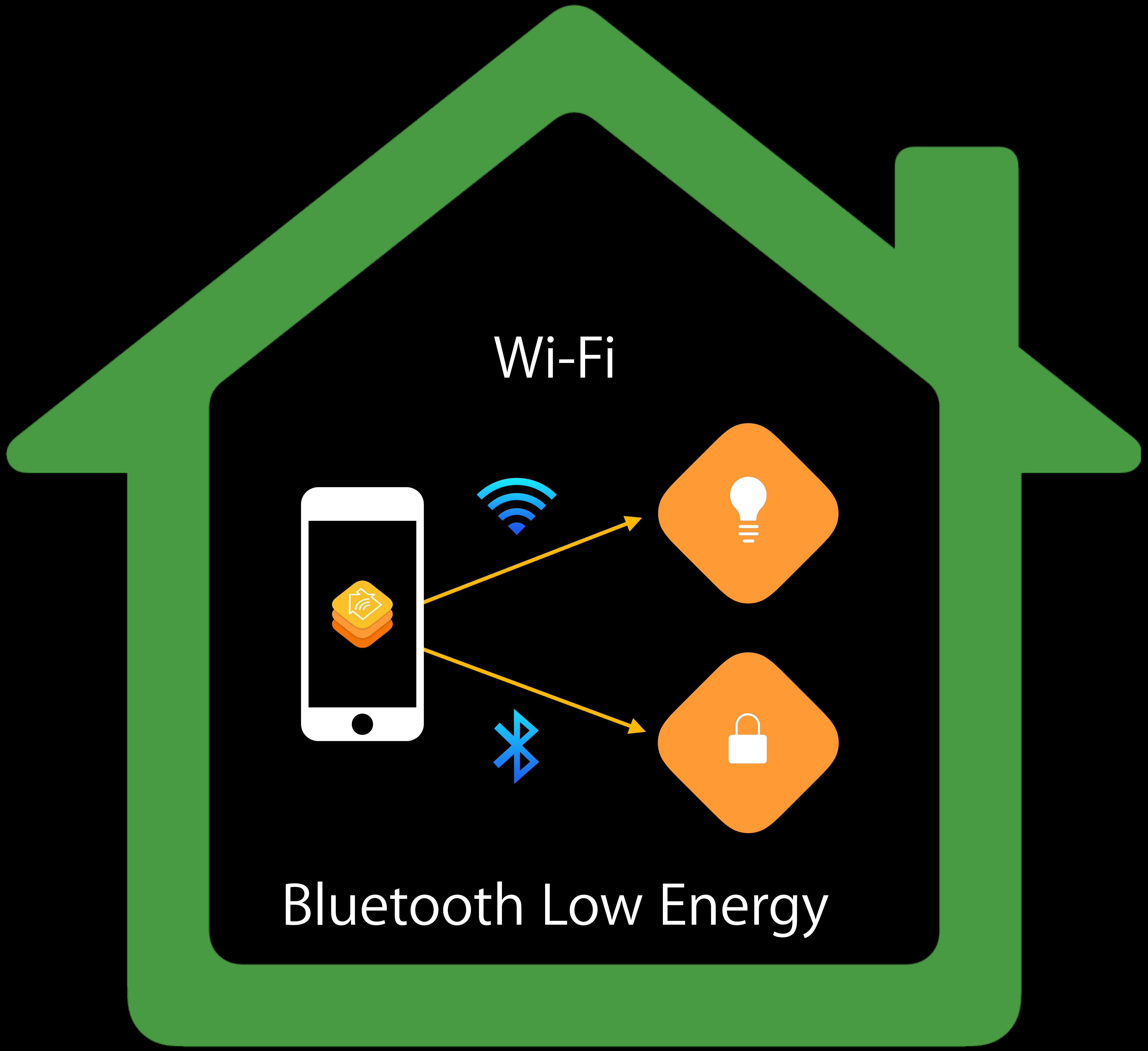


Common Protocol



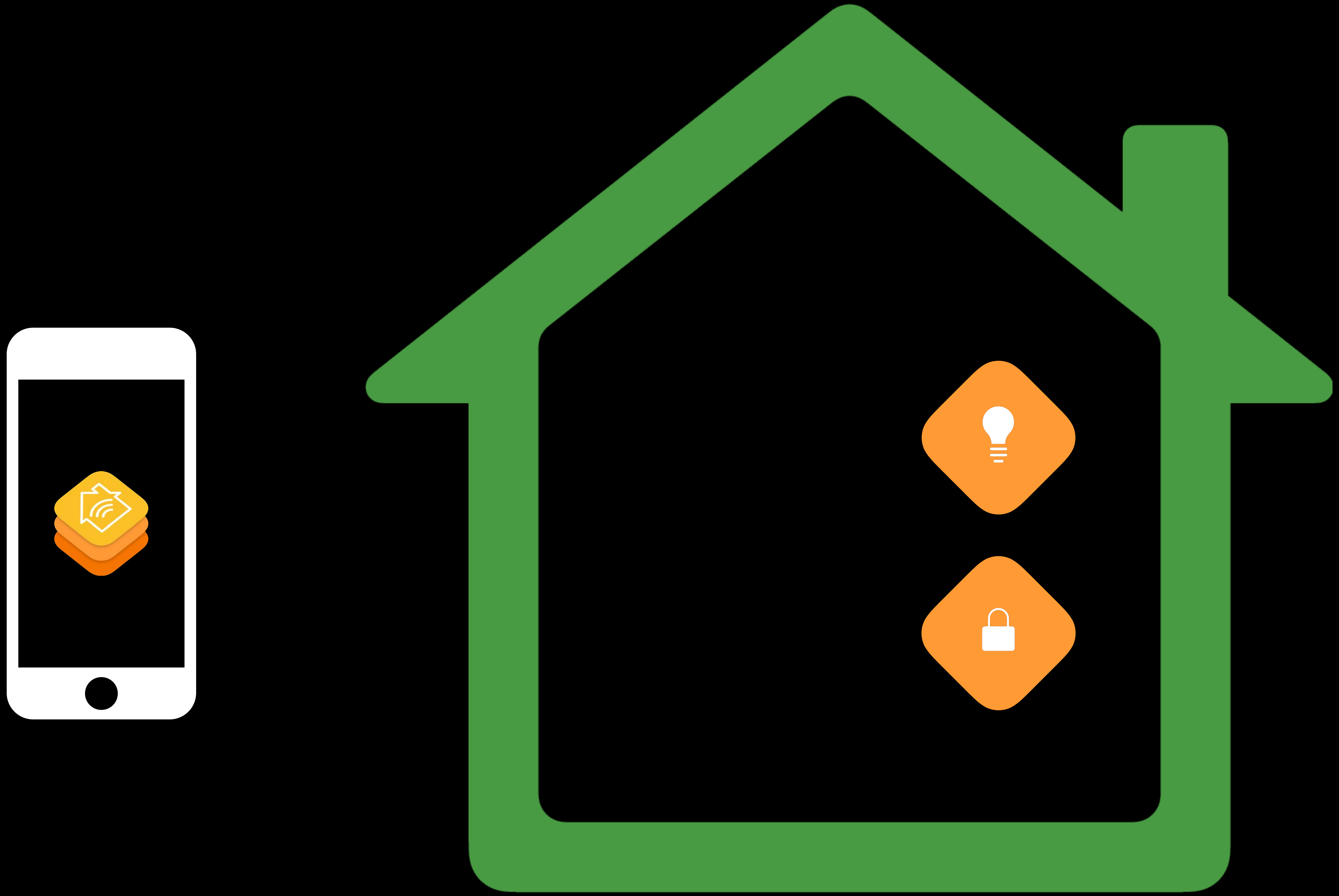
Accessory Communication

Accessory Communication



Remote Access

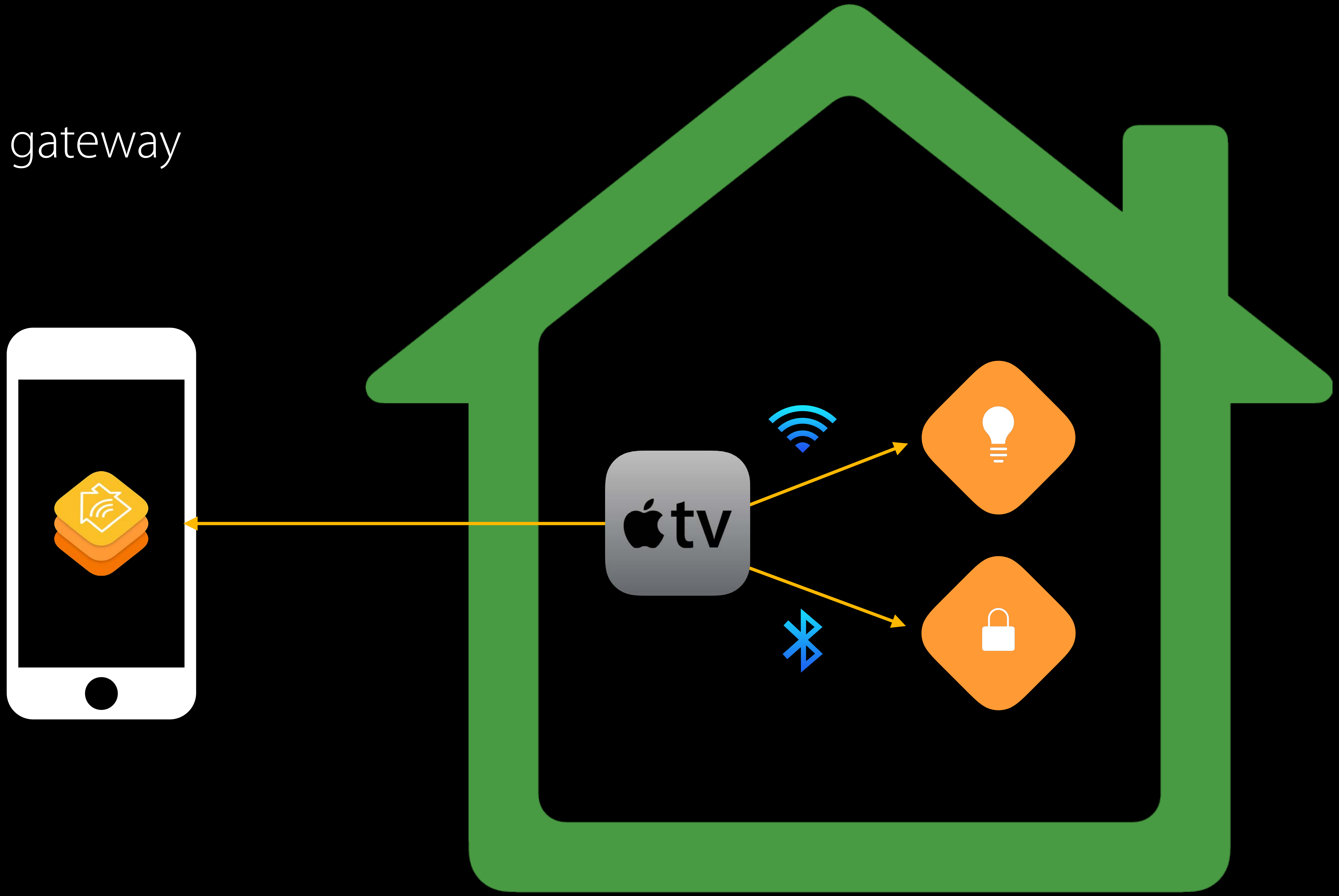
Apple TV



Remote Access

Apple TV

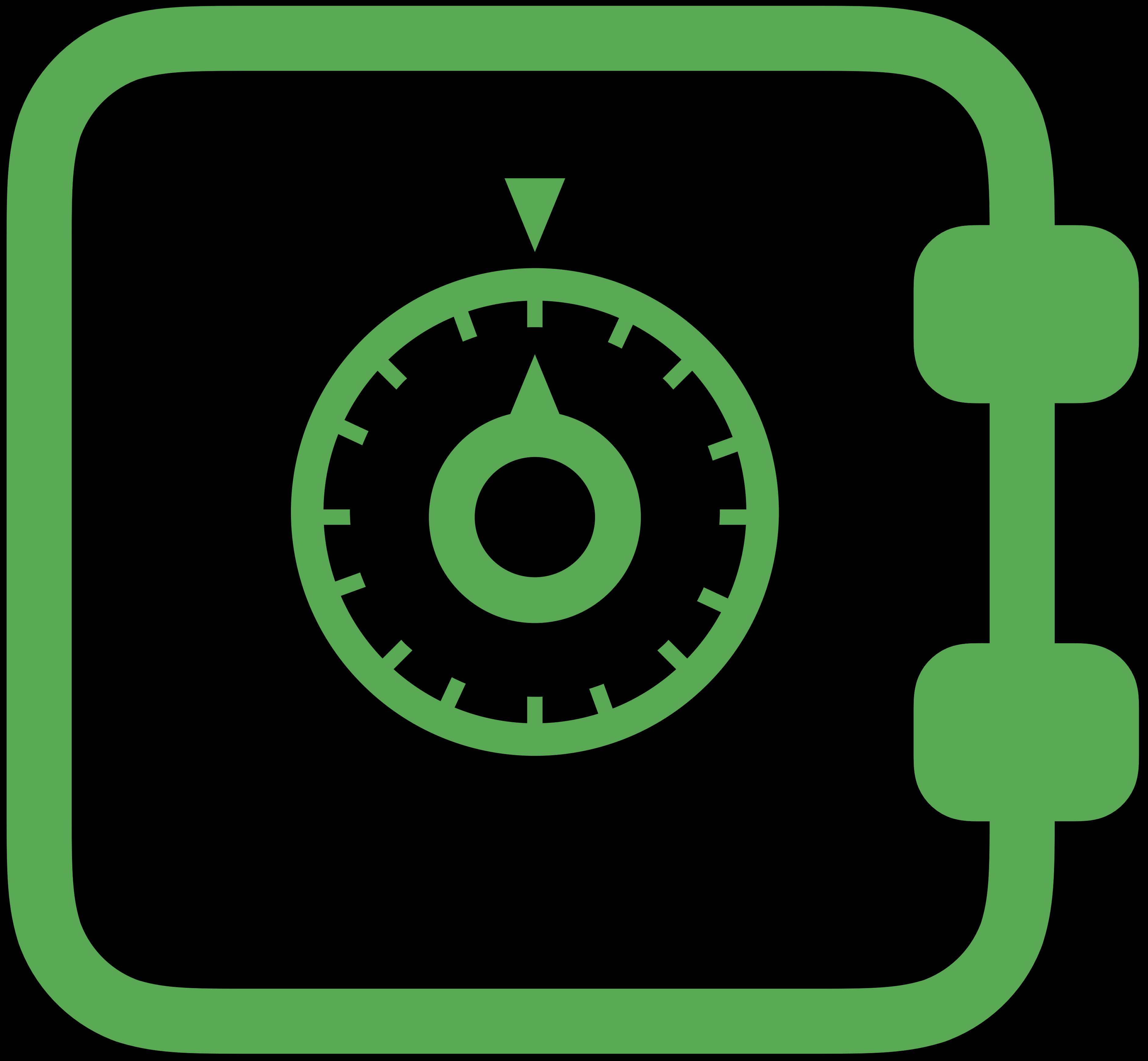
Remote access gateway



Security

Security

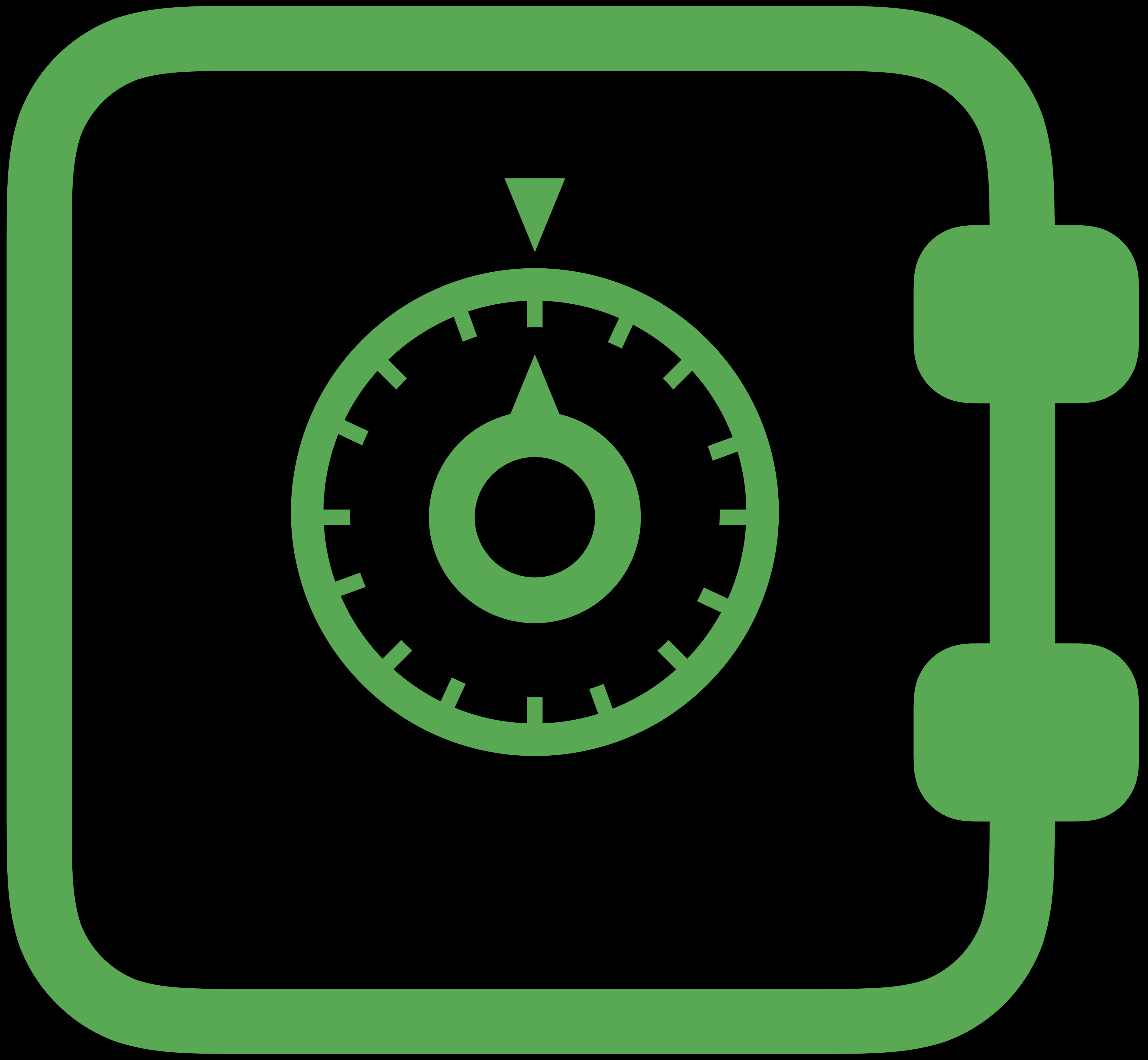
End-to-end secure



Security

End-to-end secure

Perfect Forward Secrecy

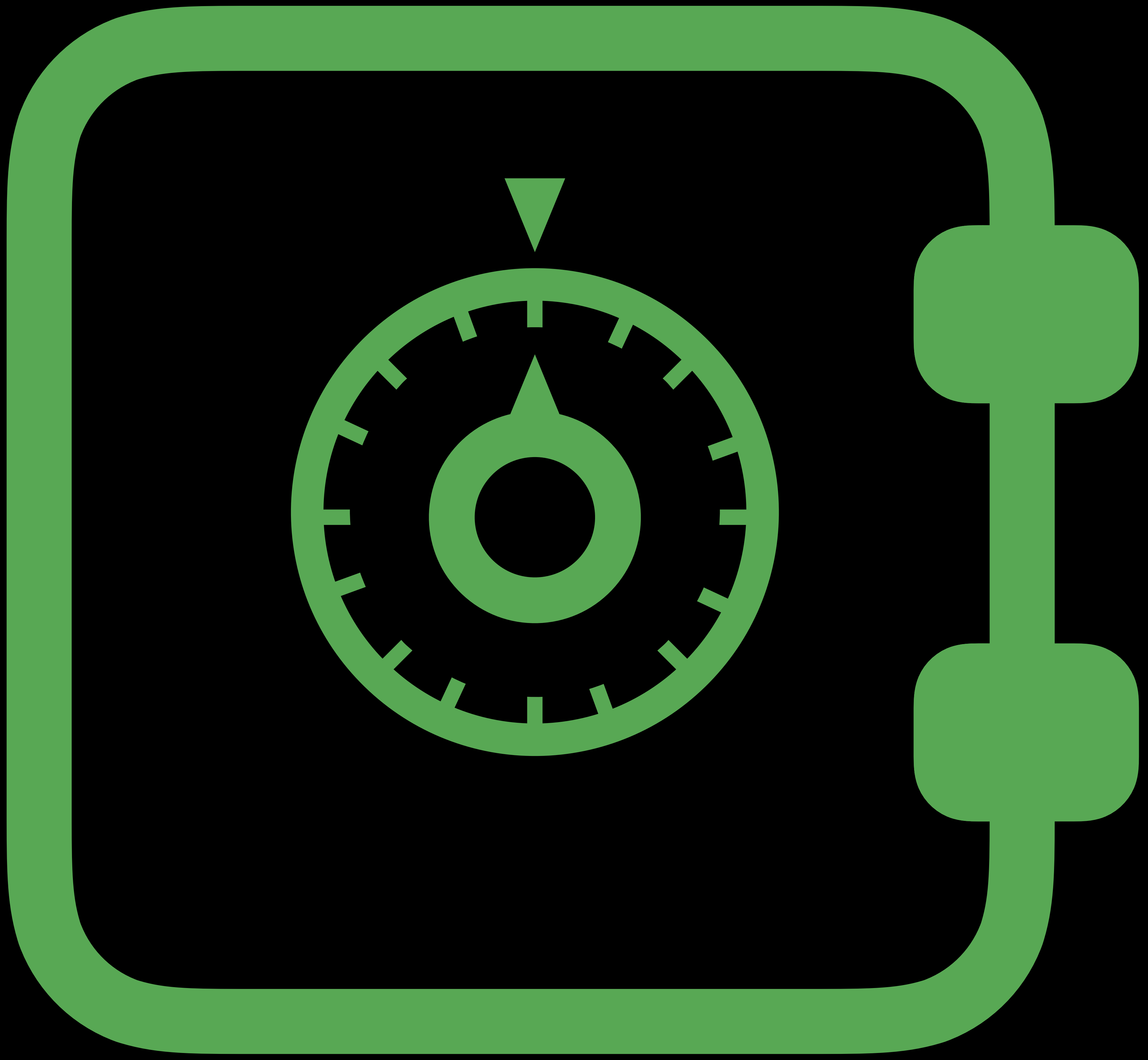


Security

End-to-end secure

Perfect Forward Secrecy

Data is private



HomeKit Accessories

HomeKit Accessories

What’s New in HomeKit

What’s New in HomeKit

What’s New in HomeKit

Platform

What's New in HomeKit

Platform

New Accessories

What’s New in HomeKit

Platform

New Accessories

Framework Updates

What’s New in HomeKit

Platform

New Accessories

Framework Updates

Platform

Platform

Home App

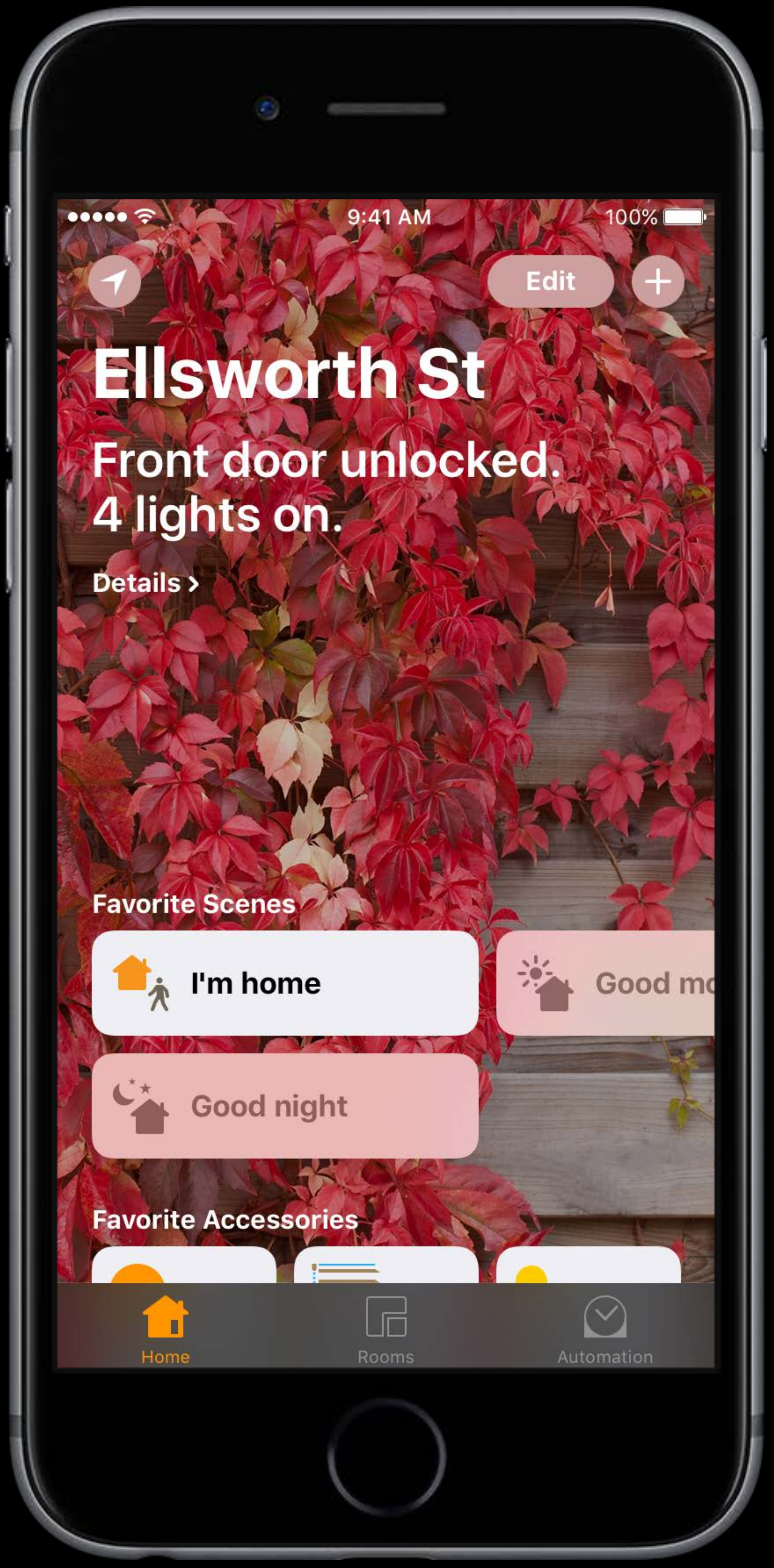
Platform

Home App

Home App

iOS devices

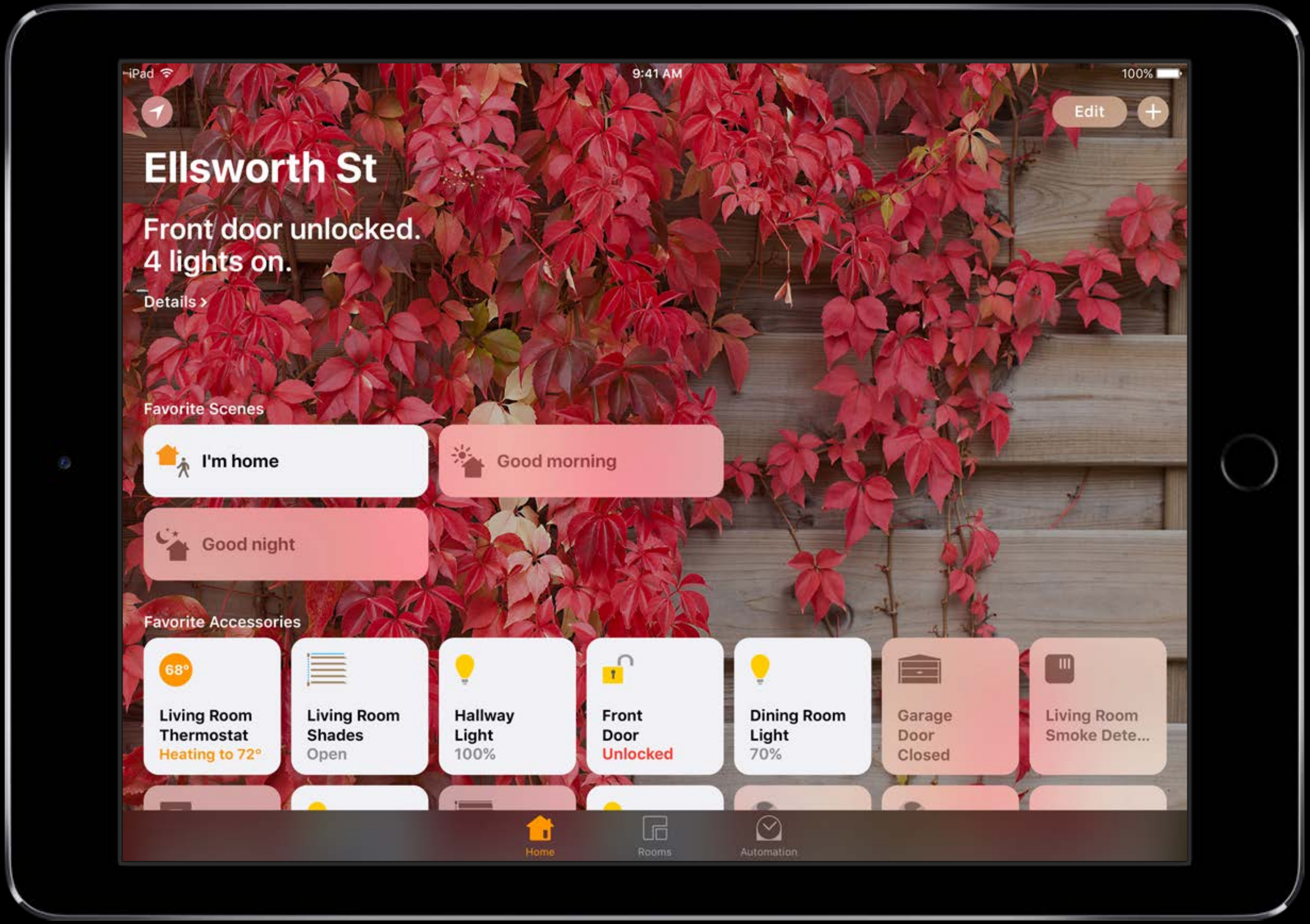
NEW



Home App

iOS devices

NEW



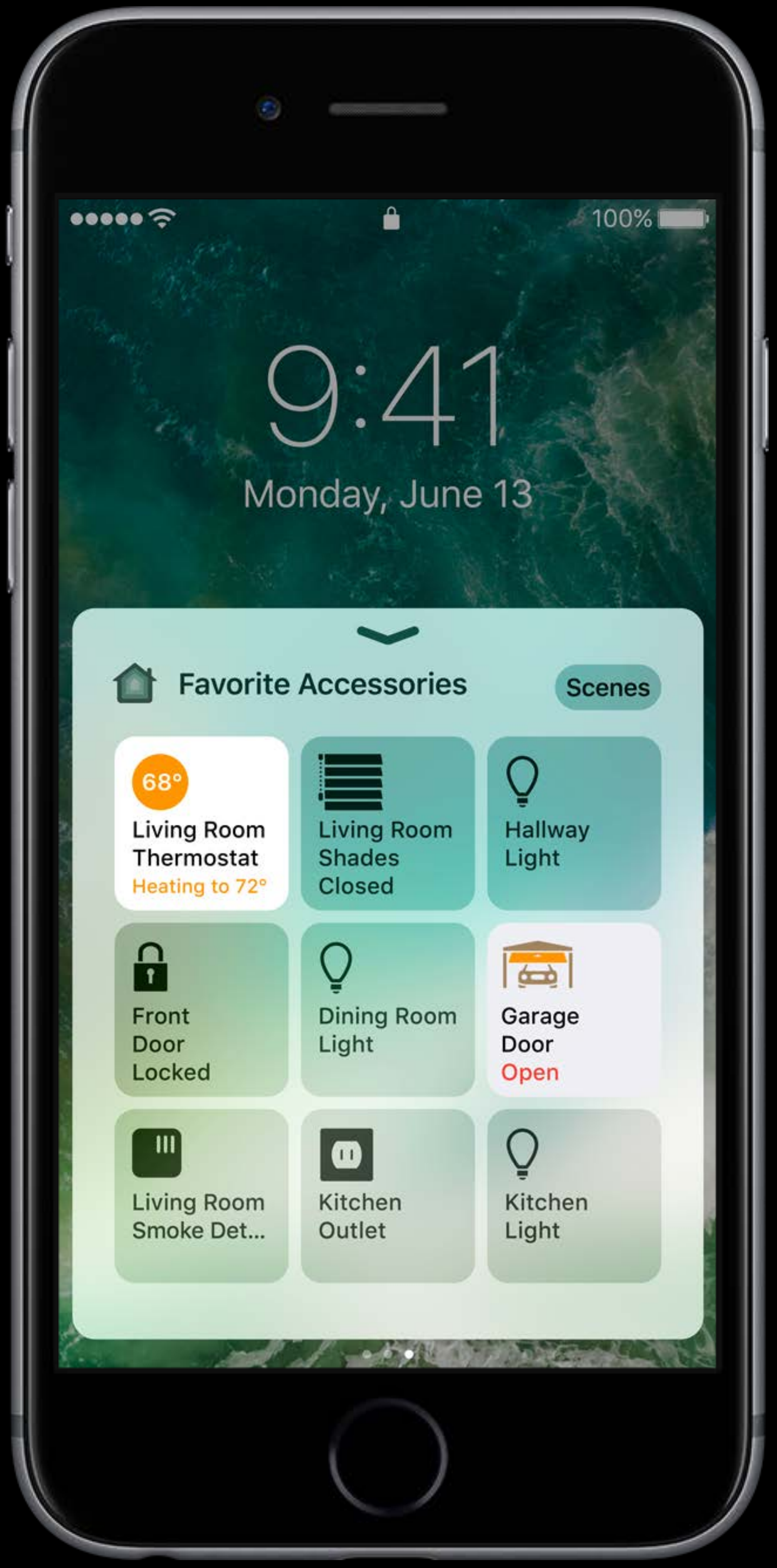
Home App

Apple Watch

NEW

Control Center

NEW



Platform

Remote Access and Automation

Remote Access and Automation

Apple TV

Remote Access and Automation

Apple TV

NEW

Remote Access and Automation

NEW

Apple TV

Remote access

Remote Access and Automation

NEW

Apple TV

Remote access

Automation

Remote Access and Automation

NEW

Apple TV

Remote access

Automation

- Event & timer triggers

