What's New in HomeKit

Session 705

Matt Lucas, HomeKit Engineering
Praveen Chegondi, HomeKit Engineering

Overview

Overview

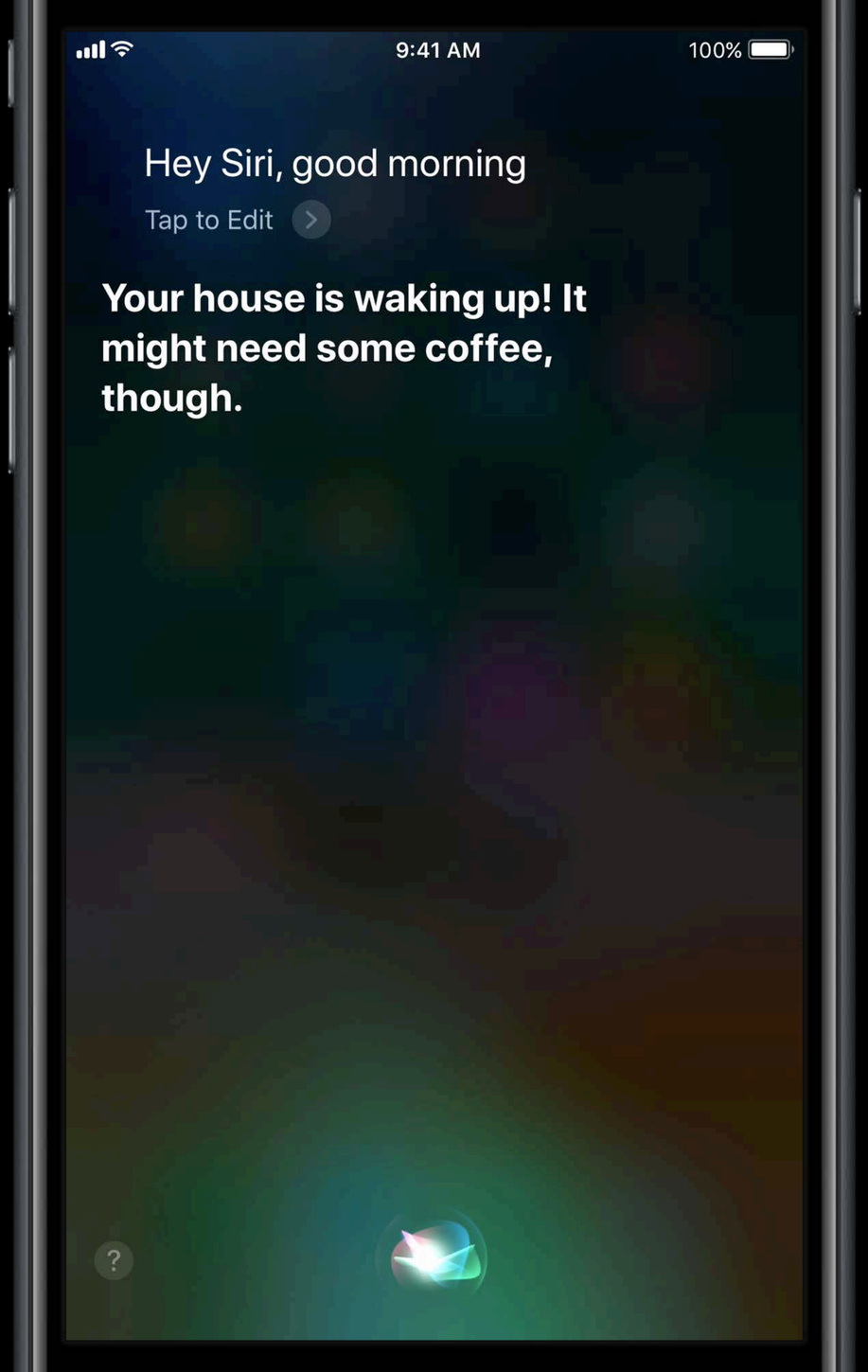
Framework updates

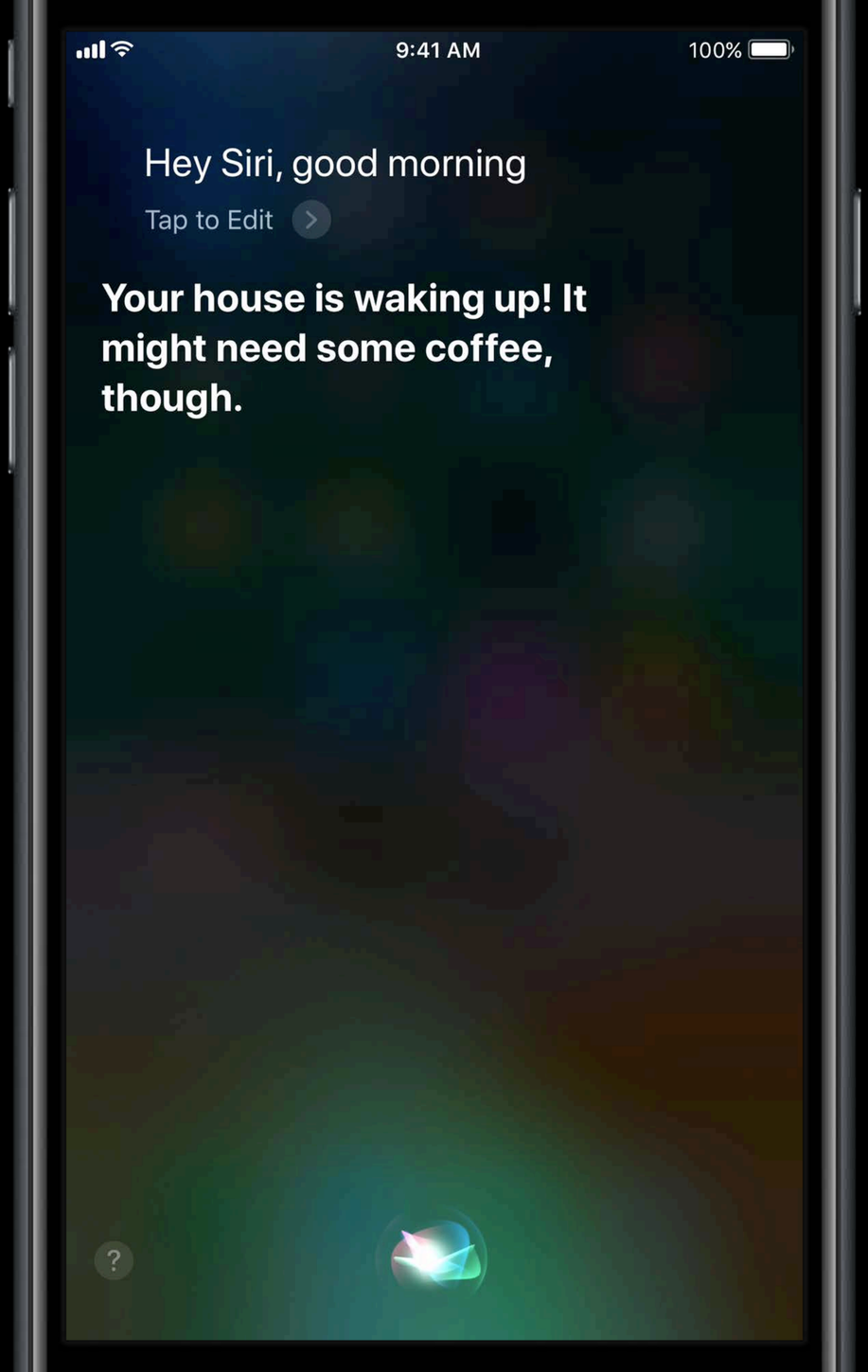
Overview

Framework updates

Accessory updates

Overview

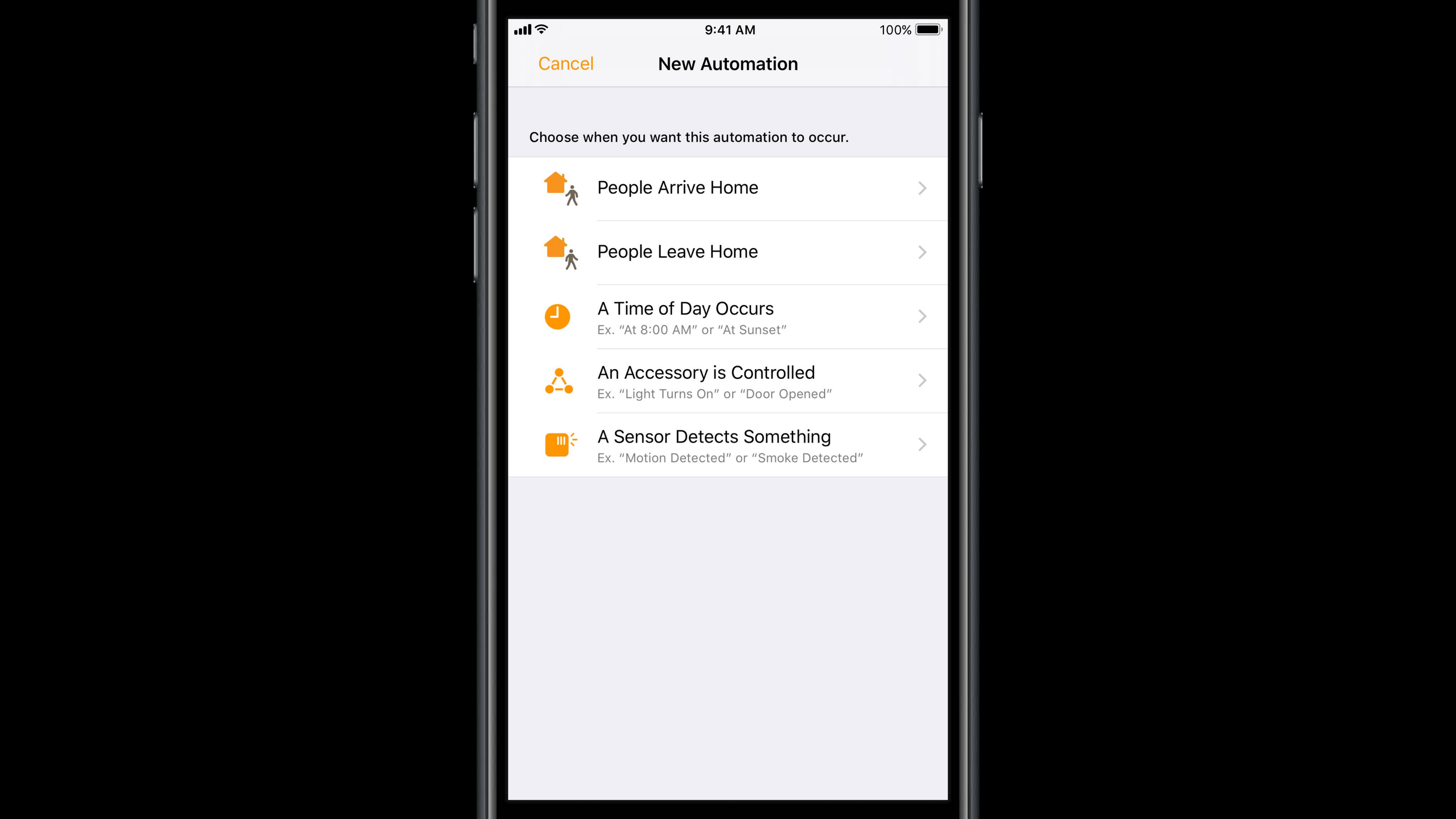

9:41 AM
100%
Hey Siri, good morning
Tap to Edit
Your house is waking up! It might need some coffee, though.