Remote Access and Automation

NEW

Apple TV

Remote access

Automation

- Event & timer triggers

Access control for shared users

Remote Access and Automation

NEW

Apple TV

Remote access

Automation

- Event & timer triggers

Access control for shared users

- Administrator access

Remote Access and Automation

NEW

Apple TV

Remote access

Automation

- Event & timer triggers

Access control for shared users

- Administrator access
- Control remote access

Remote Access and Automation

NEW

Apple TV and iPad

Remote access

Automation

- Event & timer triggers

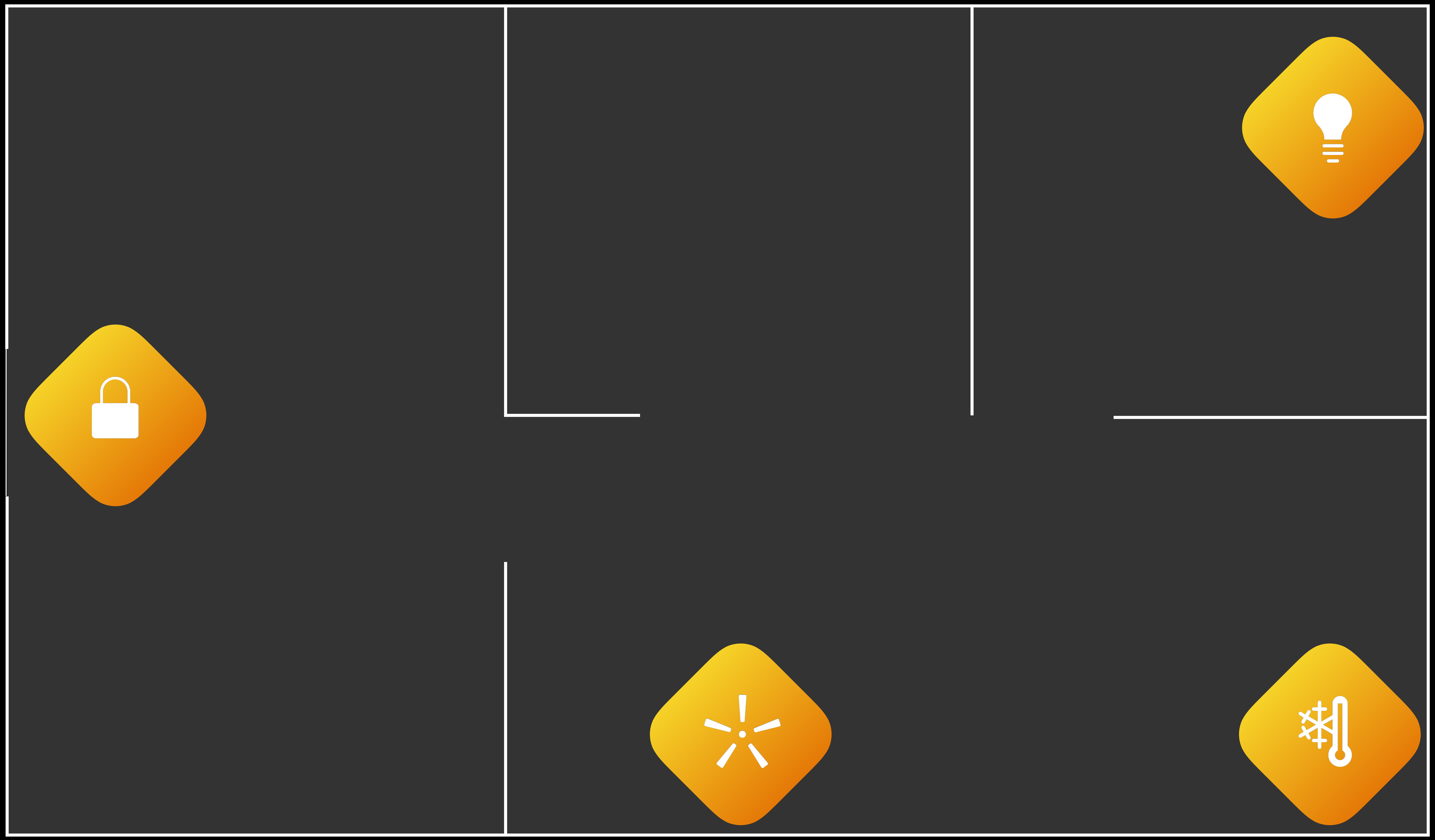
Access control for shared users

- Administrator access
- Control remote access

Also supported on iPad

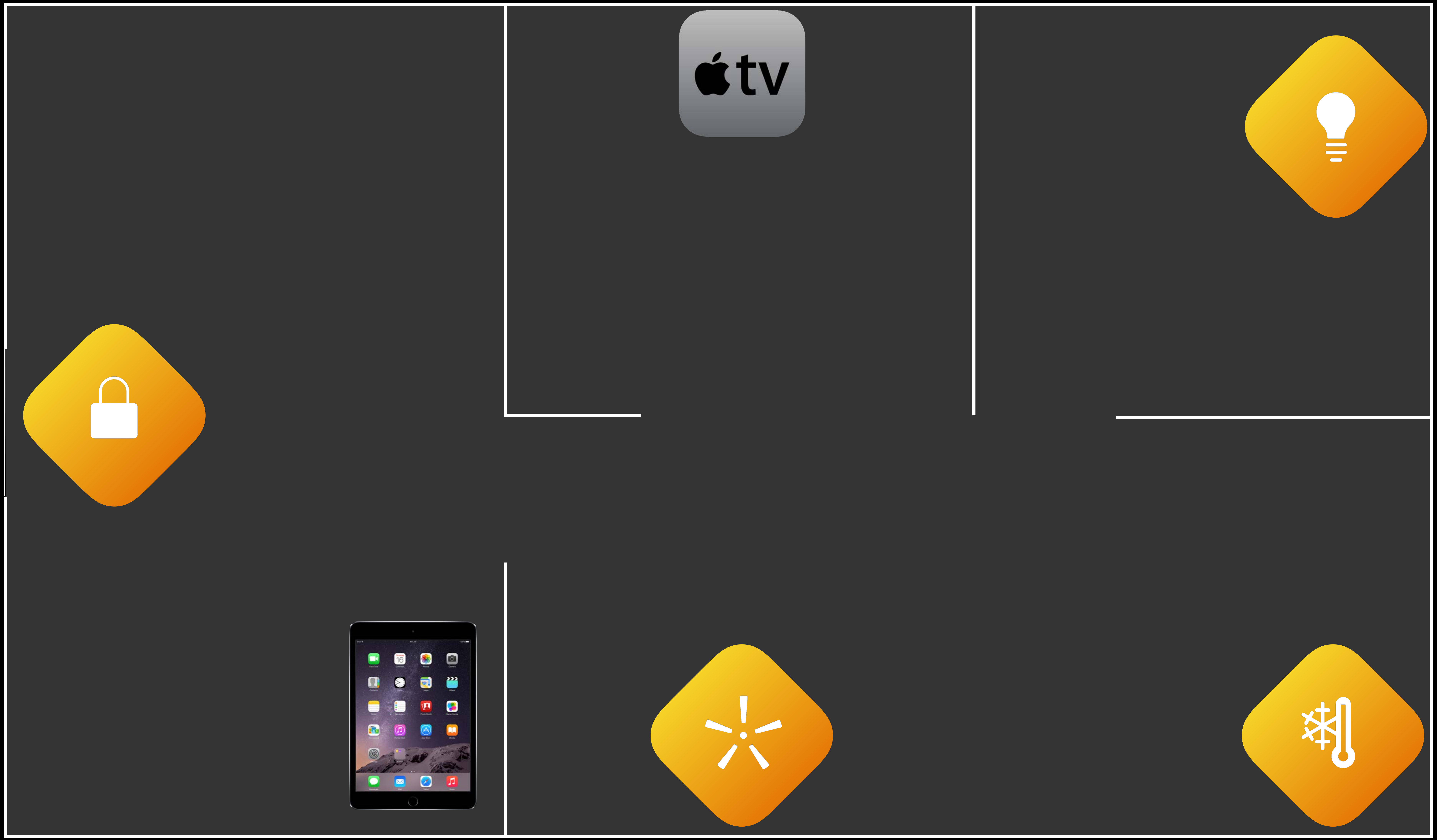
Multiple Remote Access Devices

NEW



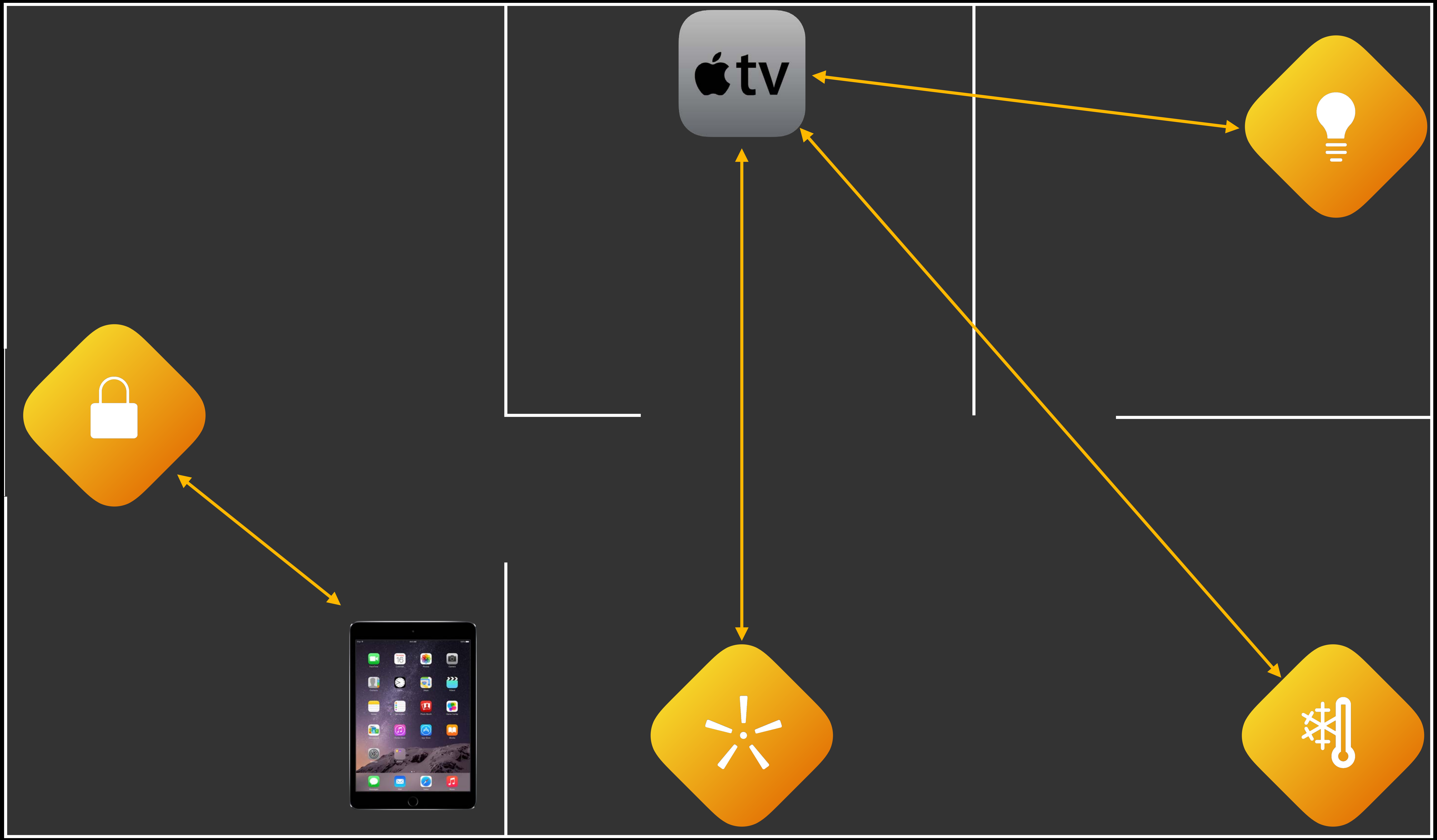
Multiple Remote Access Devices

NEW



Multiple Remote Access Devices

NEW



Platform

tvOS 10

HomeKit Framework

tvOS

NEW

HomeKit Framework

tvOS

NEW

View home configuration

HomeKit Framework

tvOS

NEW

View home configuration

Control accessories

HomeKit Framework

tvOS

NEW

View home configuration

Control accessories

Execute scenes

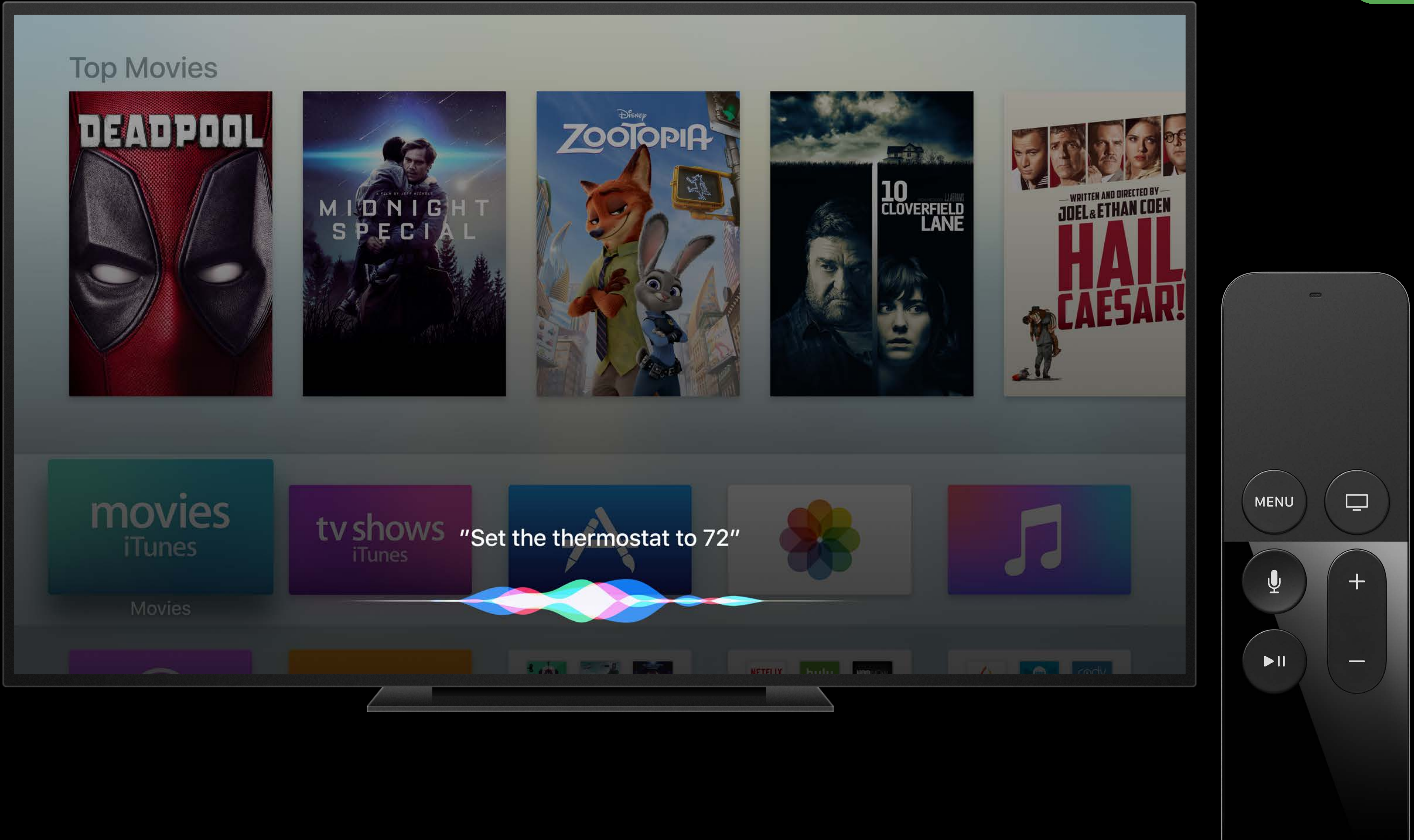
tvOS HomeKit Framework

NEW

Siri

tvOS HomeKit Framework

NEW



Platform Recap

Platform Recap

Home App

Platform Recap

Home App

Remote Access and Automation

Platform Recap

Home App

Remote Access and Automation

HomeKit Framework for tvOS

What’s New in HomeKit

Platform

New Accessories