9:41 AM
100%
Hey Siri, good morning
Tap to Edit
Your house is waking up! It might need some coffee, though.


iPad
9:41 AM
100%
Edit
Ellsworth St
Front door unlocked.
Living Room blinds open.
4 lights on.
and 2 More ›
Favorite Scenes
I'm Home
Good Morning
Good Night
Favorite Accessories
68°
Living Room Thermostat
Heating to 72°
Living Room Shades
Open
Hallway Light
70%
Front Door
Unlocked
Dining Room Light
70%
Garage Door
Closed
Living Room Smoke Dete...
Home
Rooms
Automation
9:41 AM
100%
Edit
Ellsworth St
Front door unlocked.
Living Room blinds open.
4 lights on.
and 4 More ›
Favorite Scenes
I'm Home
Good Night
Favorite Accessories
68°
Living Room Thermostat
Heating to 72°
Living Room Shades
Open
Hallway Light
70%
Home
Rooms
Automation
Ellsworth St
9:41
I'm Home
Good Night


9:41 AM
100%
Mail
Saturday
3
Calendar
Photos
Camera
280
Maps
Clock
Weather
News
Home
Notes
Stocks
Reminders
TV
App Store
iTunes Store
iBooks
Health
Wallet
Settings

Remote Access


9:41 AM
100%
Cancel
New Automation
Choose when you want this automation to occur.
People Arrive Home
People Leave Home
A Time of Day Occurs
Ex. “At 8:00 AM” or “At Sunset”
An Accessory is Controlled
Ex. “Light Turns On” or “Door Opened”
A Sensor Detects Something
Ex. “Motion Detected” or “Smoke Detected”


2GIG
艾美特
arlo
august
wemo
bîrdi
CHAMBERLAIN
ConnectSense
COWAY
D-Link
DeLonghi
BIG ASS SOLUTIONS
DONLIM
DOOYA
TUBULAR MOTOR
DOORLINK
东软载波
Eastsoft
ecobee
elgato
enno
First Alert
FIBARO
HOME INTELLIGENCE
Onelink
FLEDA
FOTILE 方太
FRIDAY
GE
GENIE
Haier
heatmiser
Hisense
Honeywell
Hunter
HunterDouglas
iDevices
iHome
IKEA
ilumi
iLuv
INCIPIO
INSIGNIA
kaiterra
Kidde
koogeek
Kwikset
LEDVANCE
legrand
LENNOX
LEVITON
LiftMaster
LIFX
Lightwave RF
logitech
LUTRON
moorgen
nanoleaf
NETATMO
Nice
NORTEK
SECURITY & CONTROL
OPPLE
欧普照明
Oprog
OPSO
Orbit
b·hyve
parce
Pella
PHILIPS
hue
PROMEMORIA
PURE GEAR
rachio
ring
sen|si
SCHLAGE
Schneider
Electric
SHANUTEC
SYLVANIA
tado°
TOP
突破
VELUX
Yale

Framework Updates

Event Triggers

Event Triggers

New Events

Event Triggers

New Events

New Conditions

Event Triggers

New Events

New Conditions

End Events

Event Triggers

New Events

New Conditions

End Events

Recurrence

Event Triggers

New Events

New Conditions

End Events

Recurrence

Mutable Events

Overview

Event Triggers

Events

Event Triggers

Events

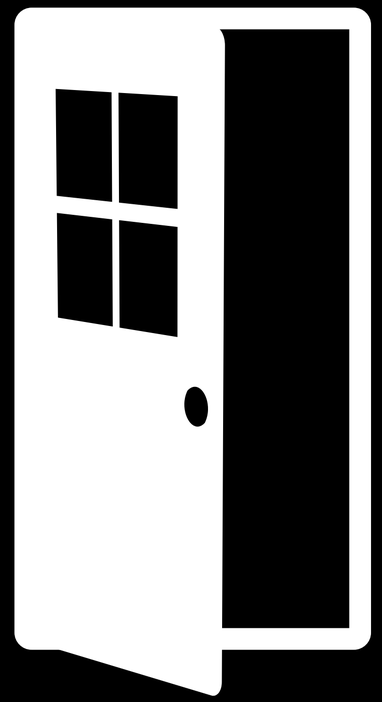
Events activate a trigger

Event Triggers

Events

Events activate a trigger

- State of an accessory

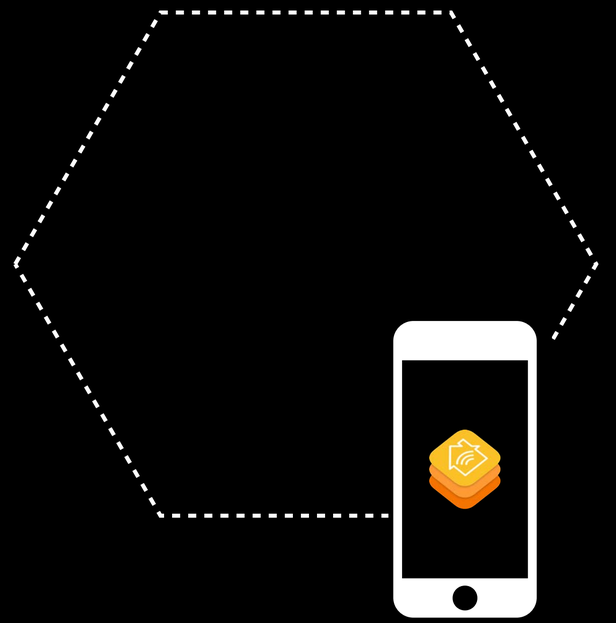


Event Triggers

Events

Events activate a trigger

- State of an accessory
- Geofence

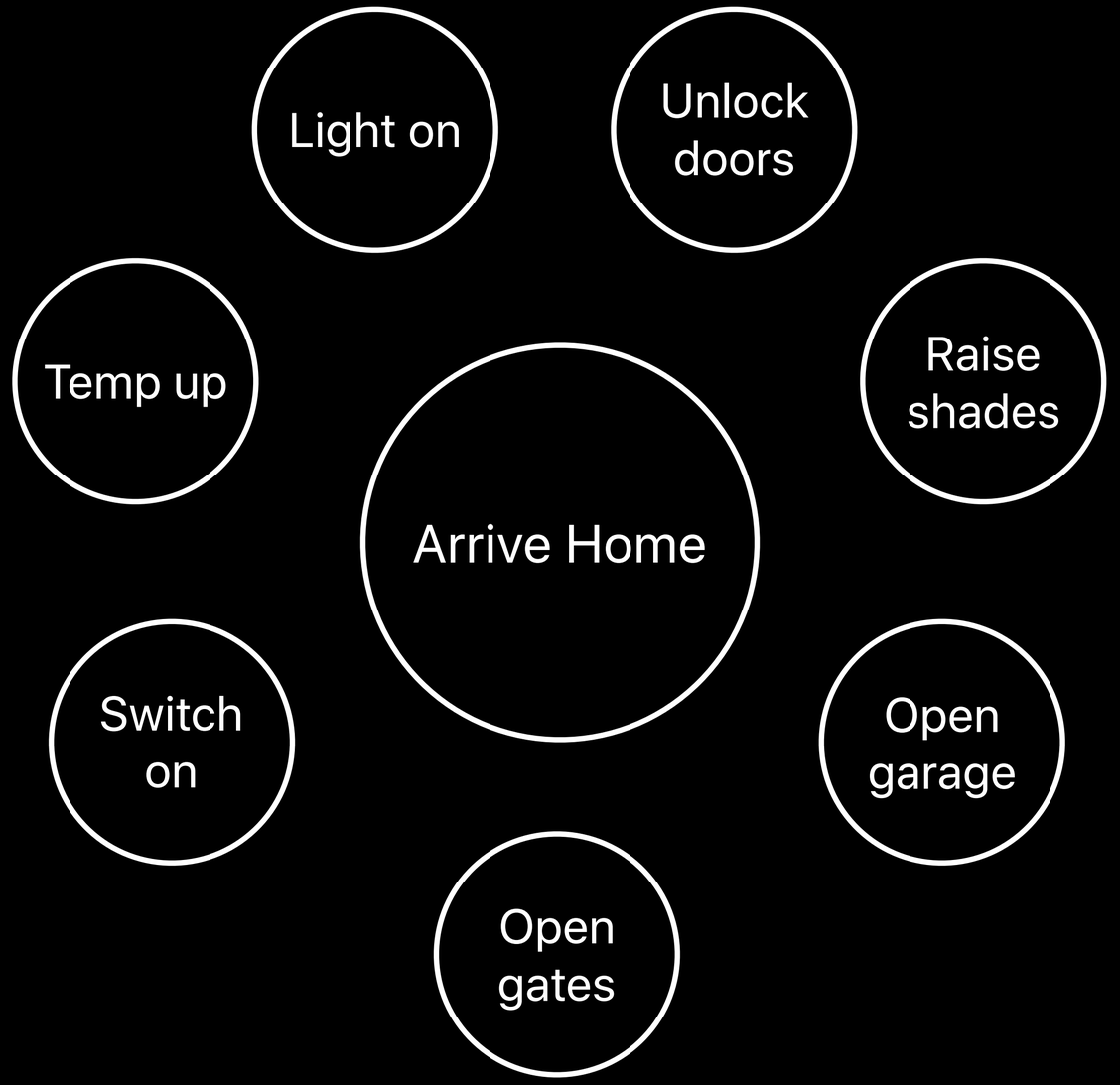


Event Triggers

Scenes

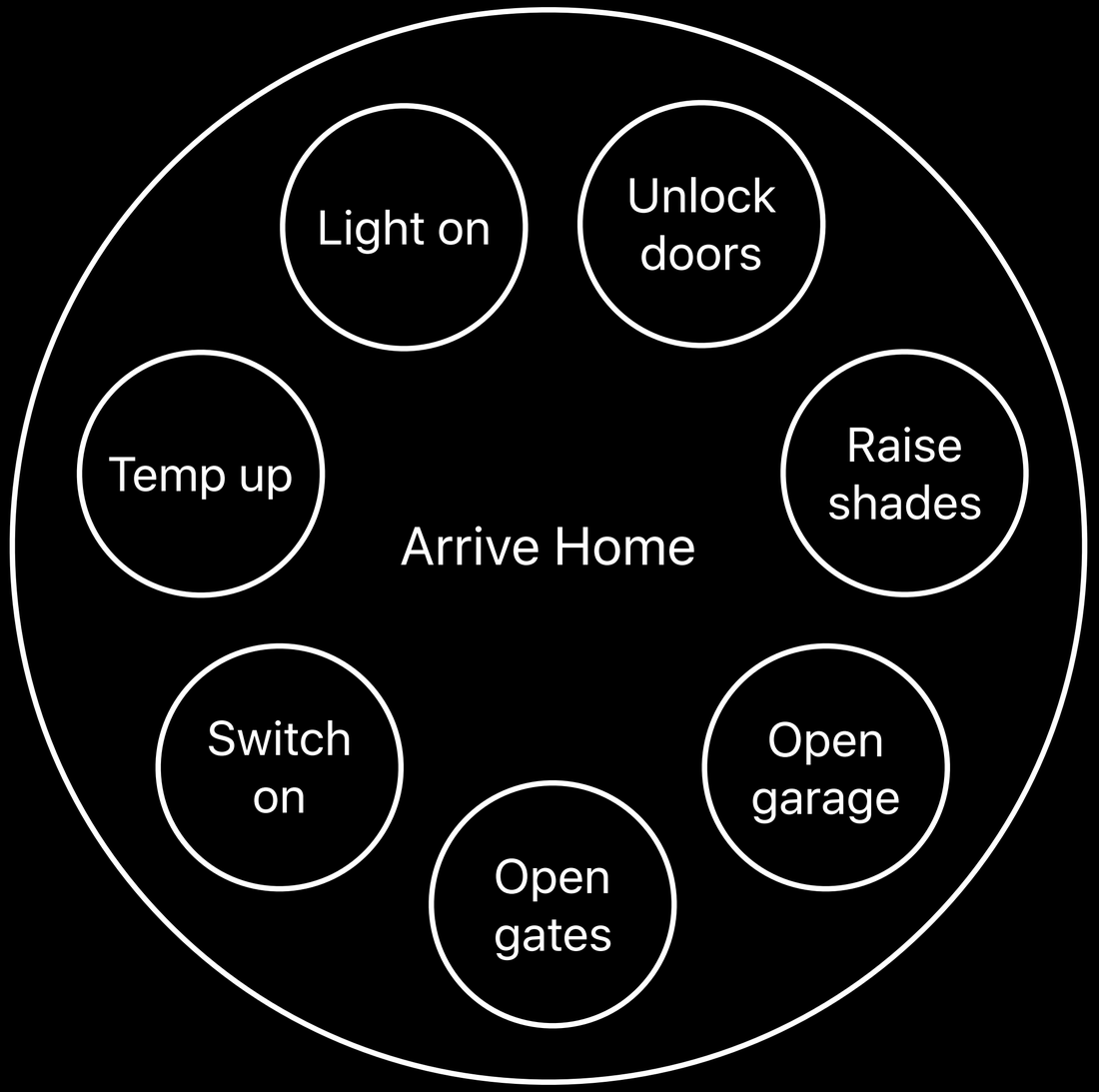
Event Triggers

Scenes



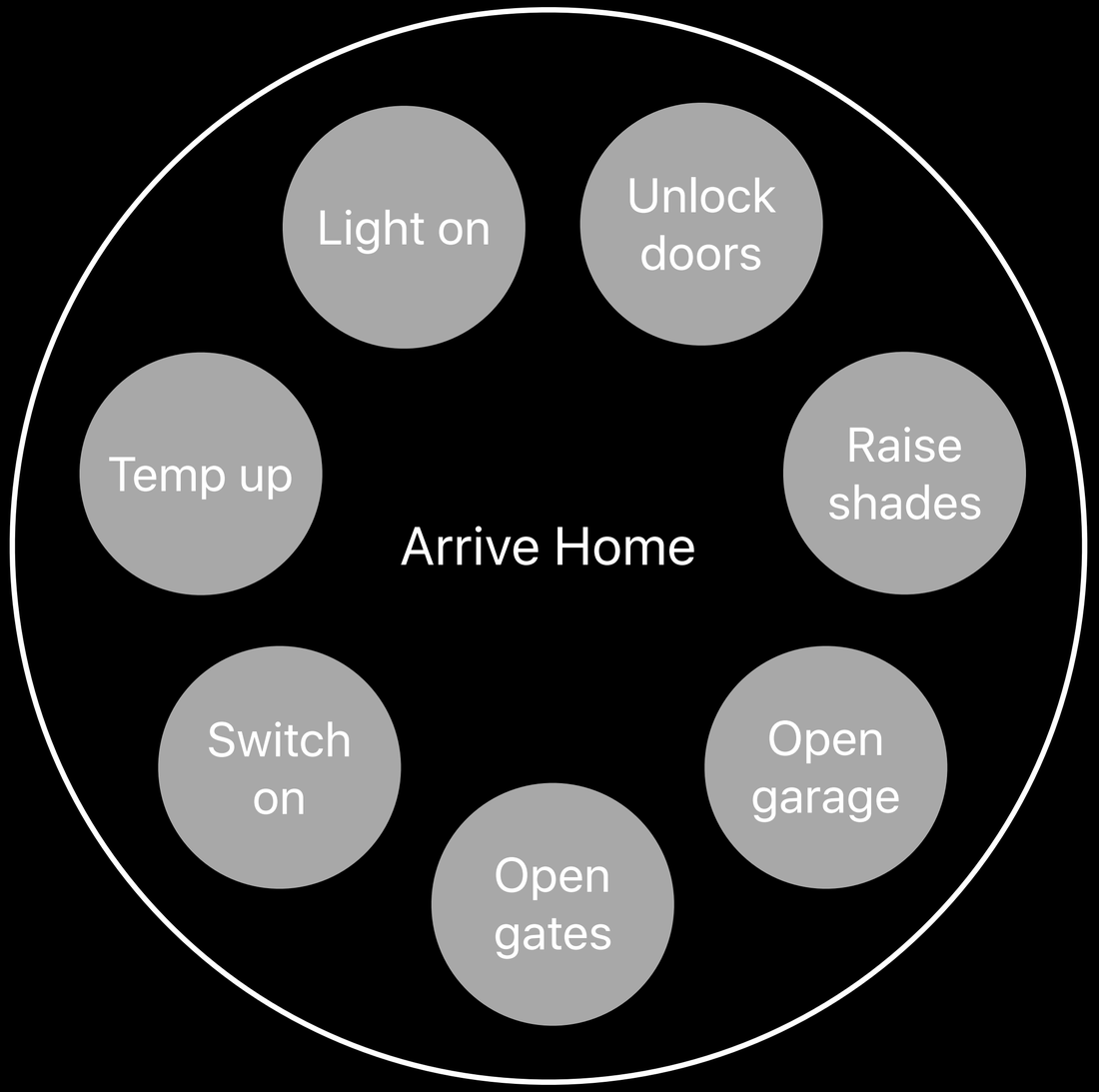
Event Triggers

Scenes



Event Triggers

Scenes



Event Triggers

Conditions

Event Triggers

Conditions

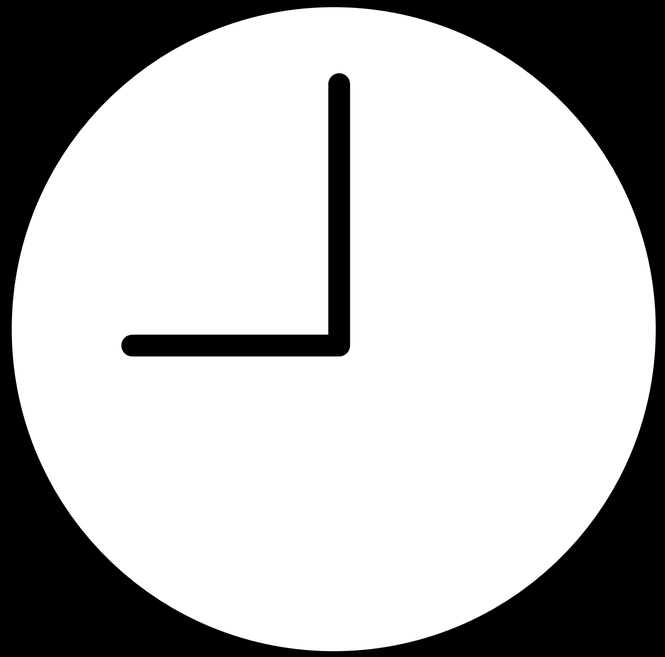
Gates execution of scenes

Event Triggers

Conditions

Gates execution of scenes

- Time-based

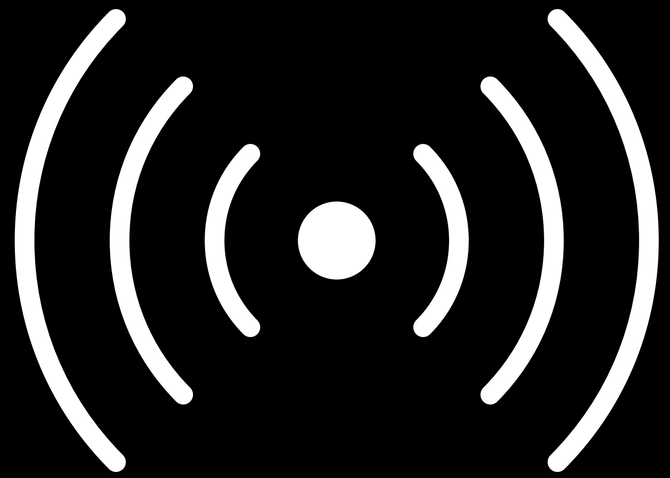


Event Triggers

Conditions

Gates execution of scenes

- Time-based
- State of an accessory



Event Triggers

Conditions

Gates execution of scenes

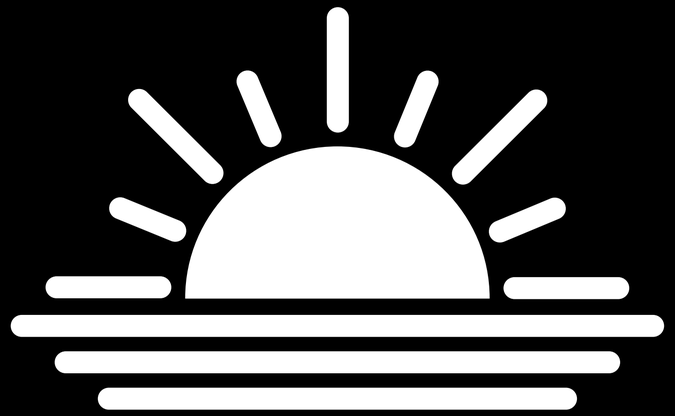
- Time-based
- State of an accessory
- Significant events

Event Triggers

Conditions

Gates execution of scenes

- Time-based
- State of an accessory
- Significant events
  - Sunrise

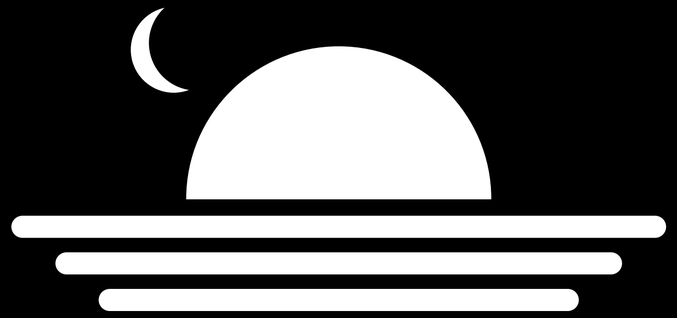


Event Triggers

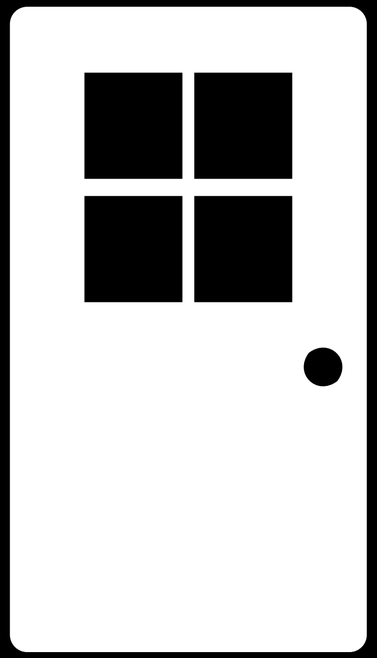
Conditions

Gates execution of scenes

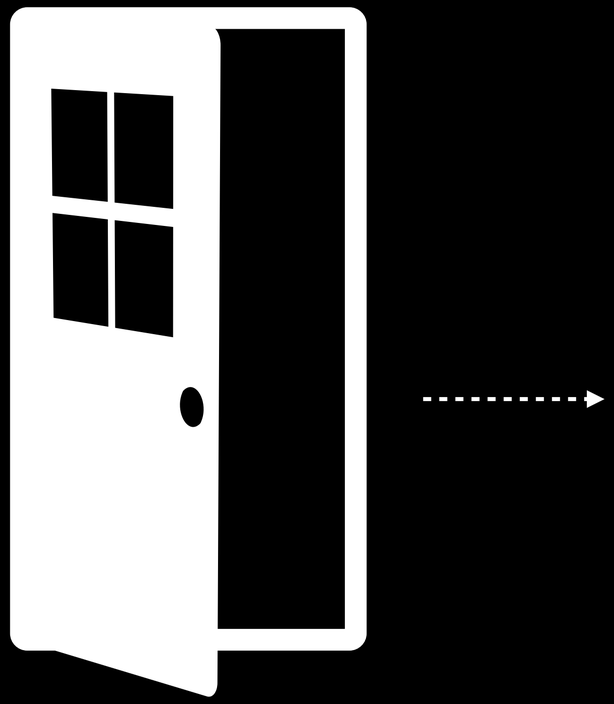
- Time-based
- State of an accessory
- Significant events
  - Sunrise
  - Sunset