Framework Updates

What’s New in HomeKit

- Platform
- New Accessories
- Framework Updates

New Accessories

Air Treatment Accessories

NEW

Air Treatment Accessories

NEW

Air Conditioners / Heaters

Air Treatment Accessories

NEW

Air Conditioners / Heaters

Air Purifiers

Air Treatment Accessories

NEW

Air Conditioners / Heaters

Air Purifiers

Humidifiers

Camera Accessories

NEW

Camera Accessories

NEW

Display live streams

Camera Accessories

NEW

Display live streams

Display still images

Camera Accessories

NEW

Display live streams

Display still images

Control the camera settings

Camera Accessories

NEW

Display live streams

Display still images

Control the camera settings

Control the speaker and microphone

Doorbell

NEW

Doorbell

NEW

Doorbell

NEW

Generates an event

Doorbell

NEW

Generates an event

Volume control

Doorbell

NEW

Generates an event

Volume control

Visual indicator control

Doorbell Camera

NEW

Generates an event

Volume control

Visual indicator control

Service Association

NEW

Service Association

NEW

Service Association

NEW

Service Association

NEW

Service Association

NEW

Automatically associates services

Service Association

NEW

Automatically associates services

Quick action controls

Rich Notifications

NEW

Rich Notifications

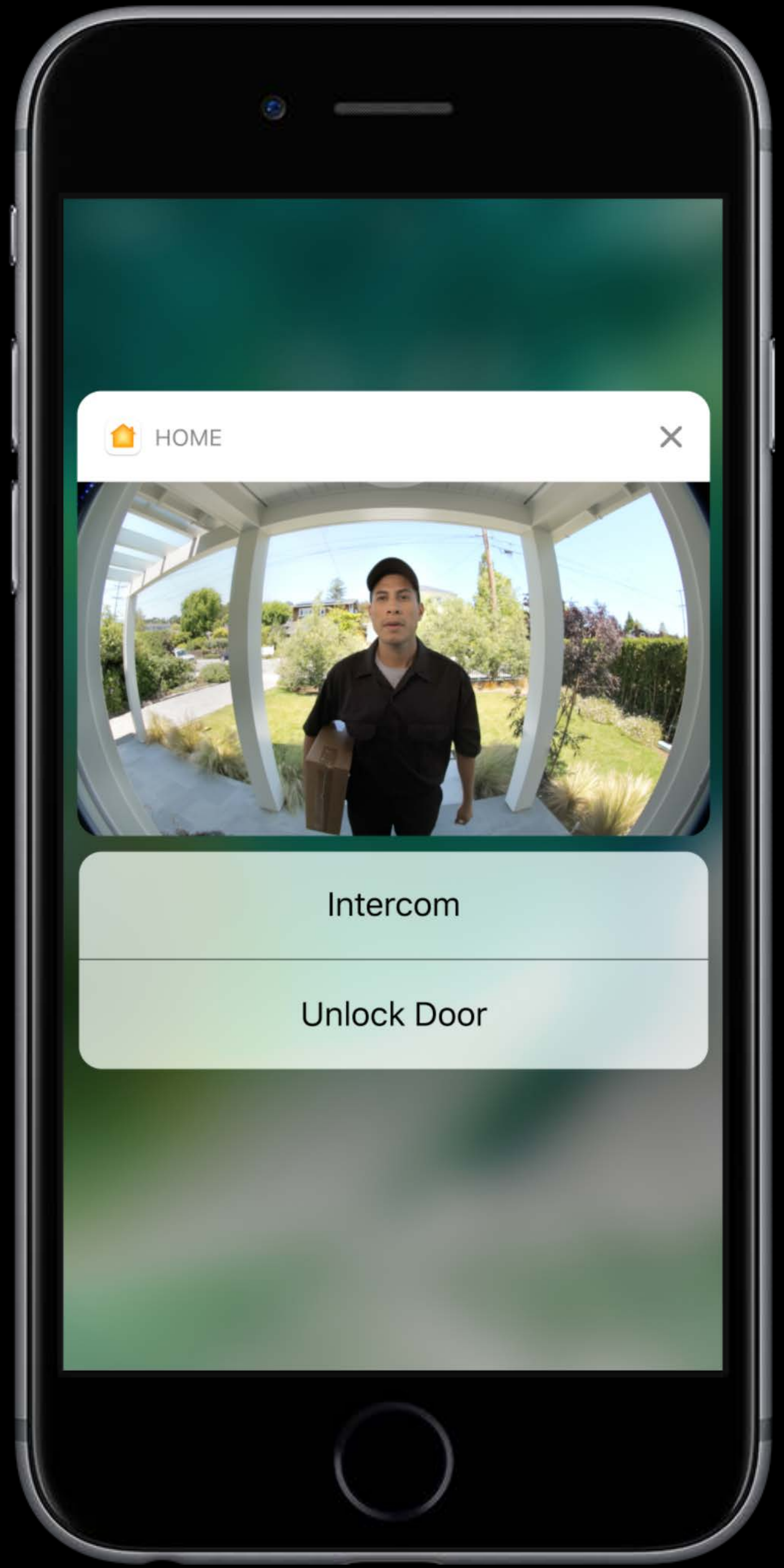
NEW

Rich Notifications

NEW

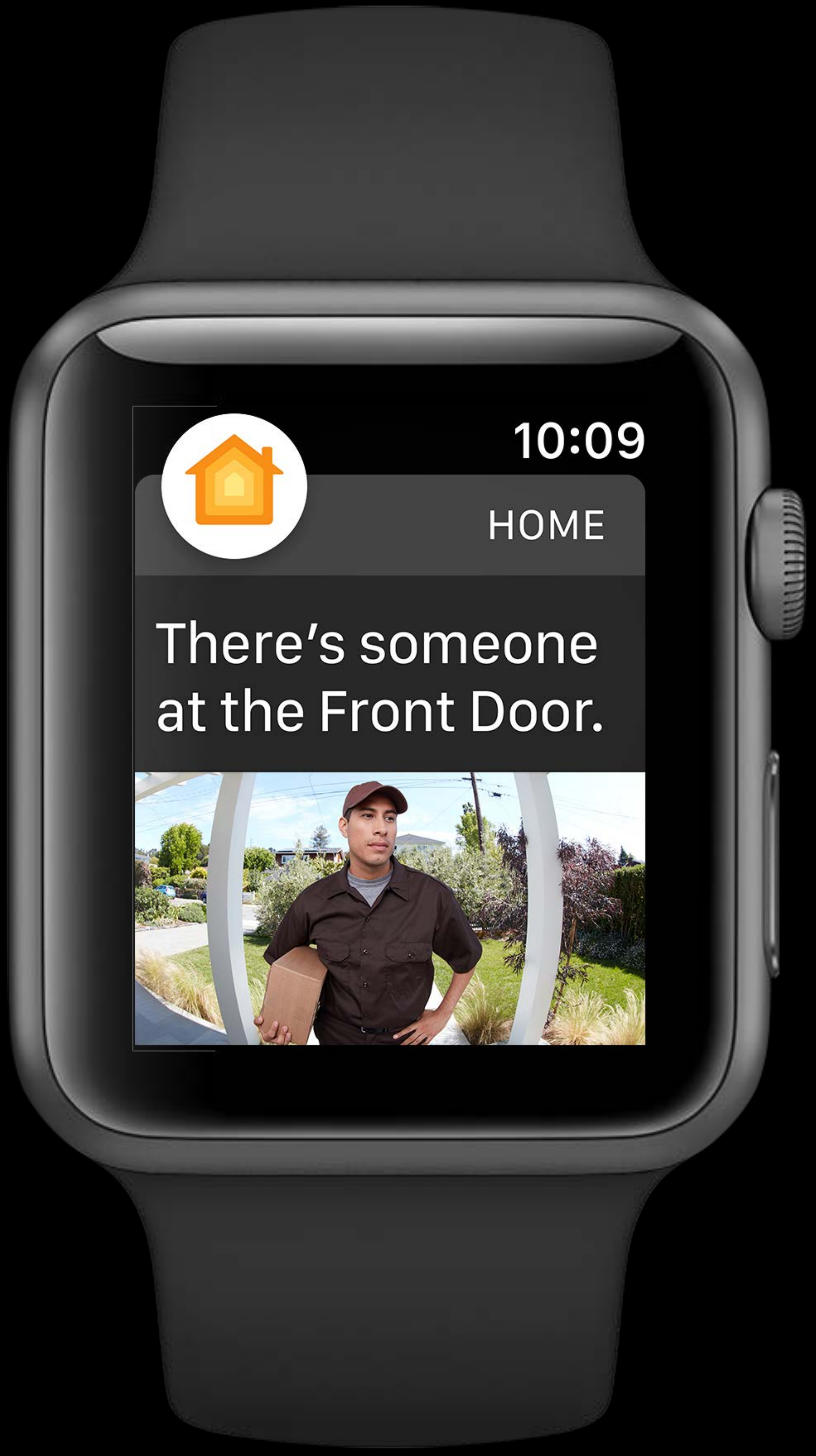
Rich Notifications

NEW



Rich Notifications

NEW



There's someone at the Front Door.