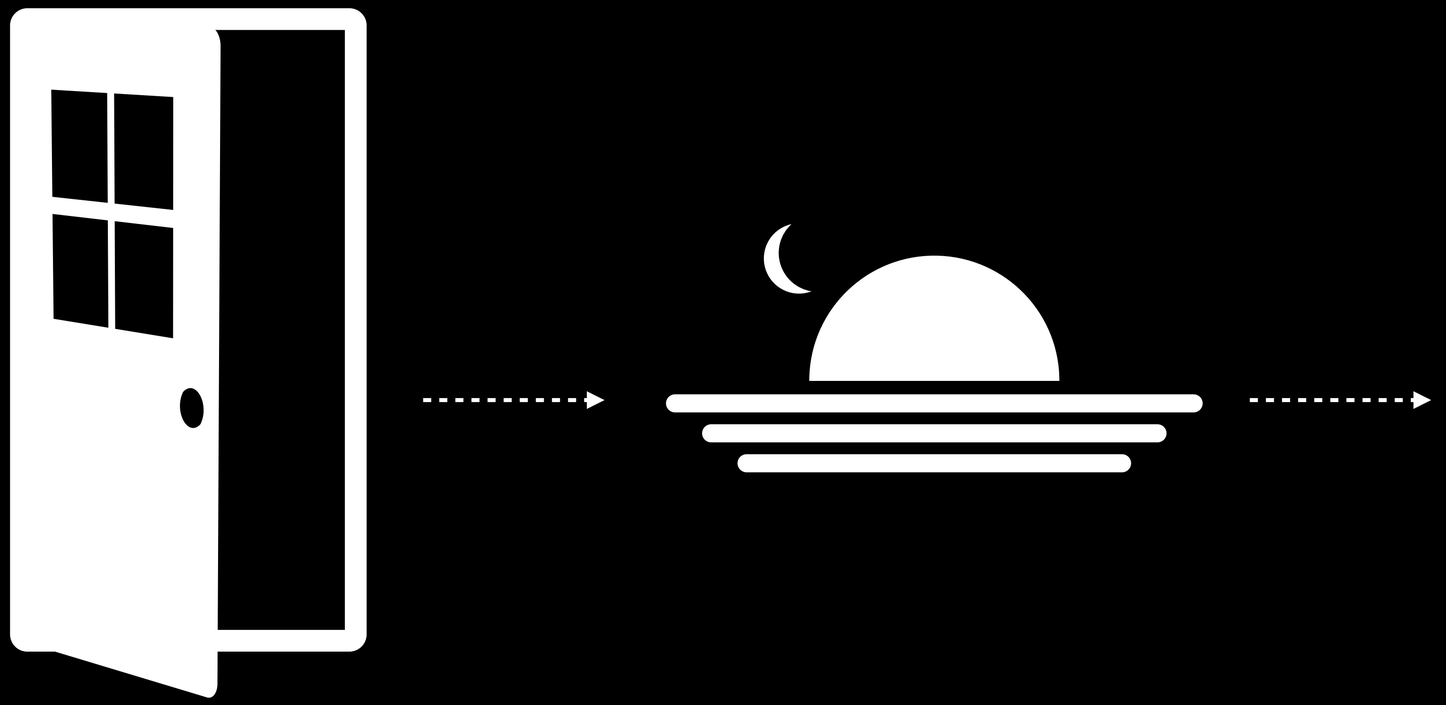
Event Triggers



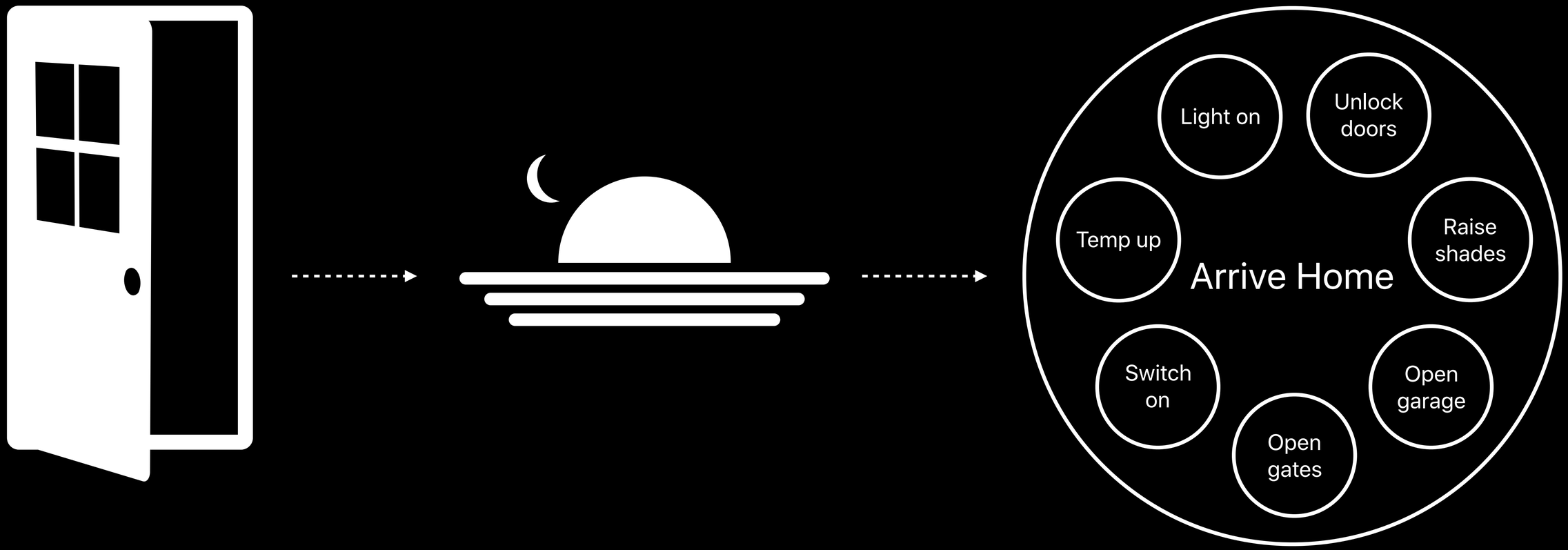
Event Triggers



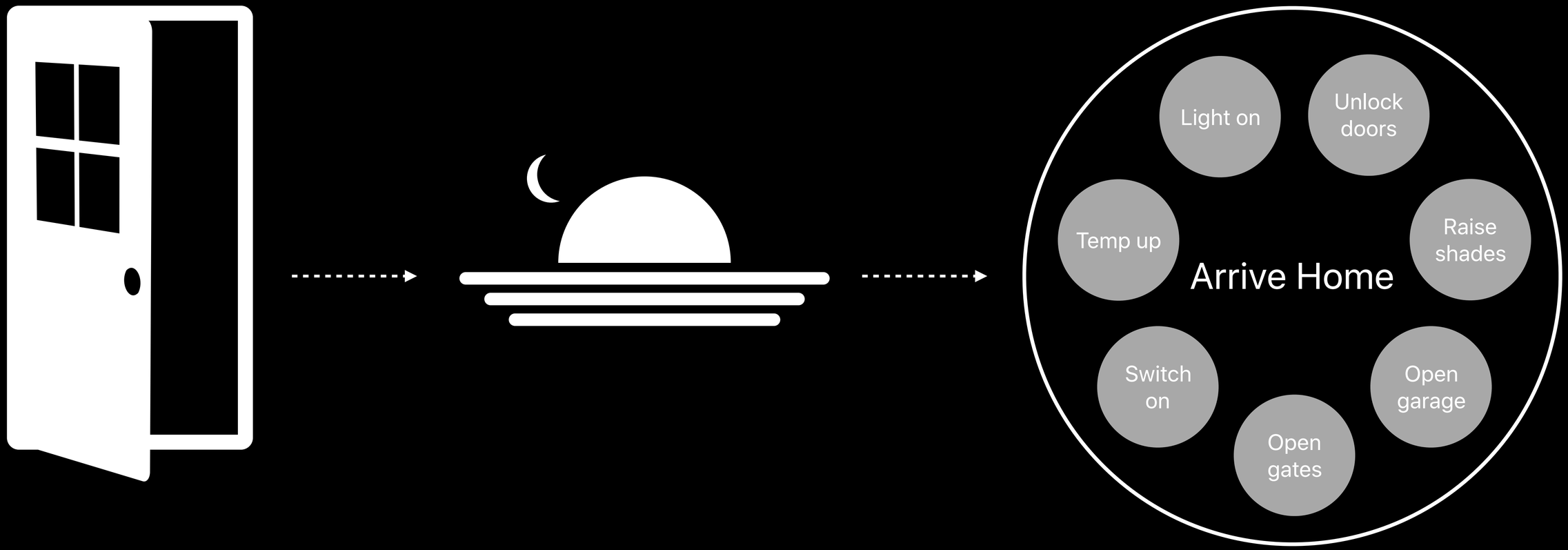
Event Triggers



Event Triggers



Event Triggers



Event Triggers

Event Triggers

HMCalendarEvent

NEW

Event Triggers

HMCalendarEvent

NEW

Supports absolute dates, including:

- Year
- Month
- Day
- Hour Minute

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

```
//  Example - Creating calendar time event

import HomeKit

var dateComponents = DateComponents()
dateComponents.hour = 17
dateComponents.minute = 30

let calendarEvent = HMCalendarEvent(fire: dateComponents)
let eventTrigger = HMEventTrigger(name: “Every day at 5:30PM”,
                                  events: [calendarEvent],
                                  predicate: nil)
```

Event Triggers

HMSignificantTimeEvent

NEW

Event Triggers

HMSignificantTimeEvent

NEW

Activates on significant events:

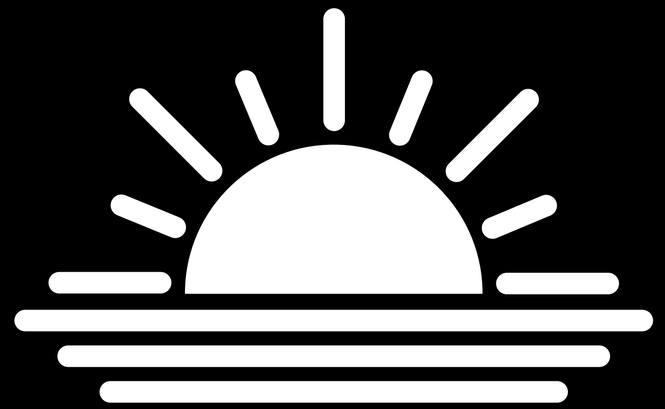
Event Triggers

HMSignificantTimeEvent

NEW

Activates on significant events:

- Sunrise



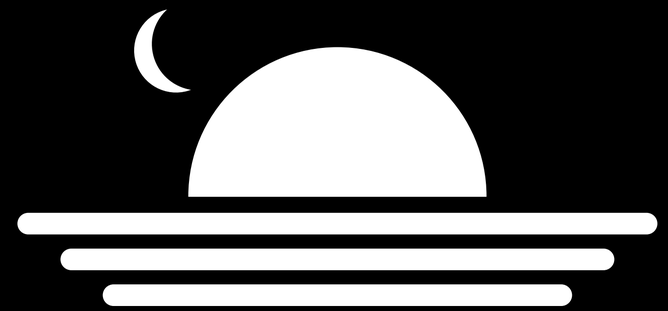
Event Triggers

HMSignificantTimeEvent

NEW

Activates on significant events:

- Sunrise
- Sunset



Event Triggers

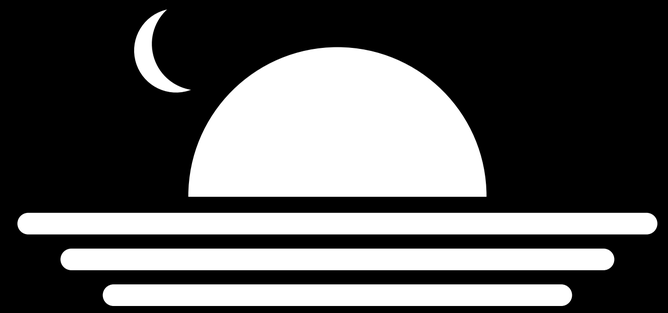
HMSignificantTimeEvent

NEW

Activates on significant events:

- Sunrise
- Sunset

Supports relative time offset



```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

```
//  Example - Creating significant time events

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: nil)
```

Event Triggers

Significant event conditions

NEW

```
extension HMEventTrigger {
    open class func predicate(forEvaluatingTriggerOccurringBetweenSignificantEvent
firstSignificantEvent: HMSignificantTimeEvent, secondSignificantEvent: HMSignificantTimeEvent)
-> NSPredicate}
```

Event Triggers

Significant event conditions

NEW

```
extension HMEventTrigger {
    open class func predicate(forEvaluatingTriggerOccurringBetweenSignificantEvent
firstSignificantEvent: HMSignificantTimeEvent, secondSignificantEvent: HMSignificantTimeEvent)
-> NSPredicate}
```

Event Triggers

HMCharacteristicThresholdRangeEvent

NEW

Event Triggers

HMCharacteristicThresholdRangeEvent

NEW

Activates on crossing threshold

Event Triggers

HMCharacteristicThresholdRangeEvent

NEW

Activates on crossing threshold

Supports:

- Minimum threshold

Event Triggers

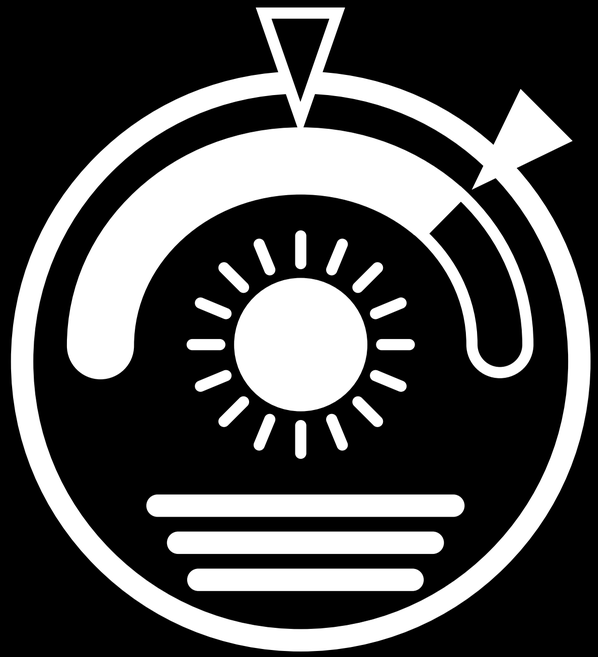
HMCharacteristicThresholdRangeEvent

NEW

Activates on crossing threshold

Supports:

- Minimum threshold
- Maximum threshold



Event Triggers

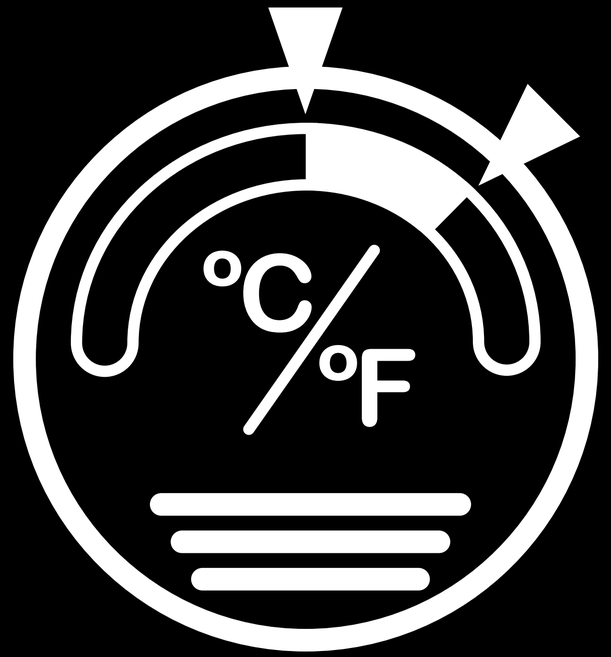
HMCharacteristicThresholdRangeEvent

NEW

Activates on crossing threshold

Supports:

- Minimum threshold
- Maximum threshold
- Range threshold



```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

```
//  Example - Characteristic threshold range events

import HomeKit

let characteristic = …

let numberRange = HMNumberRange(minValue: 76)
let event = HMCharacteristicThresholdRangeEvent(characteristic: characteristic,
                                                thresholdRange: numberRange)
let eventTrigger = HMEventTrigger(name: "Temperature over 76",
                                  events: [event],
                                  predicate: nil)
```

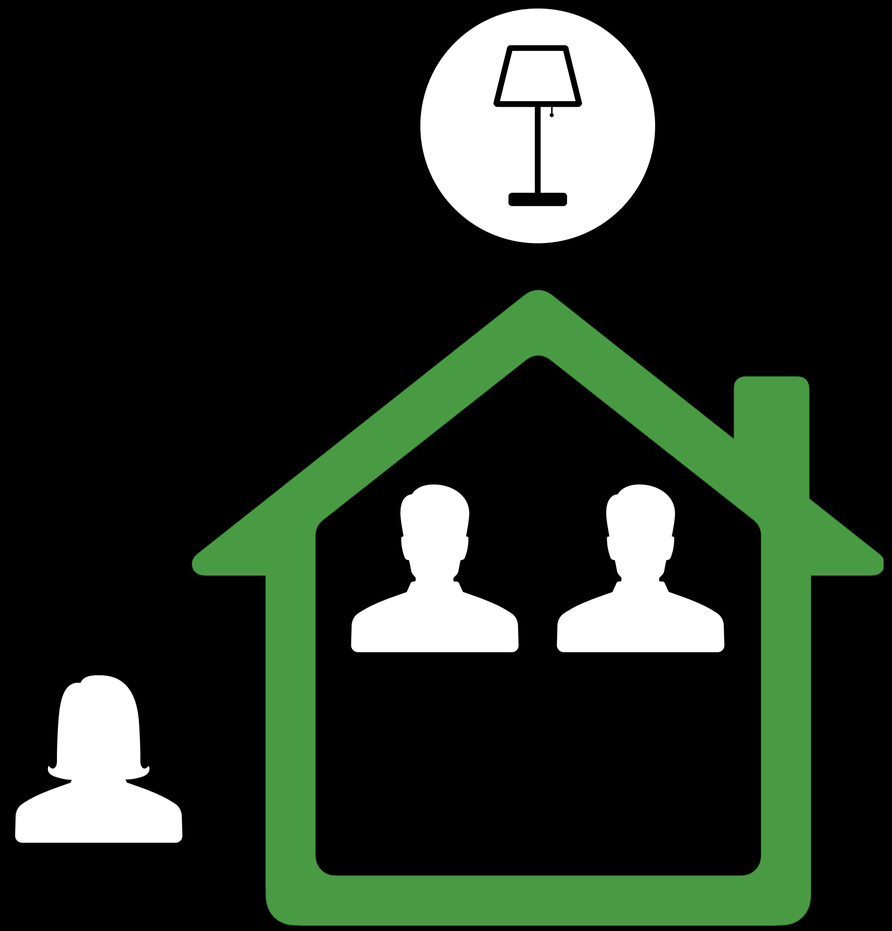
Event Triggers

HMPresenceEvent

NEW

Event Triggers

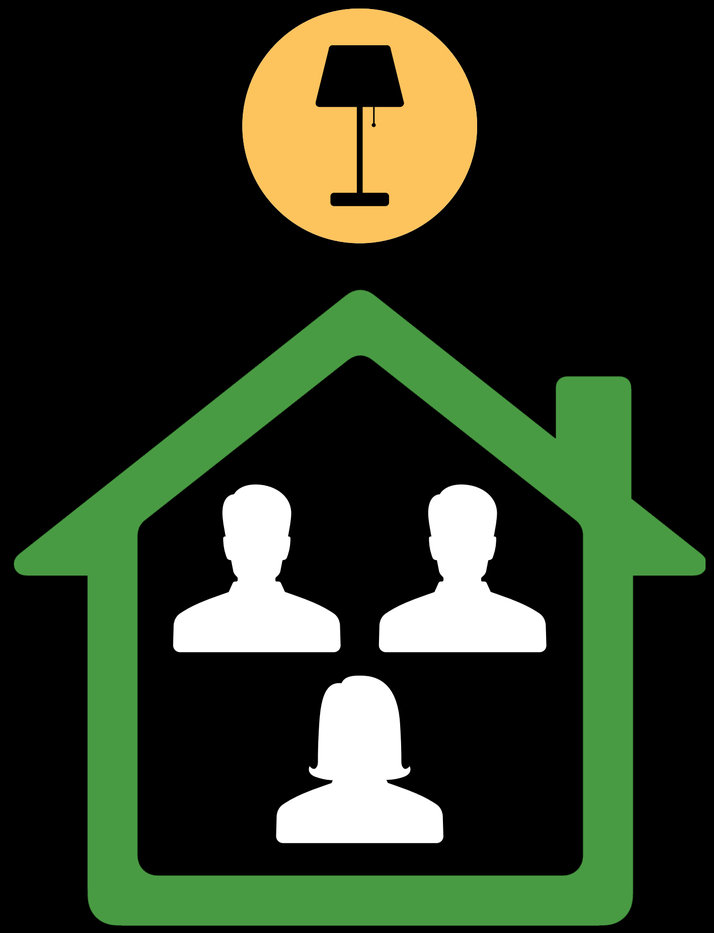
HMPresenceEvent



Event Triggers

HMPresenceEvent

Current user arrives home

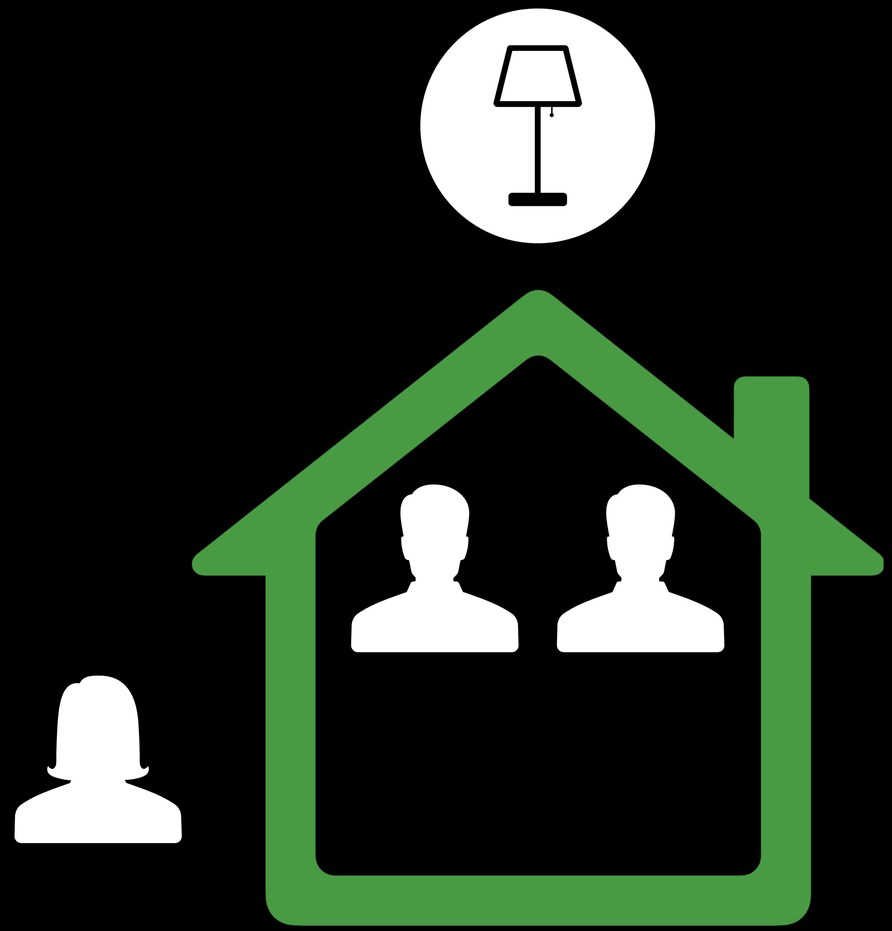


Event Triggers

HMPresenceEvent

Current user arrives home

Current user leaves home

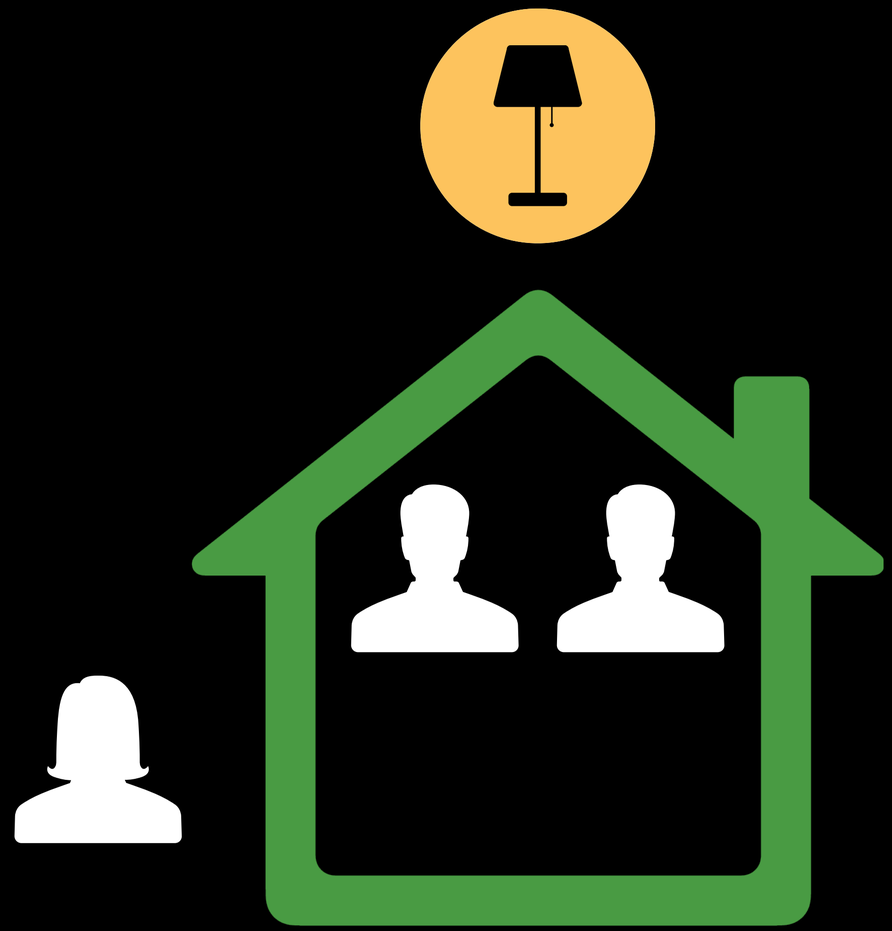


Event Triggers

HMPresenceEvent

Current user arrives home

Current user leaves home



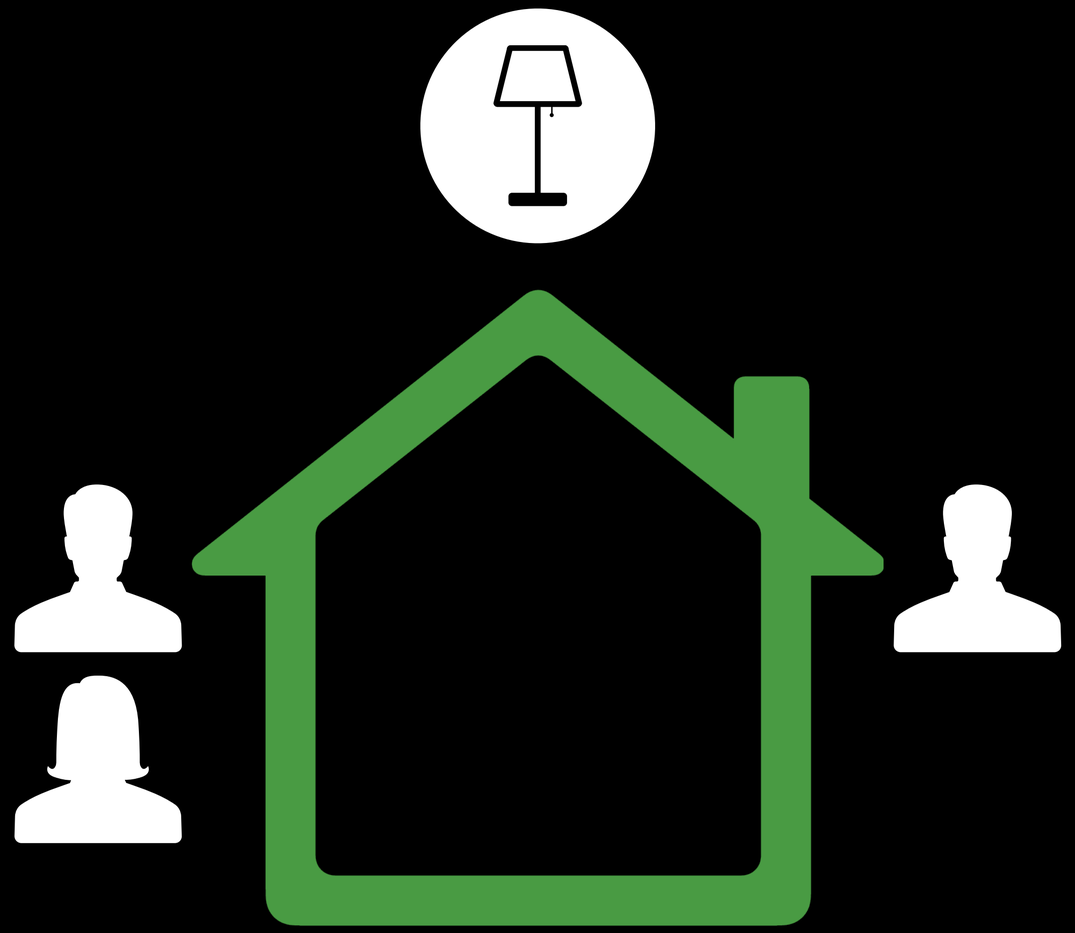
Event Triggers

HMPresenceEvent

Current user arrives home

Current user leaves home

Last user leaves home



Event Triggers

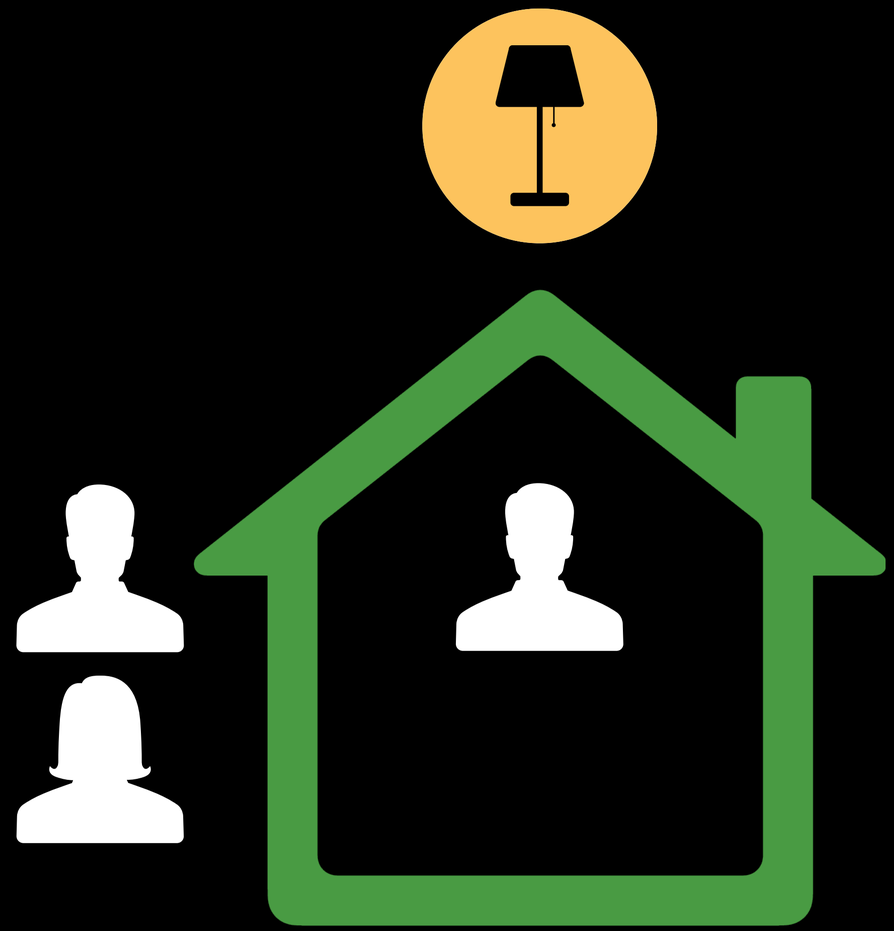
HMPresenceEvent

Current user arrives home

Current user leaves home

Last user leaves home

First user arrives home



```
// Example - Creating Presence Event

import HomeKit

let event = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let eventTrigger = HMEventTrigger(name: "Anyone Arrives Home",
                                  events: [event],
                                  predicate: nil)
```

```
// Example - Creating Presence Event

import HomeKit

let event = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let eventTrigger = HMEventTrigger(name: "Anyone Arrives Home",
                                  events: [event],