Rich Notifications

NEW

There's someone at the Front Door.

What's New in HomeKit

Platform

New Accessories

Framework Updates

What’s New in HomeKit

Platform

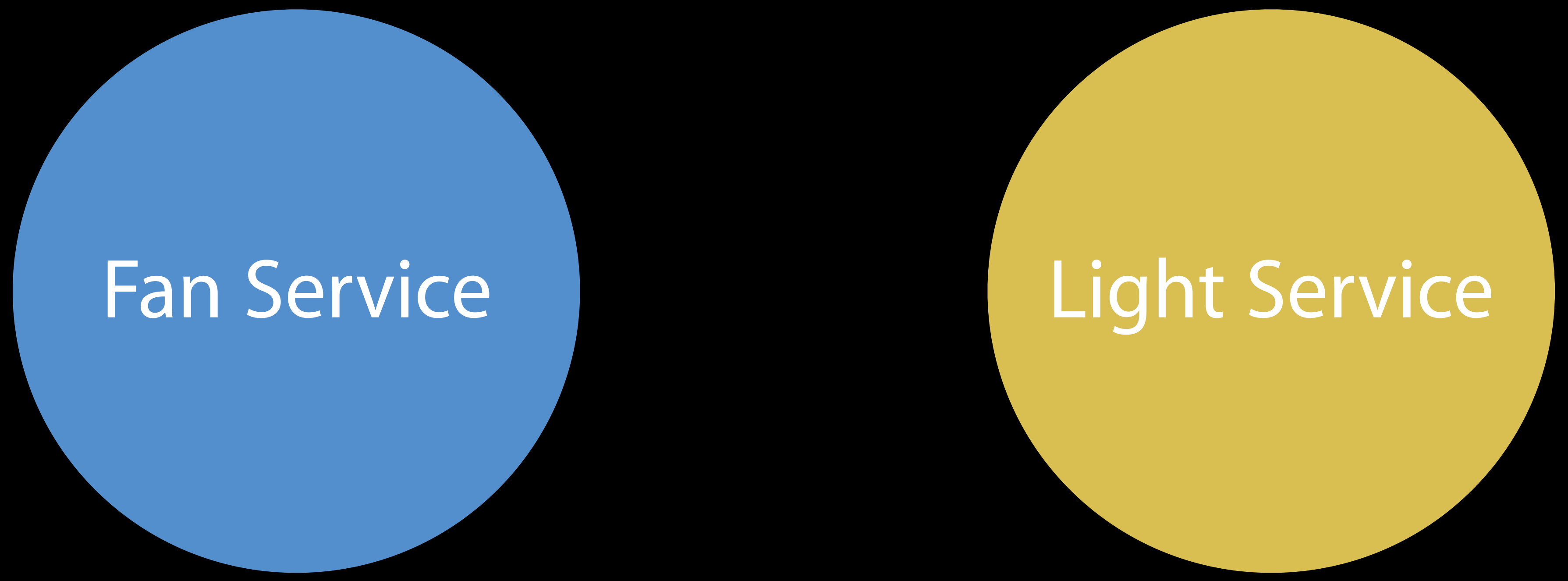
New Accessories

Framework Updates

Framework Updates

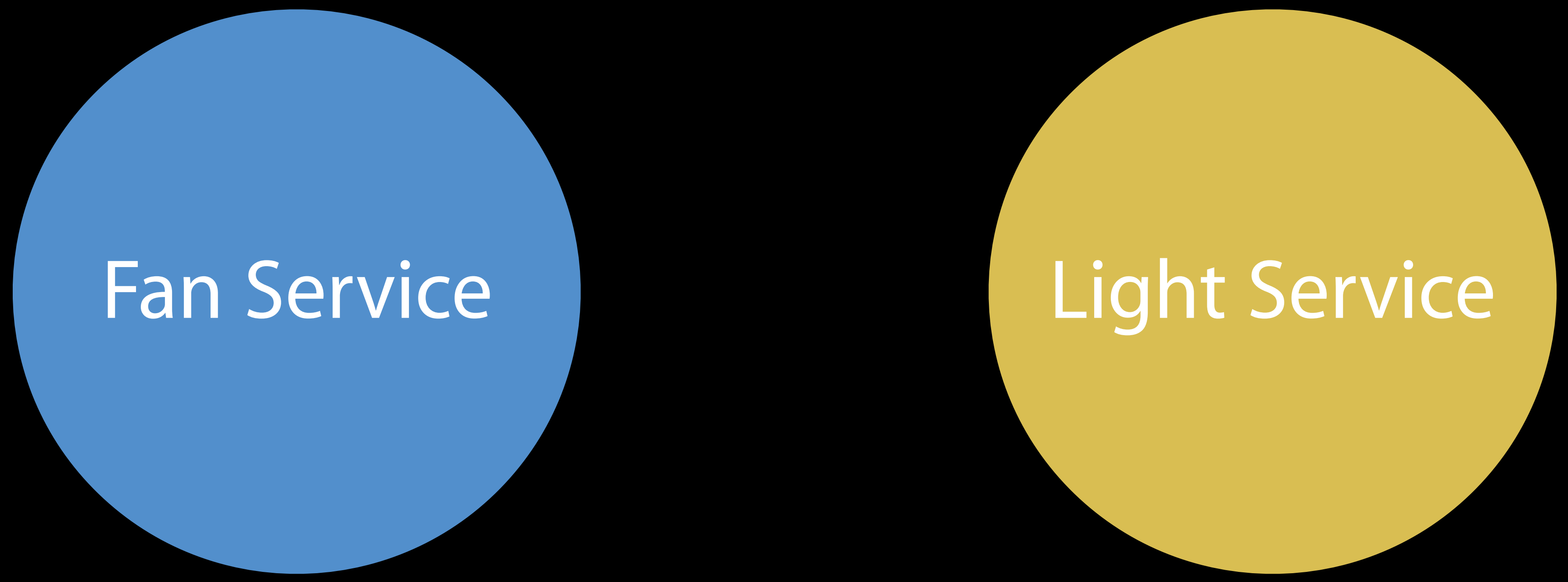
Primary Service

NEW



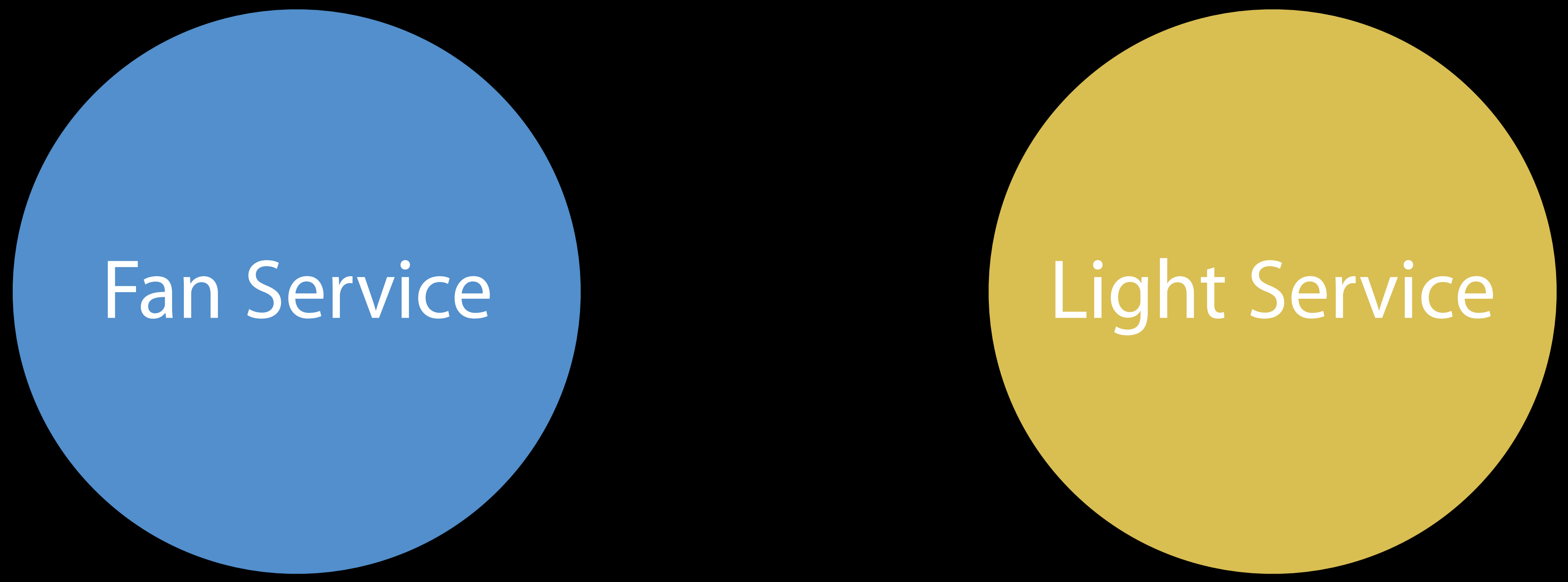
Primary Service

NEW



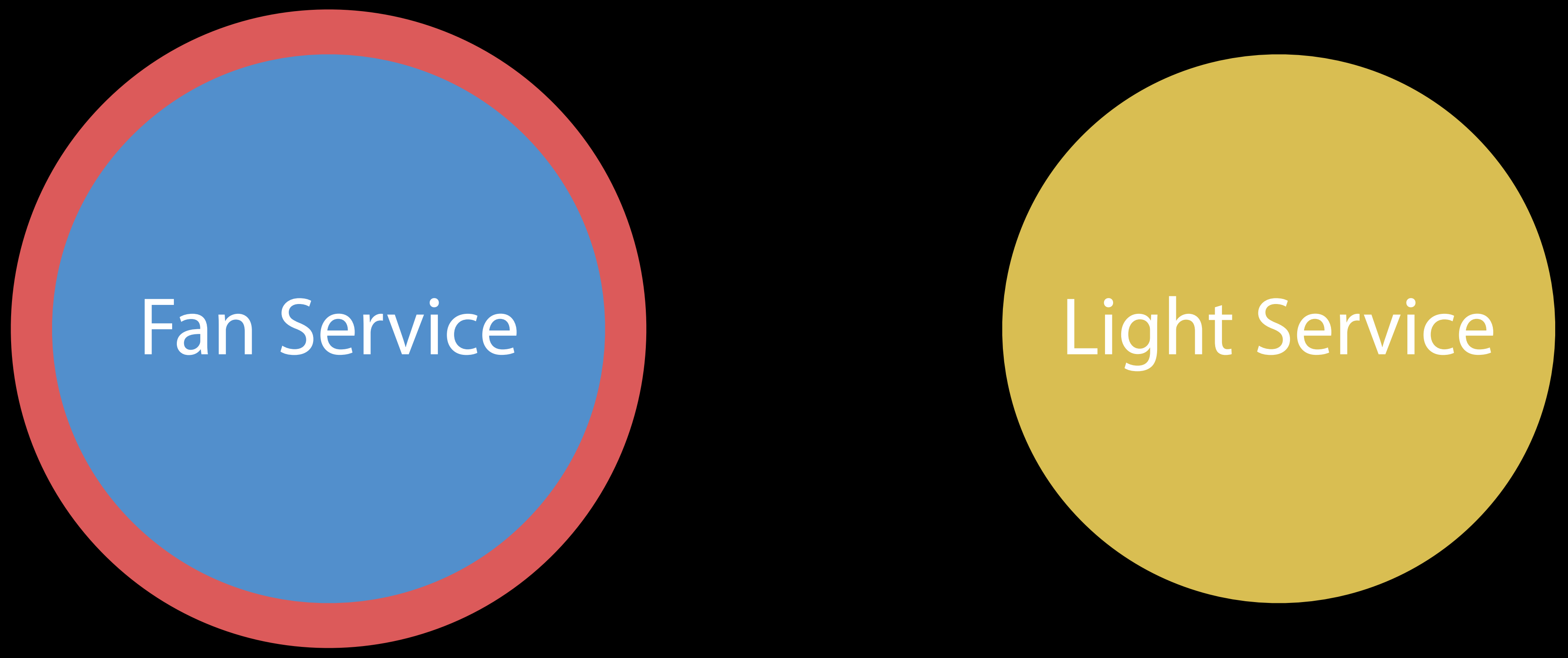
Primary Service

NEW



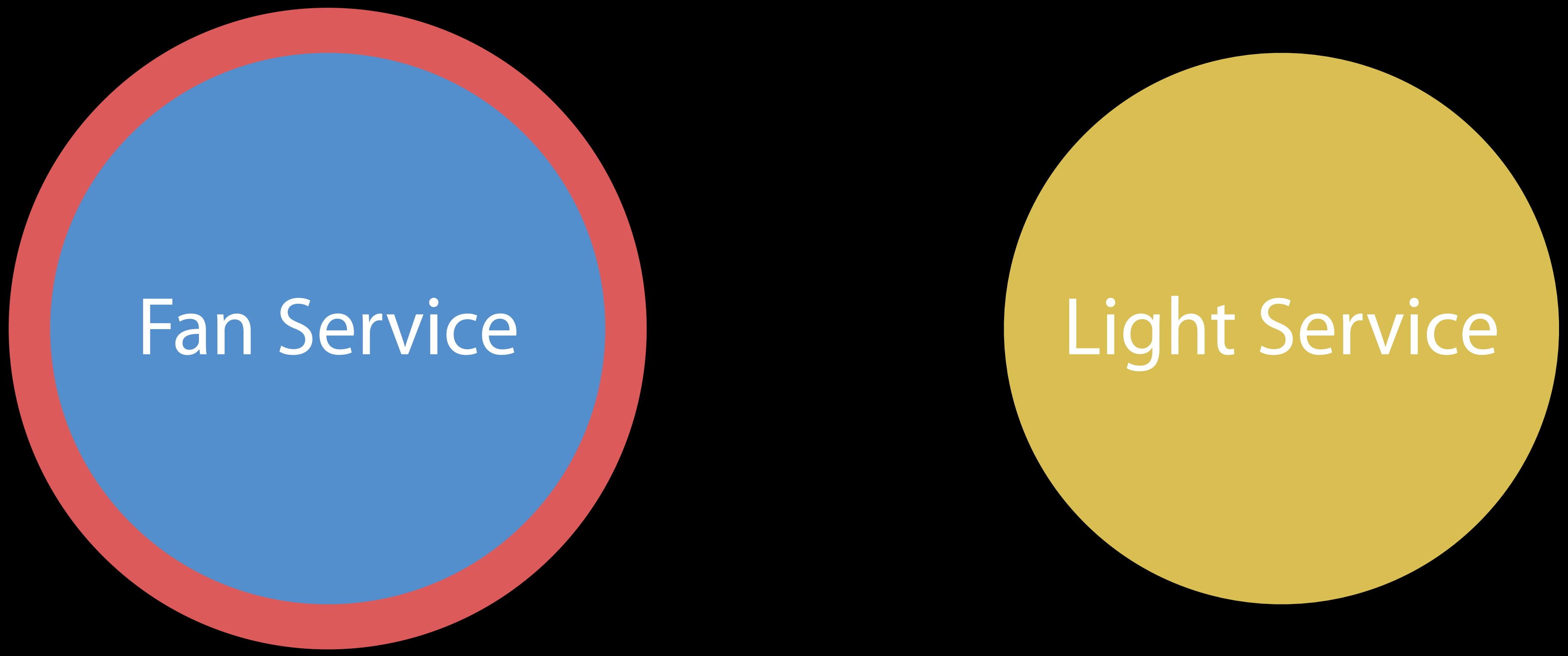
Primary Service

NEW



Primary Service

NEW

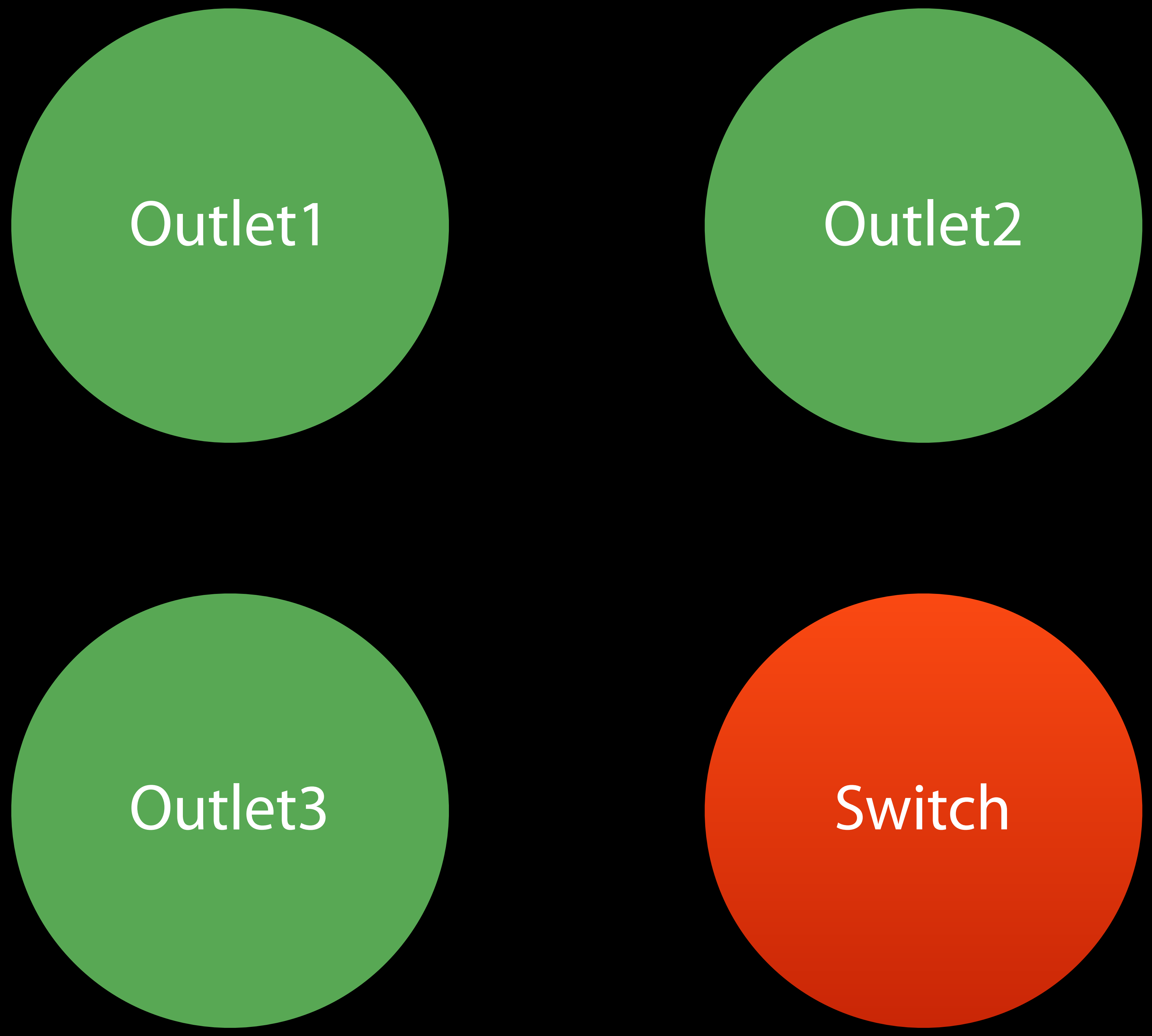


```
public class HMService : NSObject {

    ...
    public var isPrimaryService: Bool { get }
```

Linked Services

NEW



Linked Services

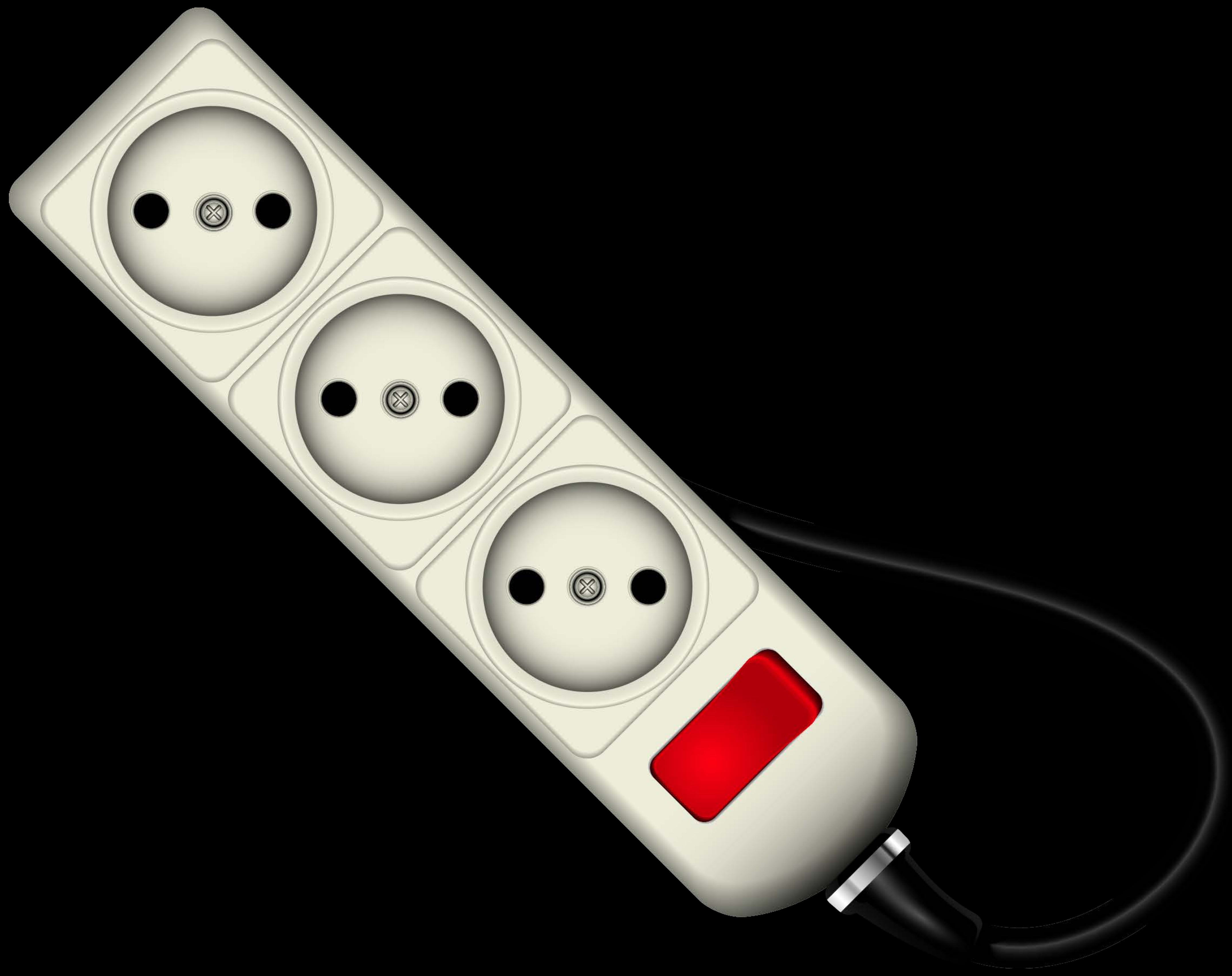
NEW

Outlet1

Outlet2

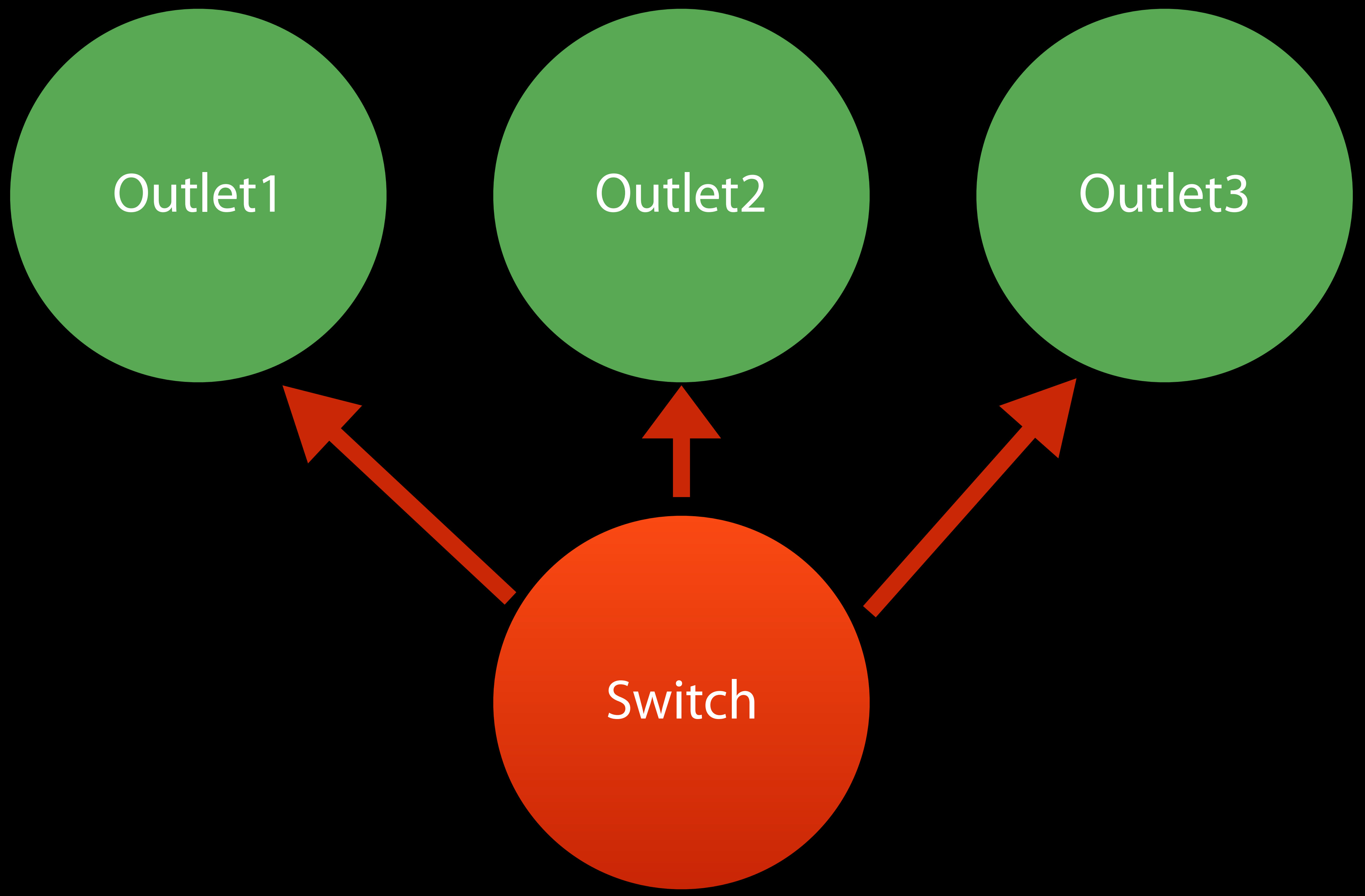
Outlet3

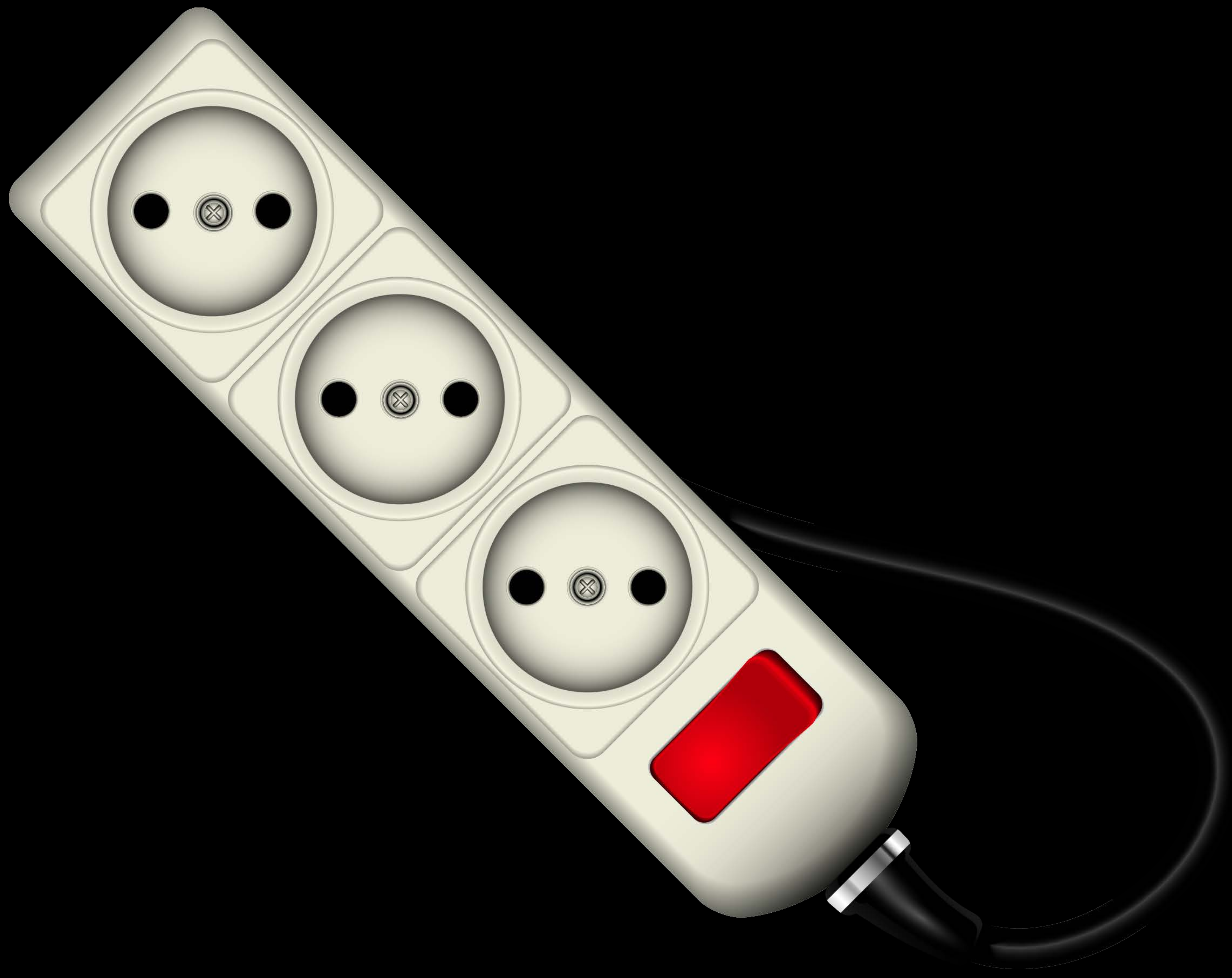
Switch



Linked Services

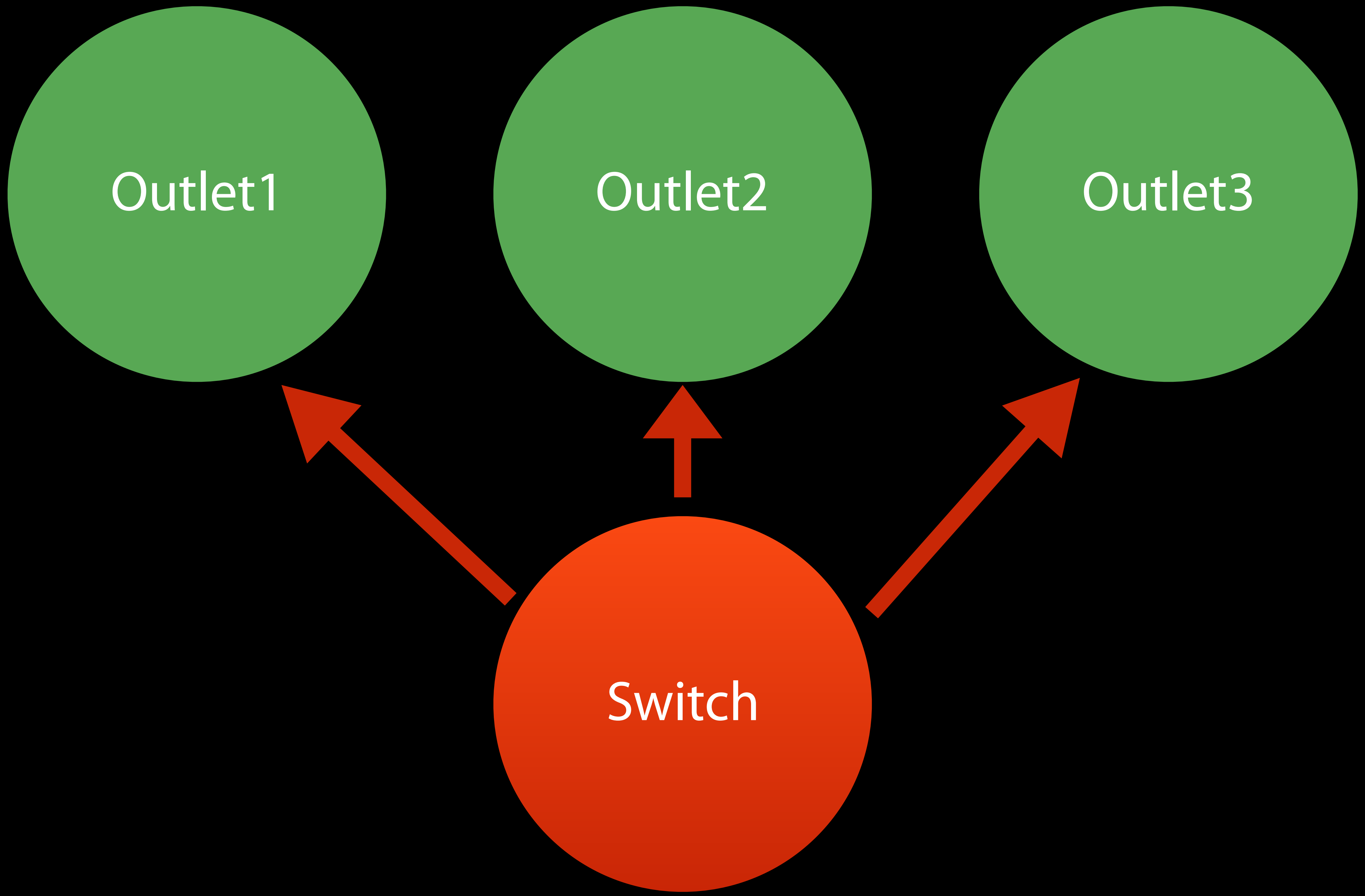
NEW

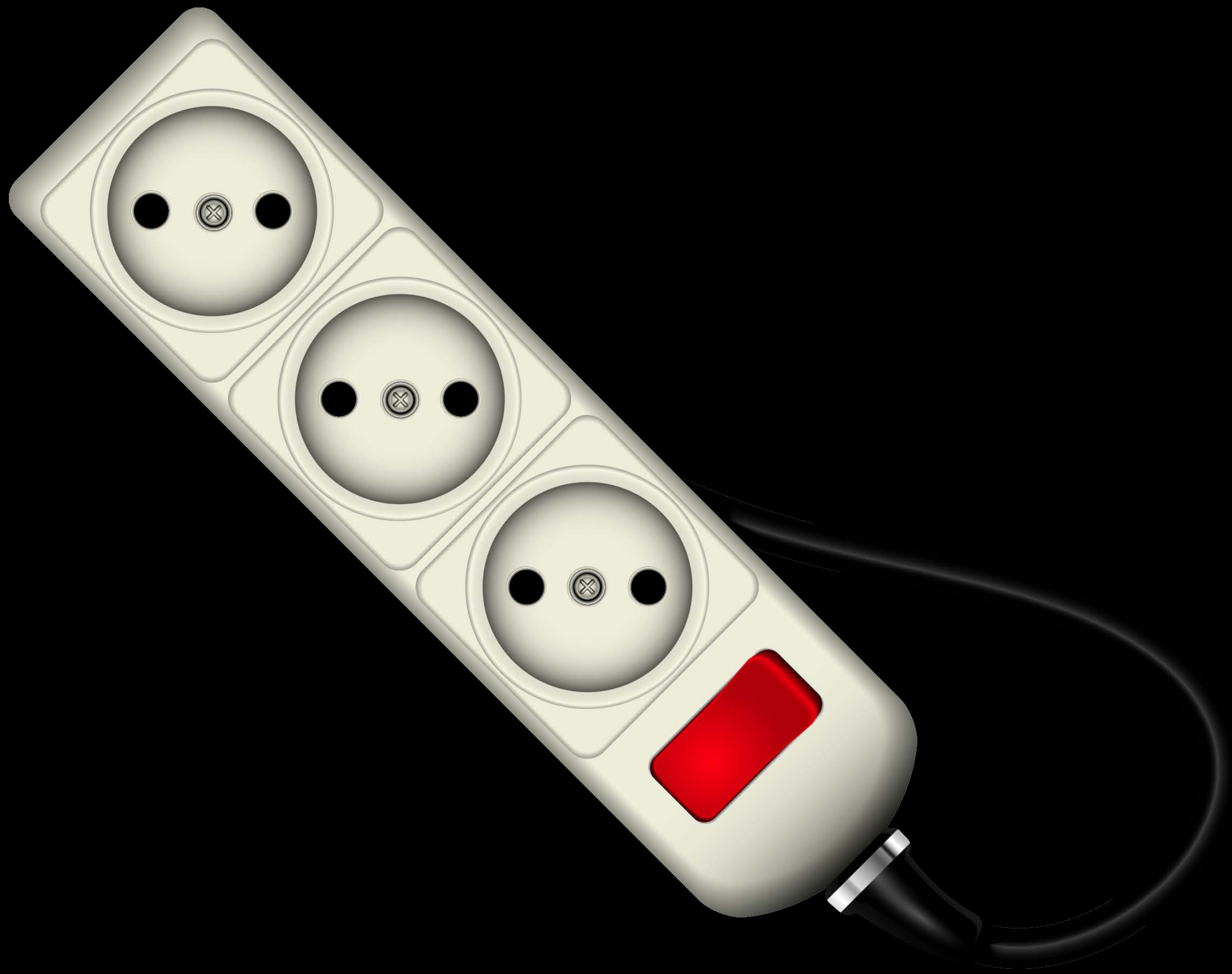




Linked Services

NEW





```
public class HMService : NSObject {

    ...
    public var linkedServices: [HMService]? { get }
```