                                  predicate: nil)
```

```
// Example - Creating Presence Event

import HomeKit

let event = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let eventTrigger = HMEventTrigger(name: "Anyone Arrives Home",
                                  events: [event],
                                  predicate: nil)
```

```
// Example - Creating Presence Event

import HomeKit

let event = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let eventTrigger = HMEventTrigger(name: "Anyone Arrives Home",
                                  events: [event],
                                  predicate: nil)
```

```
// Example - Creating Presence Event

import HomeKit

let event = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let eventTrigger = HMEventTrigger(name: "Anyone Arrives Home",
                                  events: [event],
                                  predicate: nil)
```

Event Triggers

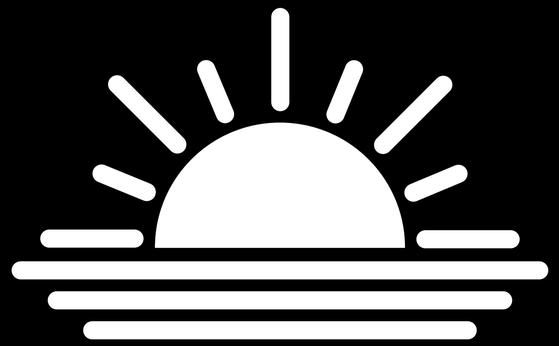
User presence conditions

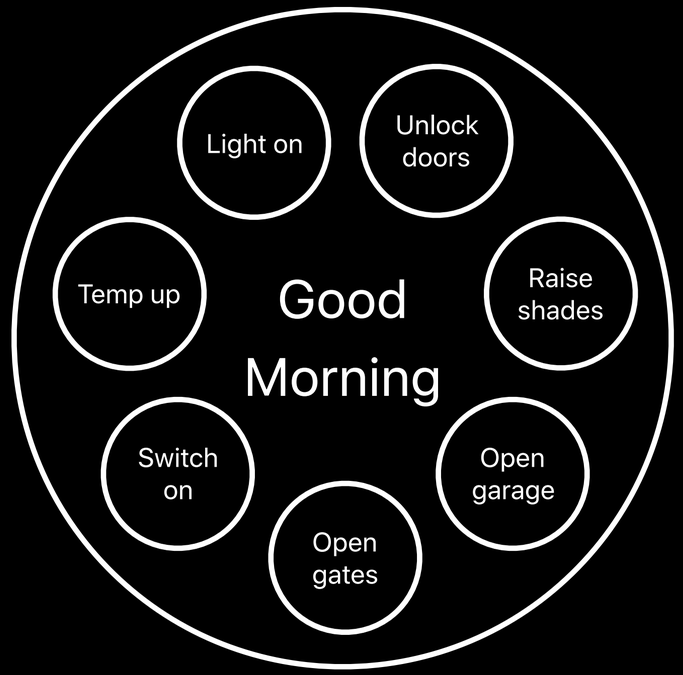
NEW

Event Triggers

User presence conditions

NEW





Event Triggers

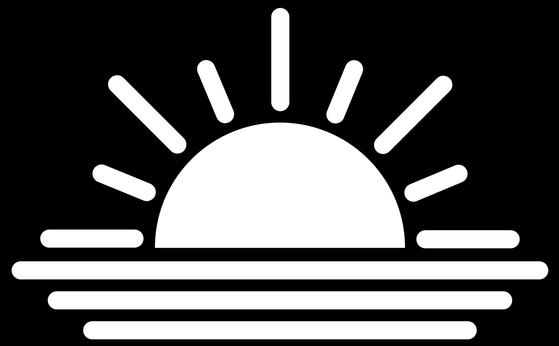
User presence conditions

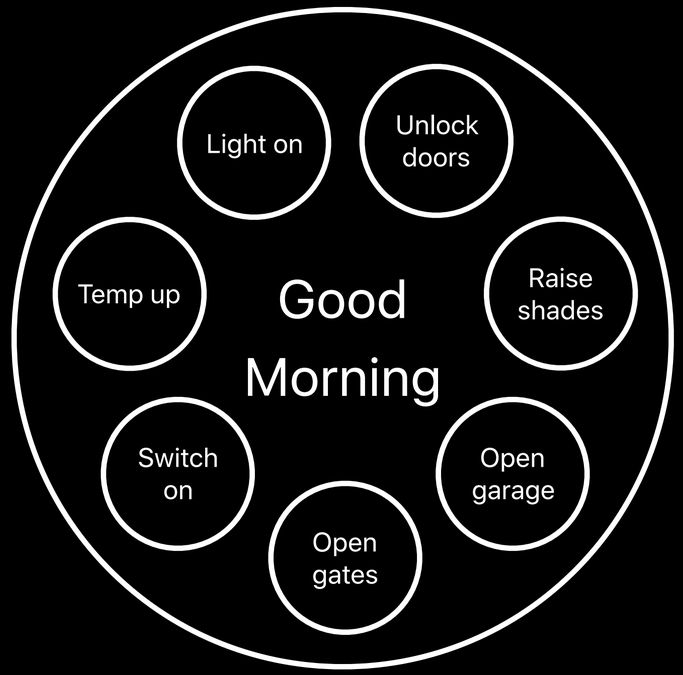
NEW

Event Triggers

User presence conditions

NEW





Event Triggers

User presence conditions

NEW

Event Triggers

User presence conditions

NEW

Event Triggers

User presence conditions

NEW

```
extension HMEventTrigger {
    open class func predicateForEvaluatingTrigger(withPresence presenceEvent: HMPresenceEvent)
-> NSPredicate
}
```

Event Triggers

User presence conditions

NEW

```
extension HMEventTrigger {
    open class func predicateForEvaluatingTrigger(withPresence presenceEvent: HMPresenceEvent)
-> NSPredicate
}
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