Valid Values

NEW

Valid Values

NEW

Target Heating Cooling State	
Min Value	0
Max Value	3
State	0 = Off 1 = Heat 2 = Cool 3 = Auto

Valid Values

NEW

Target Heating Cooling State	
Min Value	0
Max Value	3
State	0 = Off 1 = Heat ✗ 2 = Cool 3 = Auto ✗

Valid Values

NEW

Valid Values [0, 2]

Target Heating Cooling State		
Min Value	0	
Max Value	3	
State	0 = Off	
	1 = Heat	⊗
	2 = Cool	
	3 = Auto	⊗

```
public class HMCharacteristicMetadata : NSObject {

    ...
    public var validValues: [NSNumber]? { get }
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

```
// HMAccessoryBrowserDelegate

   ...
   @objc optional func accessoryBrowser(_ browser: HMAccessoryBrowser,
      didFindNewAccessory accessory: HMAccessory)
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

```
// HMAccessoryBrowserDelegate

   ...
   @objc optional func accessoryBrowser(_ browser: HMAccessoryBrowser,
      didFindNewAccessory accessory: HMAccessory)
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

```
// HMAccessoryBrowserDelegate

   ...
   @objc optional func accessoryBrowser(_ browser: HMAccessoryBrowser,
      didFindNewAccessory accessory: HMAccessory)
```

```
extension HMHome {

   ...
   public func addAccessory(_ accessory: HMAccessory, completionHandler completion:
      (NSError?) -> Void)
```

Setting Up HomeKit Accessories

Find nearby accessories and add to home

```
public class HMAccessoryBrowser : NSObject {

   ...
   public func startSearchingForNewAccessories()
```

```
// HMAccessoryBrowserDelegate

   ...
   @objc optional func accessoryBrowser(_ browser: HMAccessoryBrowser,
      didFindNewAccessory accessory: HMAccessory)
```

```
extension HMHome {

   ...
   public func addAccessory(_ accessory: HMAccessory, completionHandler completion:
      (NSError?) -> Void)
```

Setting Up HomeKit Accessories

NEW

Find nearby accessories and add to home

Setting Up HomeKit Accessories

NEW

Find nearby accessories and add to home

```swift
extension HMHome {

   ...
   public func addAndSetupAccessories(completionHandler completion:
      (NSError?) -> Void)
```

✓

Setting Up HomeKit Accessories

NEW

Find nearby accessories and add to home

```
extension HMHome {

   ...
   public func addAndSetupAccessories(completionHandler completion:
      (NSError?) -> Void)
```

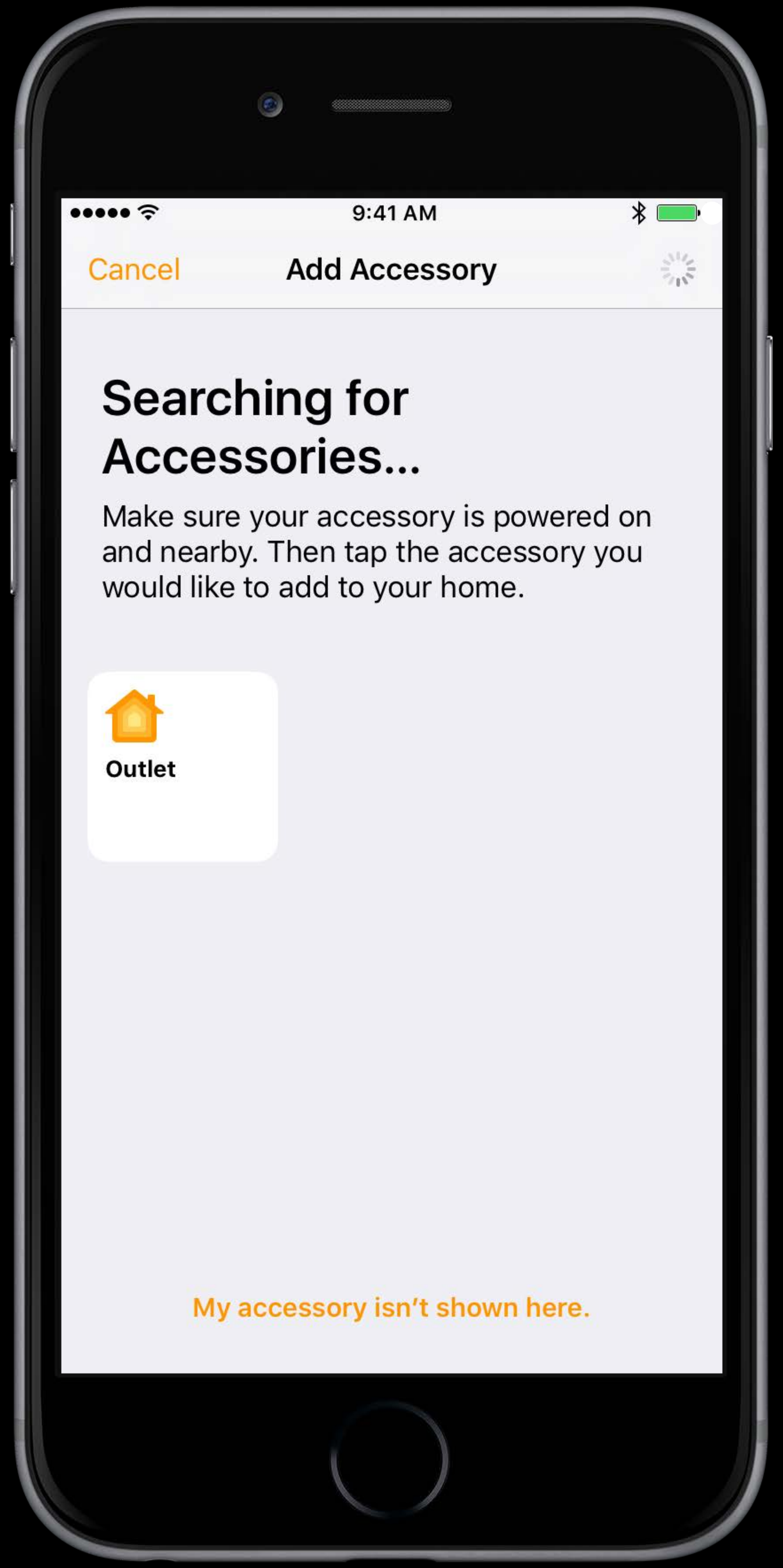
✓

Setting Up HomeKit Accessories

NEW

Setting Up HomeKit Accessories

NEW

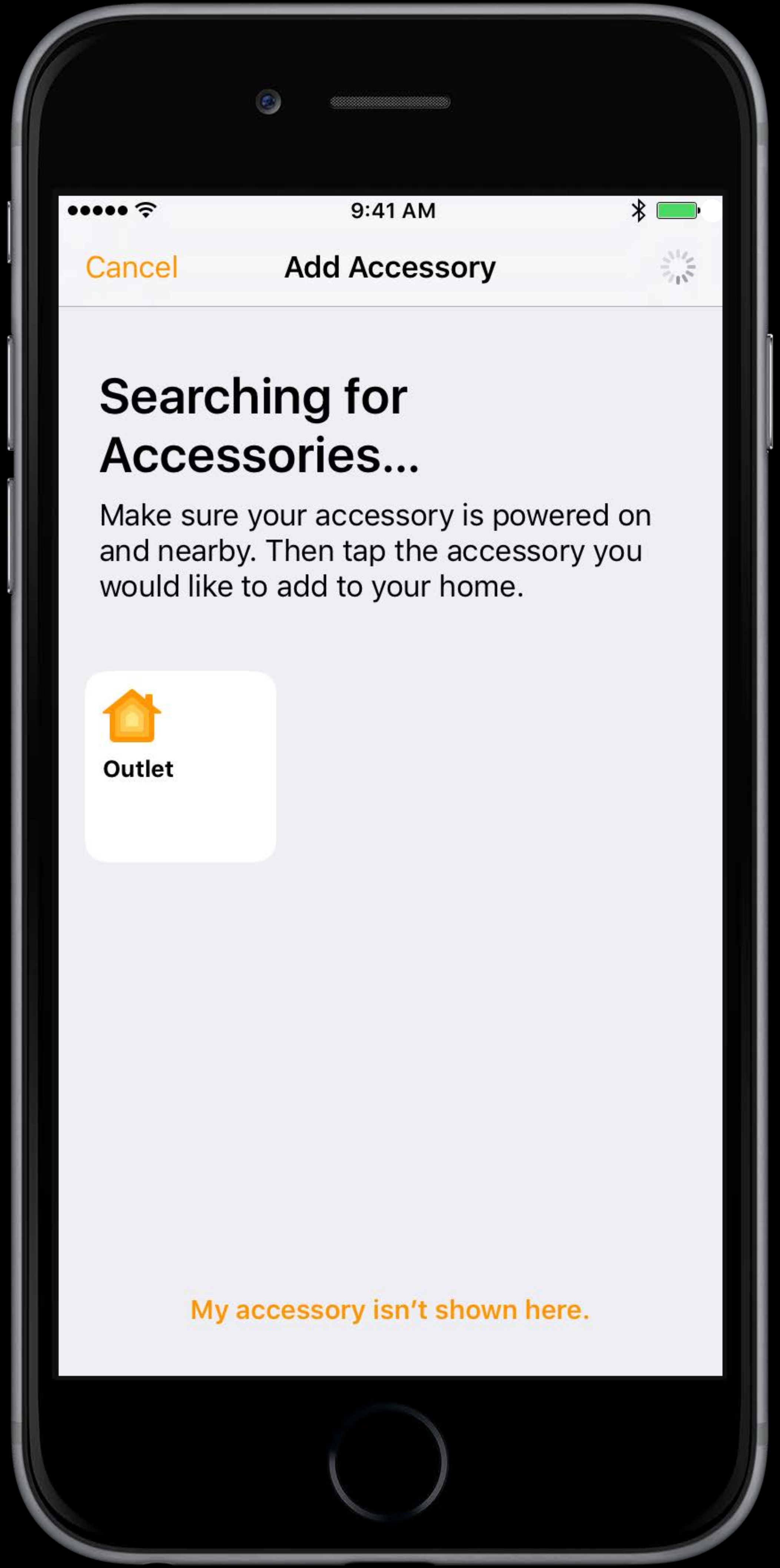


Searching for Accessories...

Make sure your accessory is powered on and nearby. Then tap the accessory you would like to add to your home.

Setting Up HomeKit Accessories

NEW

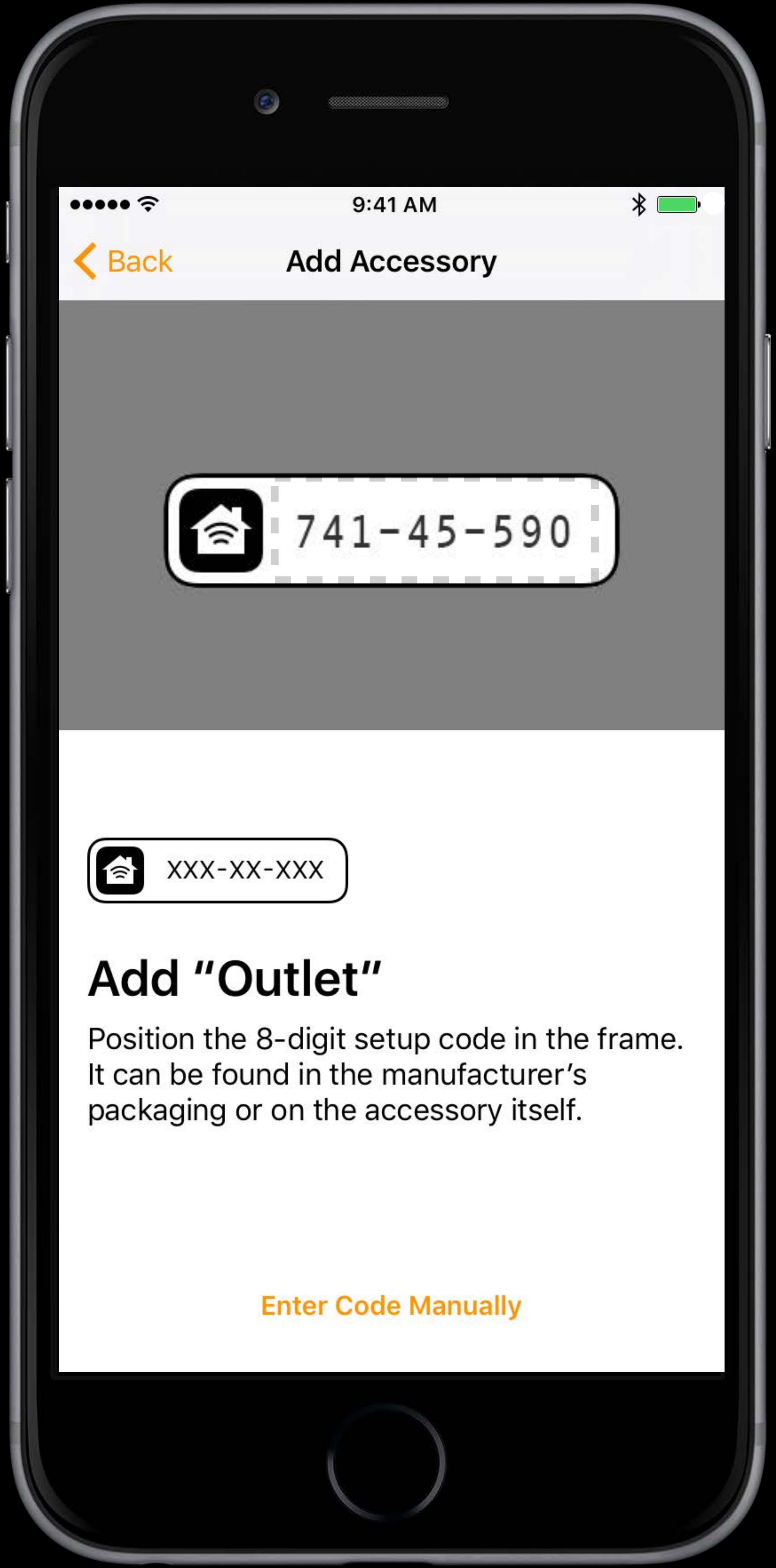

9:41 AM

Cancel Add Accessory

Searching for Accessories...

Make sure your accessory is powered on and nearby. Then tap the accessory you would like to add to your home.

My accessory isn't shown here.


9:41 AM

Back Add Accessory

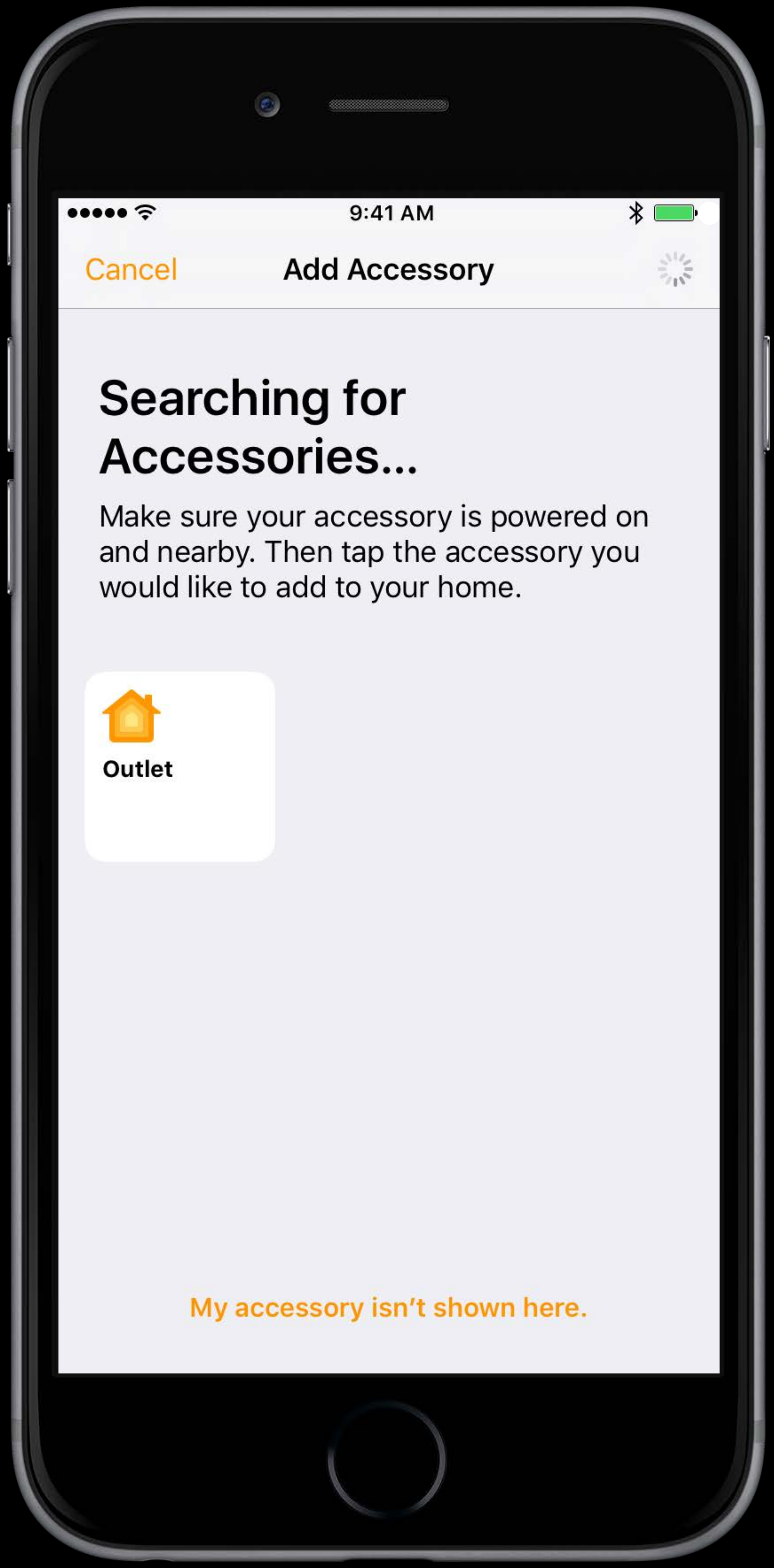
Add "Outlet"

Position the 8-digit setup code in the frame. It can be found in the manufacturer's packaging or on the accessory itself.

Enter Code Manually

Setting Up HomeKit Accessories

NEW

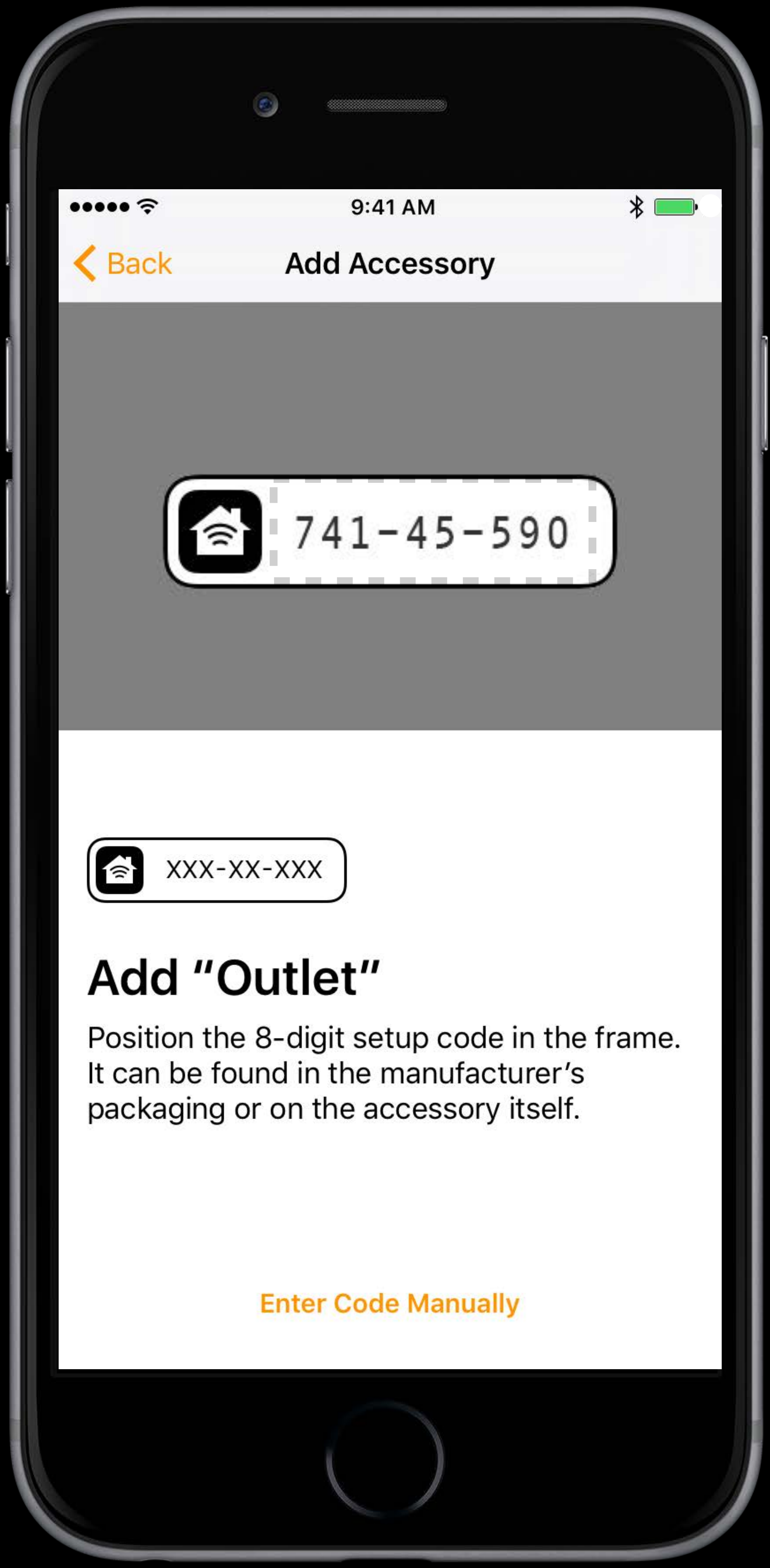

9:41 AM

Cancel Add Accessory

Searching for Accessories...

Make sure your accessory is powered on and nearby. Then tap the accessory you would like to add to your home.

My accessory isn't shown here.

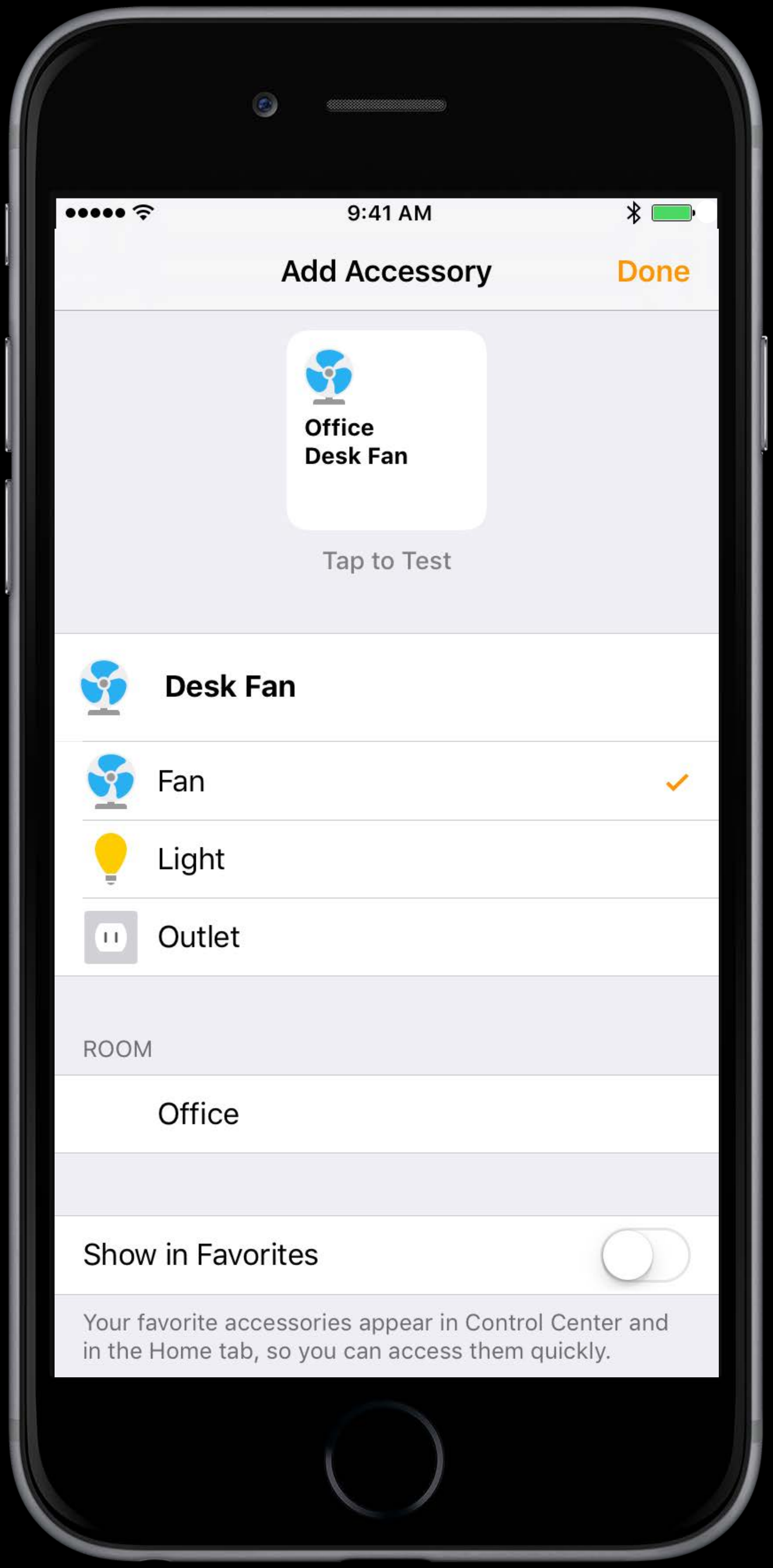

9:41 AM

Back Add Accessory

Add "Outlet"

Position the 8-digit setup code in the frame. It can be found in the manufacturer's packaging or on the accessory itself.

Enter Code Manually


9:41 AM

Add Accessory Done

Tap to Test

Desk Fan

Fan ✓

Light

Outlet

ROOM

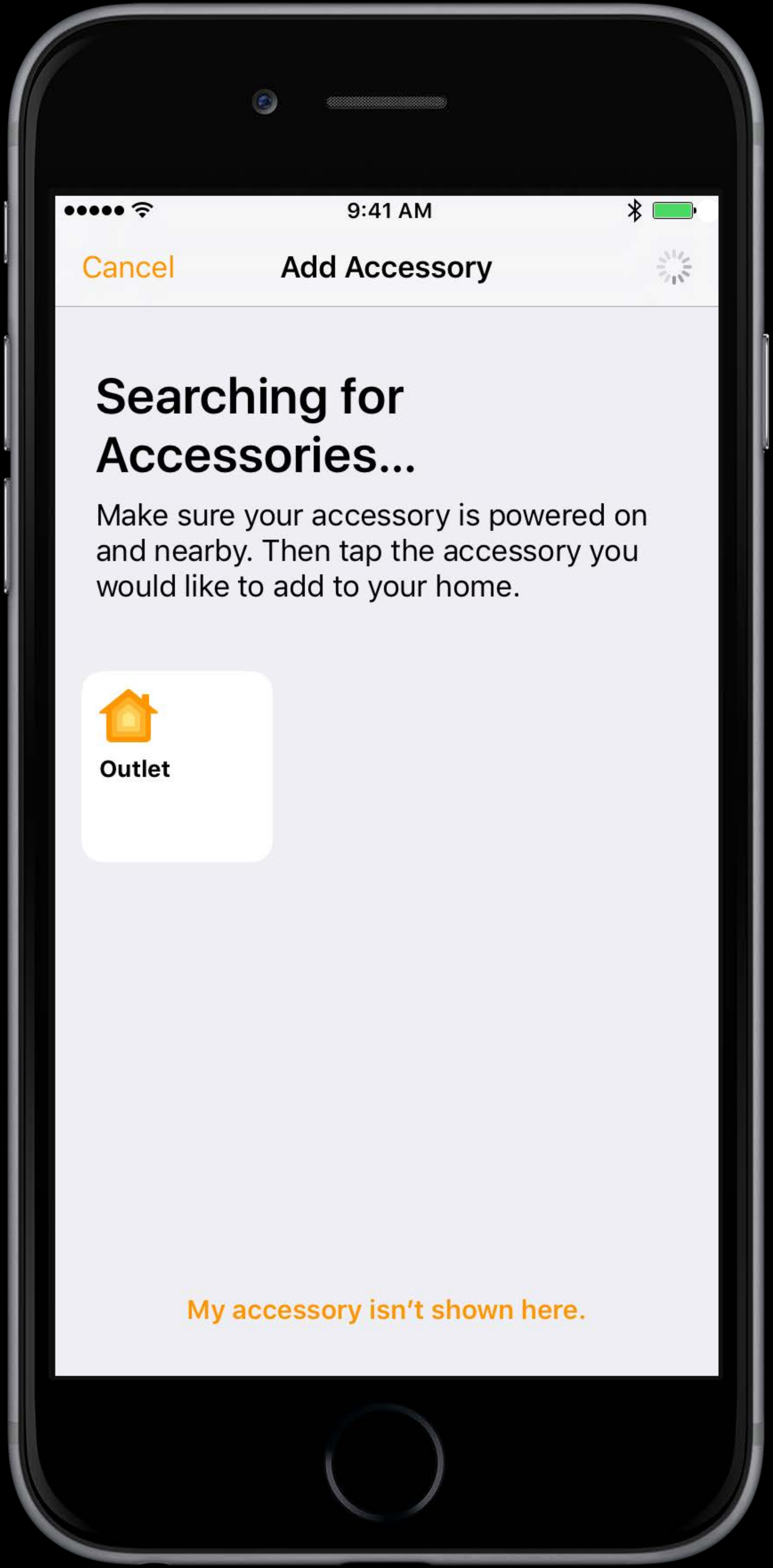
Office

Show in Favorites

Your favorite accessories appear in Control Center and in the Home tab, so you can access them quickly.

Setting Up HomeKit Accessories

NEW

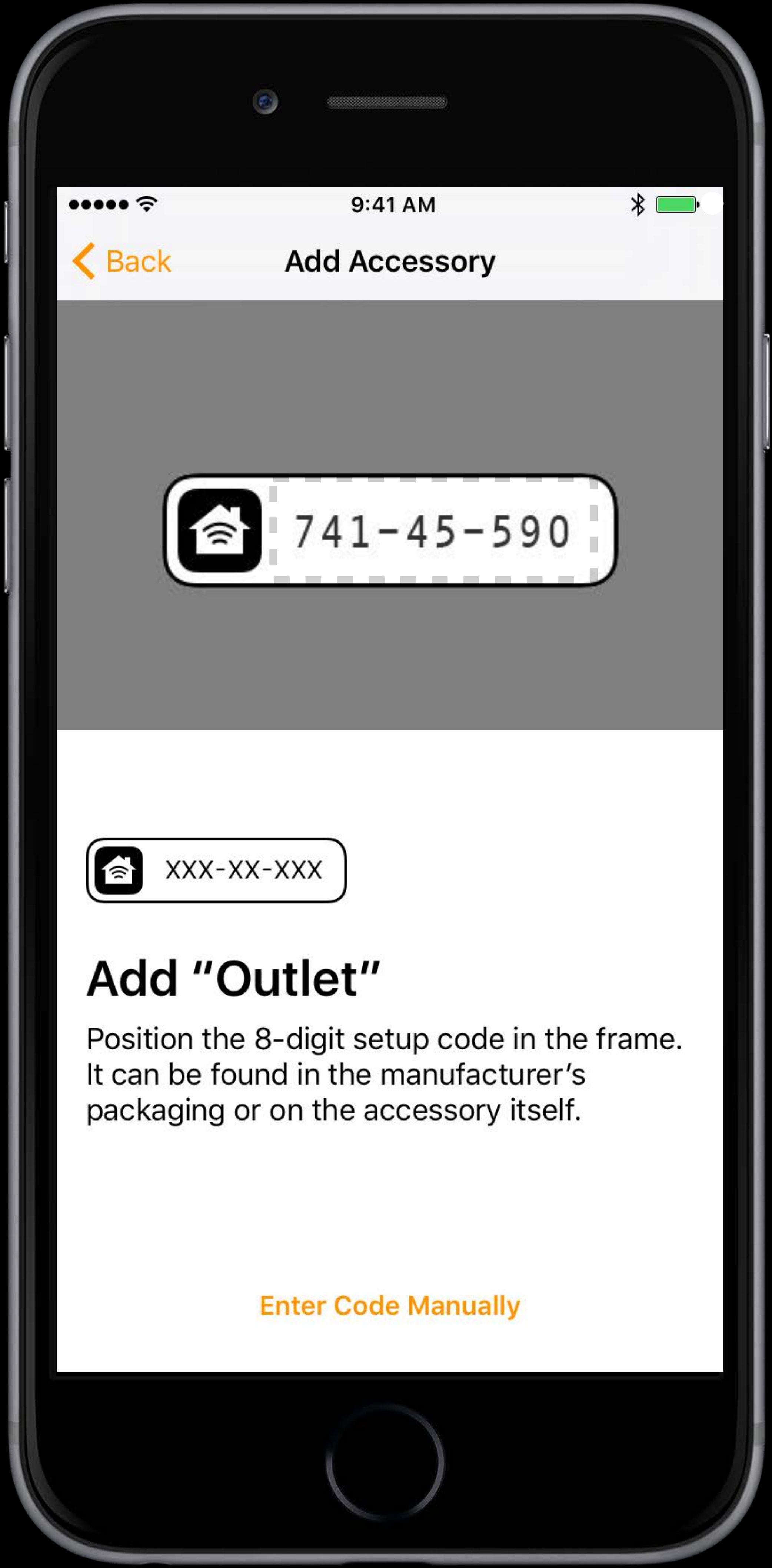

Cancel Add Accessory

Searching for Accessories...

Make sure your accessory is powered on and nearby. Then tap the accessory you would like to add to your home.

Outlet

My accessory isn't shown here.


Back Add Accessory

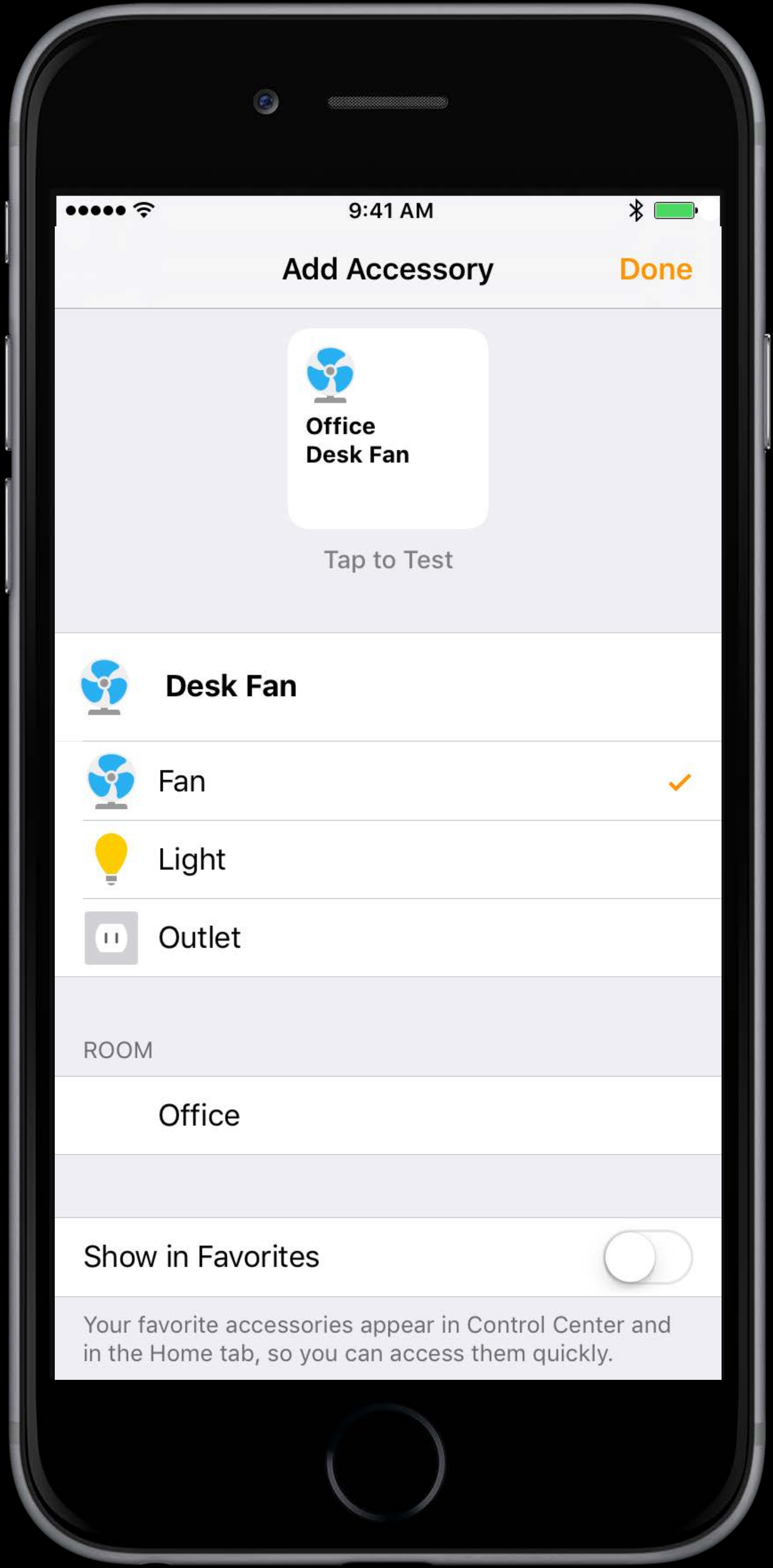
741-45-590

XXX-XX-XXX

Add "Outlet"

Position the 8-digit setup code in the frame. It can be found in the manufacturer's packaging or on the accessory itself.

Enter Code Manually


Add Accessory Done

Office Desk Fan

Tap to Test

Desk Fan

Fan ✓

Light

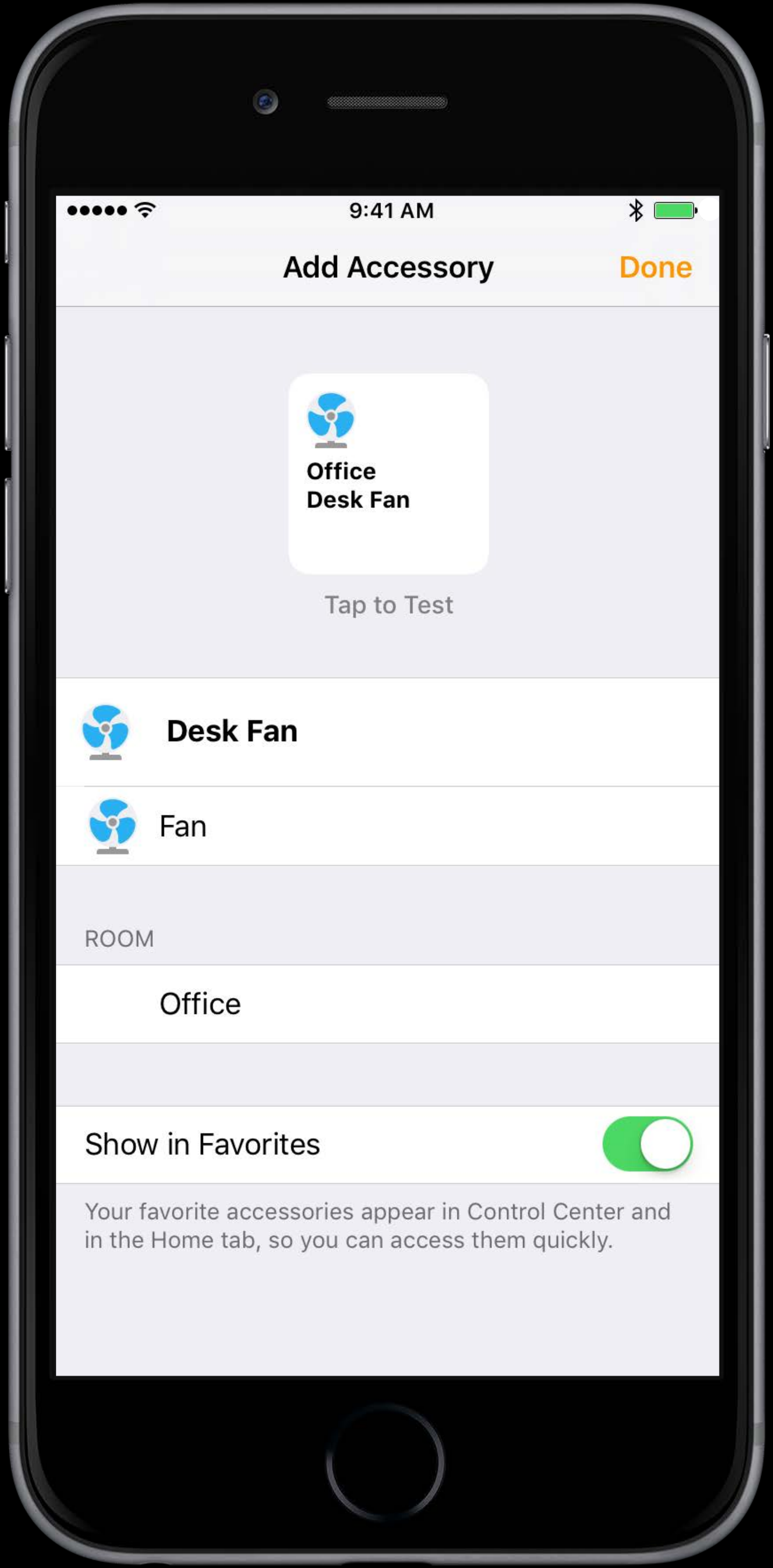
Outlet

ROOM

Office

Show in Favorites

Your favorite accessories appear in Control Center and in the Home tab, so you can access them quickly.


Add Accessory Done

Office Desk Fan

Tap to Test

Desk Fan

Fan

ROOM

Office

Show in Favorites

Your favorite accessories appear in Control Center and in the Home tab, so you can access them quickly.

Camera Accessories

New Framework APIs

HMCameraProfile

NEW

Getting the camera profile