```
//  Example - User presence condition

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)

let presenceEvent = HMPresenceEvent(presenceType: HMPresenceType.anyUserAtHome)
let predicate = HMEventTrigger.predicateForEvaluatingTrigger(withPresence: presenceEvent)

let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  predicate: predicate)
```

Event Triggers

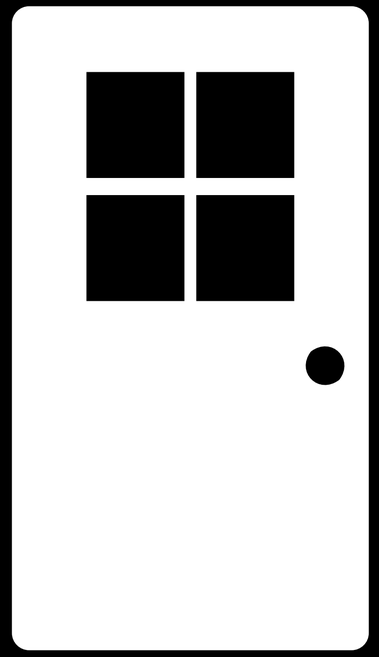
End events

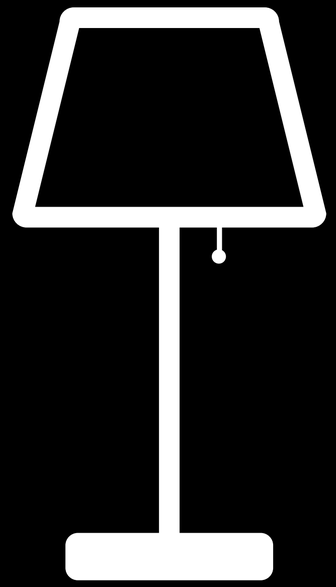
NEW

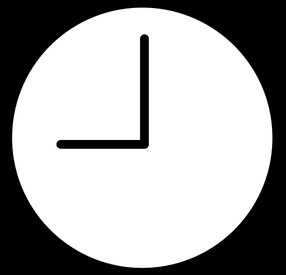
Event Triggers

End events

NEW



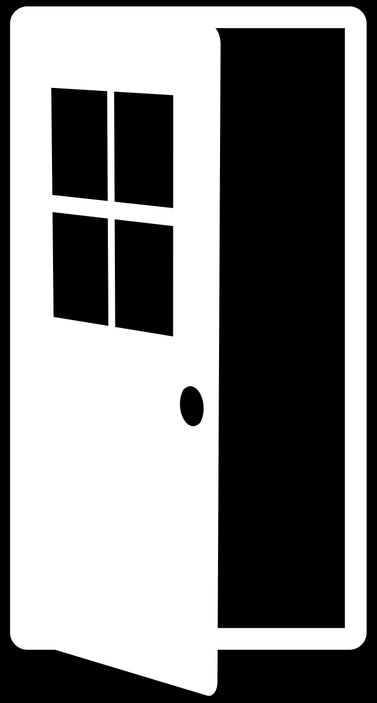


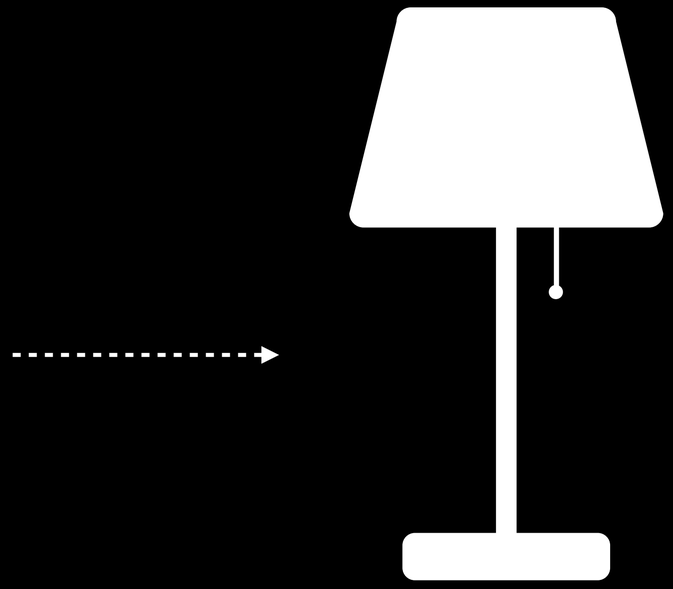


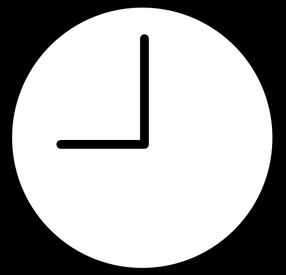
Event Triggers

End events

NEW



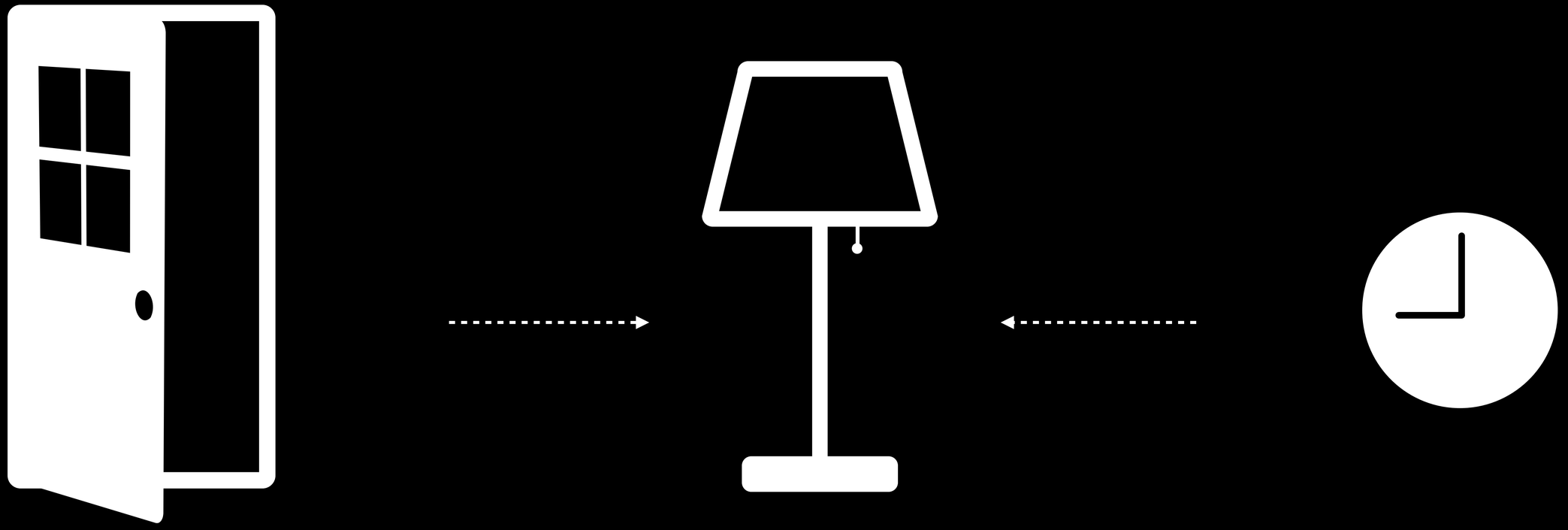




Event Triggers

End events

NEW



Event Triggers

End events

NEW

```
open class HMEventTrigger : HMTrigger {
    open var endEvents: [HMEvent] { get }
    open func updateEndEvents(_ endEvents: [HMEvent],
              completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

End events

NEW

```
open class HMEventTrigger : HMTrigger {
    open var endEvents: [HMEvent] { get }
    open func updateEndEvents(_ endEvents: [HMEvent],
              completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

End events

NEW

```
open class HMEventTrigger : HMTrigger {
    open var endEvents: [HMEvent] { get }
    open func updateEndEvents(_ endEvents: [HMEvent],
            completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

End events

NEW

```
open class HMEventTrigger : HMTrigger {
    open var endEvents: [HMEvent] { get }
    open func updateEndEvents(_ endEvents: [HMEvent],
              completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

End events

NEW

```
open class HMDurationEvent : NSCopying, NSMutableCopying {
    public init(duration: TimeInterval)
}
```

Event Triggers

Recurrence

NEW

Event Triggers

Recurrence

NEW

Event Triggers

Recurrence

NEW

```
open class HMEventTrigger : HMTrigger {
    open var recurrences: [DateComponents]? { get }
    open func updateRecurrences(_ recurrences: [DateComponents]?,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Recurrence

NEW

```
open class HMEventTrigger : HMTrigger {
    open var recurrences: [DateComponents]? { get }
    open func updateRecurrences(_ recurrences: [DateComponents]?,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Recurrence

NEW

```
open class HMEventTrigger : HMTrigger {
    open var recurrences: [DateComponents]? { get }
    open func updateRecurrences(_ recurrences: [DateComponents]?,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

```
//  Example - Recurrence

import HomeKit

let sunriseEvent = HMSignificantTimeEvent(significantEvent: HMSignificantEvent.sunrise,
                                          offset: nil)
var weekdays = [DateComponents]()
for weekday in 2...5 {
    var recurrence = DateComponents()
    recurrence.weekday = weekday
    weekdays.append(recurrence)
}
let eventTrigger = HMEventTrigger(name: "Sunrise, Daily",
                                  events: [sunriseEvent],
                                  end: nil,
                                  recurrences: weekdays,
                                  predicate: nil)
```

Event Triggers

Execute once

NEW

Event Triggers

Execute once

```
open class HMEventTrigger : HMTrigger {
    open var executeOnce: Bool { get }
    open func updateExecuteOnce(_ executeOnce: Bool,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Execute once

```
open class HMEventTrigger : HMTrigger {
    open var executeOnce: Bool { get }
    open func updateExecuteOnce(_ executeOnce: Bool,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Execute once

```
open class HMEventTrigger : HMTrigger {
    open var executeOnce: Bool { get }
    open func updateExecuteOnce(_ executeOnce: Bool,
                  completionHandler completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Mutable events

```
open class HMEventTrigger : HMTrigger {

    open func addEvent(_ event: HMEvent, completionHandler
                       completion: @escaping (Error?) -> Swift.Void)

    open func removeEvent(_ event: HMEvent, completionHandler
                          completion: @escaping (Error?) -> Swift.Void)
}
```

Event Triggers

Mutable events

```
open class HMEventTrigger : HMTrigger {
    open func updateEvents(_ events: [HMEvent], completionHandler
                           completion: @escaping (Error?) -> Swift.Void)
}
```

```
import HomeKit

if let sunriseEvent = eventTrigger.events.first as? HMSignificantTimeEvent {
    var mutableSunriseEvent = sunriseEvent.mutableCopy() as HMMutableSignificantTimeEvent

    //  Update the offset of the event

    var offset = DateComponents()
    offset.minute = 30
    mutableSunriseEvent.offset = offset

    //  Update the events of the trigger

    eventTrigger.updateEvents([mutableSunriseEvent]) { (e: Error?) in
        //  Handle error
    }
}
```

```
import HomeKit

if let sunriseEvent = eventTrigger.events.first as? HMSignificantTimeEvent {
    var mutableSunriseEvent = sunriseEvent.mutableCopy() as HMMutableSignificantTimeEvent

    //  Update the offset of the event

    var offset = DateComponents()
    offset.minute = 30
    mutableSunriseEvent.offset = offset

    //  Update the events of the trigger

    eventTrigger.updateEvents([mutableSunriseEvent]) { (e: Error?) in
        //  Handle error
    }
}
```

```swift
import HomeKit

if let sunriseEvent = eventTrigger.events.first as? HMSignificantTimeEvent {
    var mutableSunriseEvent = sunriseEvent.mutableCopy() as HMMutableSignificantTimeEvent

    //  Update the offset of the event

    var offset = DateComponents()
    offset.minute = 30
    mutableSunriseEvent.offset = offset

    //  Update the events of the trigger

    eventTrigger.updateEvents([mutableSunriseEvent]) { (e: Error?) in
        //  Handle error
    }
}
```

```
import HomeKit

if let sunriseEvent = eventTrigger.events.first as? HMSignificantTimeEvent {
    var mutableSunriseEvent = sunriseEvent.mutableCopy() as HMMutableSignificantTimeEvent

    //  Update the offset of the event

    var offset = DateComponents()
    offset.minute = 30
    mutableSunriseEvent.offset = offset

    //  Update the events of the trigger

    eventTrigger.updateEvents([mutableSunriseEvent]) { (e: Error?) in
        //  Handle error
    }
}
```

```
import HomeKit

if let sunriseEvent = eventTrigger.events.first as? HMSignificantTimeEvent {
    var mutableSunriseEvent = sunriseEvent.mutableCopy() as HMMutableSignificantTimeEvent

    //  Update the offset of the event

    var offset = DateComponents()
    offset.minute = 30
    mutableSunriseEvent.offset = offset