```
extension HMAccessory {

    public var cameraProfiles: [HMCameraProfile]? { get }
}
```

HMCameraProfile

NEW

Getting the camera profile

```
extension HMAccessory {

    public var cameraProfiles: [HMCameraProfile]? { get }
}
```

HMCameraProfile

NEW

HMCameraProfile

NEW

Interfaces to interact with a camera

HMCameraProfile

NEW

Interfaces to interact with a camera

- Display and control the camera stream

HMCameraProfile

NEW

Interfaces to interact with a camera

- Display and control the camera stream
- Display still images

HMCameraProfile

NEW

Interfaces to interact with a camera

- Display and control the camera stream
- Display still images
- Control camera settings

HMCameraProfile

NEW

Interfaces to interact with a camera

- Display and control the camera stream
- Display still images
- Control camera settings
- Control speaker and microphone on the camera

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {

   public var streamControl: HMCameraStreamControl? { get }

...




}
```

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {

    public var streamControl: HMCameraStreamControl? { get }

...



}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

```
@objc public protocol HMCameraStreamControlDelegate : NSObjectProtocol {

   @objc optional func cameraStreamControlDidStartStream( ... )

   @objc optional func cameraStreamControl( ... didStopStreamWithError error: ... )

}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

```
@objc public protocol HMCameraStreamControlDelegate : NSObjectProtocol {

   @objc optional func cameraStreamControlDidStartStream( ... )

   @objc optional func cameraStreamControl( ... didStopStreamWithError error: ... )

}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

```
@objc public protocol HMCameraStreamControlDelegate : NSObjectProtocol {

   @objc optional func cameraStreamControlDidStartStream( ... )

   @objc optional func cameraStreamControl( ... didStopStreamWithError error: ... )

}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

```
@objc public protocol HMCameraStreamControlDelegate : NSObjectProtocol {

   @objc optional func cameraStreamControlDidStartStream( ... )

   @objc optional func cameraStreamControl( ... didStopStreamWithError error: ... )

}
```

HMCameraStreamControl

NEW

Controls the camera stream

```
public class HMCameraStreamControl : HMCameraControl {

...
   public func startStream()

   public func stopStream()

   public var cameraStream: HMCameraStream? { get }
}
```

```
@objc public protocol HMCameraStreamControlDelegate : NSObjectProtocol {

   @objc optional func cameraStreamControlDidStartStream( ... )

   @objc optional func cameraStreamControl( ... didStopStreamWithError error: ... )

}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

   public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

    public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

    public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

```
public enum HMCameraAudioStreamSetting : UInt {

    case muted

    case incomingAudioAllowed

    case bidirectionalAudioAllowed
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

   public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

```
public enum HMCameraAudioStreamSetting : UInt {

   case muted

   case incomingAudioAllowed

   case bidirectionalAudioAllowed
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

   public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

```
public enum HMCameraAudioStreamSetting : UInt {

   case muted

   case incomingAudioAllowed

   case bidirectionalAudioAllowed
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

   public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

```
public enum HMCameraAudioStreamSetting : UInt {

   case muted

   case incomingAudioAllowed

   case bidirectionalAudioAllowed
}
```

HMCameraStream

NEW

```
public class HMCameraStream : HMCameraSource {

    public var audioStreamSetting: HMCameraAudioStreamSetting
}
```

```
public enum HMCameraAudioStreamSetting : UInt {

    case muted

    case incomingAudioAllowed

    case bidirectionalAudioAllowed
}
```

HMCameraView

iOS 10, tvOS 10

NEW

View that can render a camera source

```
public class HMCameraView : UIView {

   public var cameraSource: HMCameraSource?

}
```

HMCameraView

iOS 10, tvOS 10

NEW

View that can render a camera source

```
public class HMCameraView : UIView {

   public var cameraSource: HMCameraSource?

}
```

WKInterfaceHMCamera

watchOS 3

NEW

View that can render a camera source

```
public class WKInterfaceHMCamera : WKInterfaceObject {

    public func setCameraSource(_ cameraSource: HMCameraSource?)

}
```

WKInterfaceHMCamera

watchOS 3

NEW

View that can render a camera source

```
public class WKInterfaceHMCamera : WKInterfaceObject {

    public func setCameraSource(_ cameraSource: HMCameraSource?)

}
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.streamControl?.startStream()
   }

   let liveStreamView = HMCameraView()

   // MARK: HMCameraStreamControlDelegate

   func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
      liveStreamView.cameraSource = cameraStreamControl.cameraStream
   }
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

    let cameraAccessory : HMAccessory = ...
    if let cameraProfile = cameraAccessory.cameraProfiles?.first {
        cameraProfile.streamControl?.startStream()
    }

    let liveStreamView = HMCameraView()

    // MARK: HMCameraStreamControlDelegate

    func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
        liveStreamView.cameraSource = cameraStreamControl.cameraStream
    }
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

    let cameraAccessory : HMAccessory = ...
    if let cameraProfile = cameraAccessory.cameraProfiles?.first {
        cameraProfile.streamControl?.startStream()
    }

    let liveStreamView = HMCameraView()

    // MARK: HMCameraStreamControlDelegate

    func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
        liveStreamView.cameraSource = cameraStreamControl.cameraStream
    }
```

```
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.streamControl?.startStream()
   }

   let liveStreamView = HMCameraView()

   // MARK: HMCameraStreamControlDelegate

   func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
      liveStreamView.cameraSource = cameraStreamControl.cameraStream
   }
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.streamControl?.startStream()
   }

   let liveStreamView = HMCameraView()

   // MARK: HMCameraStreamControlDelegate

   func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
      liveStreamView.cameraSource = cameraStreamControl.cameraStream
   }
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.streamControl?.startStream()
   }

   let liveStreamView = HMCameraView()

   // MARK: HMCameraStreamControlDelegate

   func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
      liveStreamView.cameraSource = cameraStreamControl.cameraStream
   }
```

```swift
// Example - Start and display a live stream from a camera accessory

import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.streamControl?.startStream()
   }

   let liveStreamView = HMCameraView()

   // MARK: HMCameraStreamControlDelegate

   func cameraStreamControlDidStartStream(_ cameraStreamControl: HMCameraStreamControl) {
      liveStreamView.cameraSource = cameraStreamControl.cameraStream
   }
```

HMCameraProfile

NEW

Interfaces to interact with a Camera

```
public class HMCameraProfile : HMAccessoryProfile {
   public var streamControl: HMCameraStreamControl? { get }
   public var snapshotControl: HMCameraSnapshotControl? { get }
..


}
```

HMCameraProfile

NEW

Interfaces to interact with a Camera

```
public class HMCameraProfile : HMAccessoryProfile {

   public var streamControl: HMCameraStreamControl? { get }

   public var snapshotControl: HMCameraSnapshotControl? { get }

..


}
```

HMCameraSnapshotControl

NEW

Capture a still image

```
public class HMCameraSnapshotControl : HMCameraControl {

...

   public func takeSnapshot()

   public var mostRecentSnapshot: HMCameraSnapshot? { get }
}
```

HMCameraSnapshotControl

NEW

Capture a still image

```
public class HMCameraSnapshotControl : HMCameraControl {

...

   public func takeSnapshot()

   public var mostRecentSnapshot: HMCameraSnapshot? { get }
}
```

HMCameraSnapshotControl

NEW

Capture a still image

```
public class HMCameraSnapshotControl : HMCameraControl {

...

   public func takeSnapshot()

   public var mostRecentSnapshot: HMCameraSnapshot? { get }
}
```

```
@objc public protocol HMCameraSnapshotControlDelegate : NSObjectProtocol {

   @objc optional func cameraSnapshotControl( ... didTake snapshot: ...)
}
```

HMCameraSnapshotControl

NEW

Capture a still image

```
public class HMCameraSnapshotControl : HMCameraControl {

...

   public func takeSnapshot()

   public var mostRecentSnapshot: HMCameraSnapshot? { get }
}
```

```
@objc public protocol HMCameraSnapshotControlDelegate : NSObjectProtocol {

   @objc optional func cameraSnapshotControl( ... didTake snapshot: ...)
}
```

HMCameraSnapshotControl

NEW

Capture a still image

```
public class HMCameraSnapshotControl : HMCameraControl {

...

   public func takeSnapshot()

   public var mostRecentSnapshot: HMCameraSnapshot? { get }
}
```

```
@objc public protocol HMCameraSnapshotControlDelegate : NSObjectProtocol {

   @objc optional func cameraSnapshotControl( ... didTake snapshot: ...)
}
```

HMCameraSnapshot

NEW

```
public class HMCameraSnapshot : HMCameraSource {

    public var captureDate: Date { get }
}
```

HMCameraSnapshot

NEW

```
public class HMCameraSnapshot : HMCameraSource {

    public var captureDate: Date { get }
}
```

HMCameraSnapshot

NEW

```
public class HMCameraSnapshot : HMCameraSource {

    public var captureDate: Date { get }
}
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.snapshotControl?.takeSnapshot()
   }
   let snapshotView = HMCameraView()

   // MARK: HMCameraSnapshotControlDelegate

   func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
      didTake snapshot: HMCameraSnapshot?, error: NSError?) {
      if error == nil {
         snapshotView.cameraSource = snapshot
      } else {
         // Error handling
      }
   }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.snapshotControl?.takeSnapshot()
   }
   let snapshotView = HMCameraView()

   // MARK: HMCameraSnapshotControlDelegate

   func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
      didTake snapshot: HMCameraSnapshot?, error: NSError?) {
      if error == nil {
         snapshotView.cameraSource = snapshot
      } else {
         // Error handling
      }
   }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

    let cameraAccessory : HMAccessory = ...
    if let cameraProfile = cameraAccessory.cameraProfiles?.first {
        cameraProfile.snapshotControl?.takeSnapshot()
    }
    let snapshotView = HMCameraView()

    // MARK: HMCameraSnapshotControlDelegate

    func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
        didTake snapshot: HMCameraSnapshot?, error: NSError?) {
        if error == nil {
            snapshotView.cameraSource = snapshot
        } else {
            // Error handling
        }
    }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

    let cameraAccessory : HMAccessory = ...
    if let cameraProfile = cameraAccessory.cameraProfiles?.first {
       cameraProfile.snapshotControl?.takeSnapshot()
    }
    let snapshotView = HMCameraView()

    // MARK: HMCameraSnapshotControlDelegate

    func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
       didTake snapshot: HMCameraSnapshot?, error: NSError?) {
       if error == nil {
          snapshotView.cameraSource = snapshot
       } else {
          // Error handling
       }
    }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.snapshotControl?.takeSnapshot()
   }
   let snapshotView = HMCameraView()

   // MARK: HMCameraSnapshotControlDelegate

   func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
     didTake snapshot: HMCameraSnapshot?, error: NSError?) {
      if error == nil {
         snapshotView.cameraSource = snapshot
      } else {
         // Error handling
      }
   }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.snapshotControl?.takeSnapshot()
   }
   let snapshotView = HMCameraView()

   // MARK: HMCameraSnapshotControlDelegate

   func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
      didTake snapshot: HMCameraSnapshot?, error: NSError?) {
      if error == nil {
         snapshotView.cameraSource = snapshot
      } else {
         // Error handling
      }
   }
```

```swift
// Example - Take and display snapshot from the camera accessory
import HomeKit
...

   let cameraAccessory : HMAccessory = ...
   if let cameraProfile = cameraAccessory.cameraProfiles?.first {
      cameraProfile.snapshotControl?.takeSnapshot()
   }
   let snapshotView = HMCameraView()

   // MARK: HMCameraSnapshotControlDelegate

   func cameraSnapshotControl(_ cameraSnapshotControl: HMCameraSnapshotControl,
      didTake snapshot: HMCameraSnapshot?, error: NSError?) {
      if error == nil {
         snapshotView.cameraSource = snapshot
      } else {
         // Error handling
      }
   }
```

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {
   public var streamControl: HMCameraStreamControl? { get }
   public var snapshotControl: HMCameraSnapshotControl? { get }
   public var settingsControl: HMCameraSettingsControl? { get }
...

}
```

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {
   public var streamControl: HMCameraStreamControl? { get }
   public var snapshotControl: HMCameraSnapshotControl? { get }
   public var settingsControl: HMCameraSettingsControl? { get }
...

}
```

HMCameraSettingsControl

NEW

HMCameraSettingsControl

NEW

Control the settings on the camera

HMCameraSettingsControl

NEW

Control the settings on the camera

- Night Vision

HMCameraSettingsControl

NEW

Control the settings on the camera

- Night Vision
- Tilt

HMCameraSettingsControl

NEW

Control the settings on the camera

- Night Vision
- Tilt
- Zoom

HMCameraSettingsControl

NEW

Control the settings on the camera

- Night Vision
- Tilt
- Zoom
- Rotation

HMCameraSettingsControl

NEW

Control the settings on the camera

- Night Vision
- Tilt
- Zoom
- Rotation
- Mirroring

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {

   public var streamControl: HMCameraStreamControl? { get }

   public var snapshotControl: HMCameraSnapshotControl? { get }

   public var settingsControl: HMCameraSettingsControl? { get }

   public var speakerControl: HMCameraAudioControl? { get }

   public var microphoneControl: HMCameraAudioControl? { get }
}
```

HMCameraProfile

NEW

Interfaces to interact with a camera

```
public class HMCameraProfile : HMAccessoryProfile {
   public var streamControl: HMCameraStreamControl? { get }
   public var snapshotControl: HMCameraSnapshotControl? { get }
   public var settingsControl: HMCameraSettingsControl? { get }
   public var speakerControl: HMCameraAudioControl? { get }
   public var microphoneControl: HMCameraAudioControl? { get }
}
```

HMCameraAudioControl

NEW

HMCameraAudioControl

NEW

Controls the audio settings on the camera

HMCameraAudioControl

NEW

Controls the audio settings on the camera

- Change mute settings on the microphone

HMCameraAudioControl

NEW

Controls the audio settings on the camera

- Change mute settings on the microphone
- Change mute settings on the speaker

HMCameraAudioControl

NEW

Controls the audio settings on the camera

- Change mute settings on the microphone
- Change mute settings on the speaker
- Control the microphone gain

HMCameraAudioControl

NEW

Controls the audio settings on the camera

- Change mute settings on the microphone
- Change mute settings on the speaker
- Control the microphone gain
- Control the speaker volume

Summary

Summary

Platform

Summary

Platform

- Home App

Summary

Platform

- Home App
- Remote Access and Automation

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Framework Updates

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Framework Updates

- Primary Service, Linked Services, Valid Values

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Framework Updates

- Primary Service, Linked Services, Valid Values
- Accessory Setup

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Framework Updates

- Primary Service, Linked Services, Valid Values
- Accessory Setup
- Camera Accessory Classes

Summary

Platform

- Home App
- Remote Access and Automation
- tvOS HomeKit Framework

New Accessories

- Air Treatment
- Cameras
- Doorbells

Framework Updates

- Primary Service, Linked Services, Valid Values
- Accessory Setup
- Camera Accessory Classes

Accessibility

MFi Program

https://developer.apple.com/mfi/

More Information

https://developer.apple.com/wwdc16/710

Related Sessions

Disability and Innovation: The Universal Benefits of Accessible Design	Presidio	Tuesday 12:20PM
Designing for tvOS	Presidio	Tuesday 4:00PM
Auditing Your Apps for Accessibility	Nob Hill	Wednesday, 10:00AM

Labs

tvOS Lab	Frameworks Lab D	Thursday 09:00AM
HomeKit Lab	Frameworks Lab C	Thursday 09:00AM
Accessories Lab	Frameworks Lab C	Friday 09:00AM
HomeKit Lab	Frameworks Lab A	Friday 11:30AM

WWDC16