    //  Update the events of the trigger

    eventTrigger.updateEvents([mutableSunriseEvent]) { (e: Error?) in
        //  Handle error
    }
}
```

Event Triggers

Event Triggers

New Events

Event Triggers

New Events

New Conditions

Event Triggers

New Events

New Conditions

End Events

Event Triggers

New Events

New Conditions

End Events

Recurrence

Event Triggers

New Events

New Conditions

End Events

Recurrence

Mutable Events

Accessory Updates

Praveen Chegondi, HomeKit Engineering

Specification

Specification

Protocol Enhancements

Specification

Protocol Enhancements

Categories

Specification

Protocol Enhancements

Categories

Authentication

Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

HomeKit Accessory Protocol Specification

HomeKit Accessory Protocol Specification

Communication between iOS and Accessory

HomeKit Accessory Protocol Specification

Communication between iOS and Accessory

Security

HomeKit Accessory Protocol Specification

Communication between iOS and Accessory

Security

Transports

- IP
- Bluetooth LE

HomeKit Accessory Protocol Specification

Communication between iOS and Accessory

Security

Transports

- IP
- Bluetooth LE

Accessory categories

HomeKit Accessory Protocol Specification

HomeKit Accessory Protocol Specification

HomeKit Accessory Protocol Specification

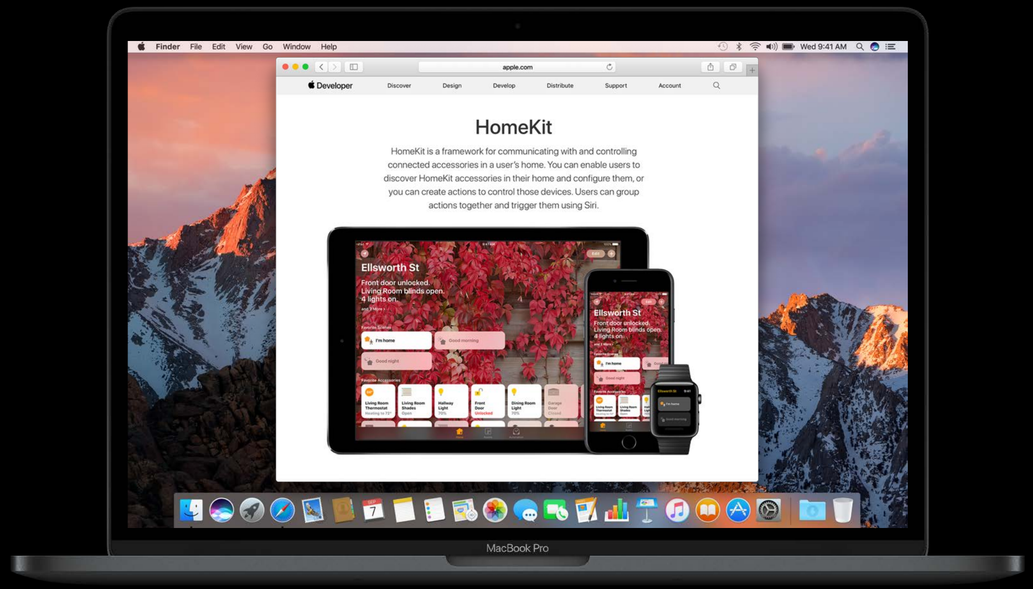
NEW

HomeKit Accessory Protocol Specification

NEW

Available to all developers today

http://developer.apple.com/homekit



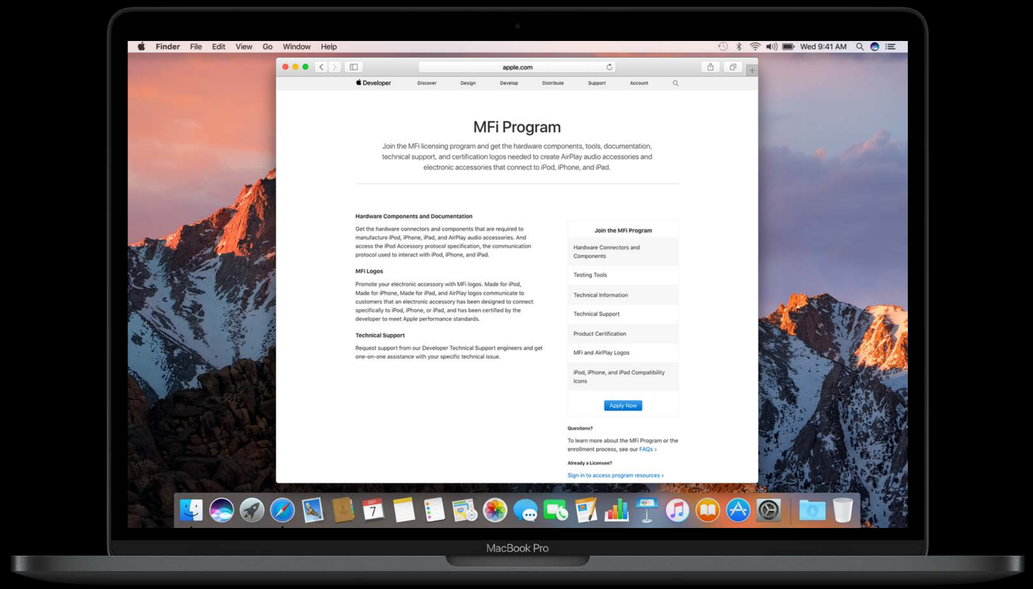
HomeKit Accessory Protocol Specification

NEW

Available to all developers today
http://developer.apple.com/homekit

Commercialize HomeKit accessory

Become MFi Licensee
http://developer.apple.com/mfi/



Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

Accessory Setup

Accessory Setup

Power on accessory

Accessory Setup

Power on accessory

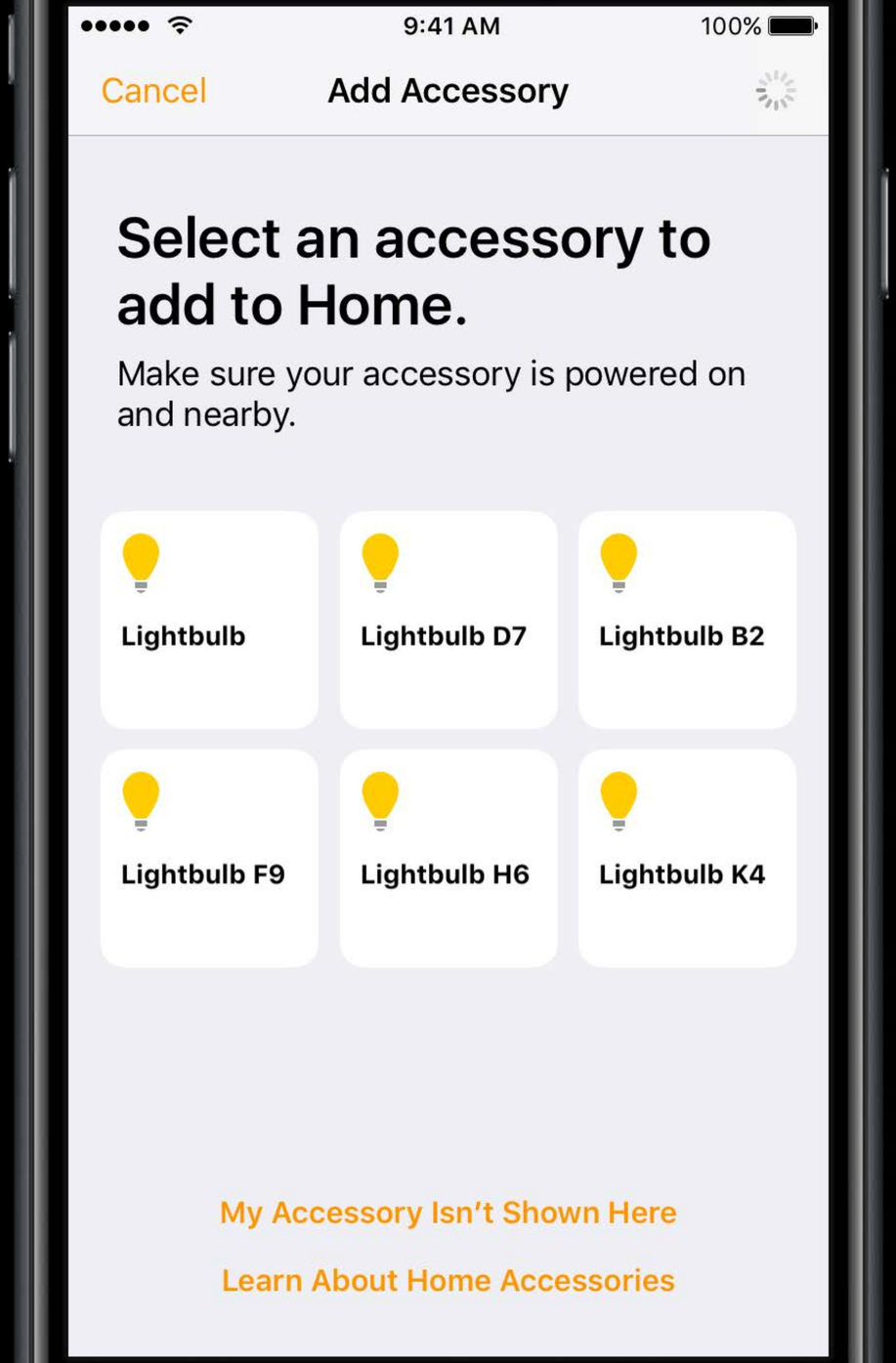
Choose accessory

Accessory Setup

Power on accessory

Choose accessory

HomeKit pairing


9:41 AM
100%
Cancel
Add Accessory
Select an accessory to add to Home.
Make sure your accessory is powered on and nearby.
Lightbulb
Lightbulb D7
Lightbulb B2
Lightbulb F9
Lightbulb H6
Lightbulb K4
My Accessory Isn't Shown Here
Learn About Home Accessories

Enhanced Setup Code

NEW

Enhanced Setup Code

NEW

Scan first

Enhanced Setup Code

NEW

Scan first

Automatic selection

Enhanced Setup Code

NEW

Scan first

Automatic selection

Enhanced setup code

- Setup ID
- Setup code (8-digit)

Enhanced Setup Code

NEW

Enhanced Setup Code

NEW

QR Code

Enhanced Setup Code

NEW

QR Code

- Standard-based

Enhanced Setup Code

NEW

QR Code

- Standard-based
- Small-form factor

10mm x 10mm

Enhanced Setup Code

NEW

Tap-to-pair

- NFC tags

9:41 AM 100%

Cancel

Add Accessory

Pair with HomeKit Code

Position the HomeKit code in the frame.

Look for the HomeKit code in the packaging or on the accessory.

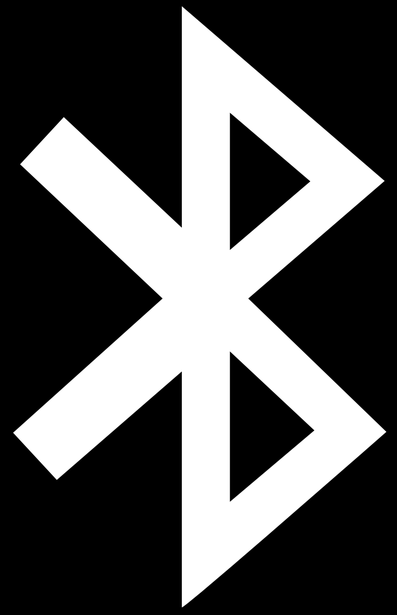
Pair with NFC

Place your iPhone near the NFC logo.

If the accessory is not recognized, make sure that it is powered on or try resetting the accessory.

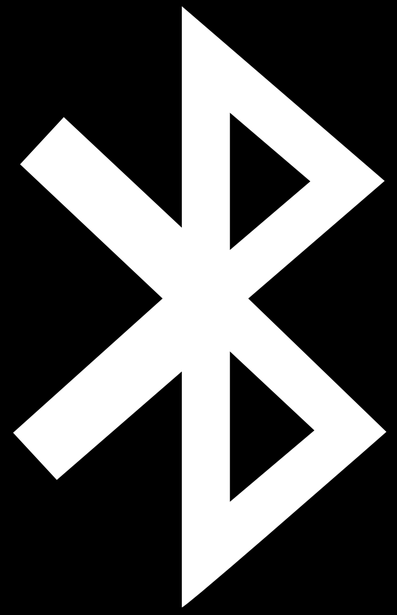
Don’t Have or Can’t Scan Code

HomeKit Bluetooth LE Accessories



HomeKit Bluetooth LE Accessories

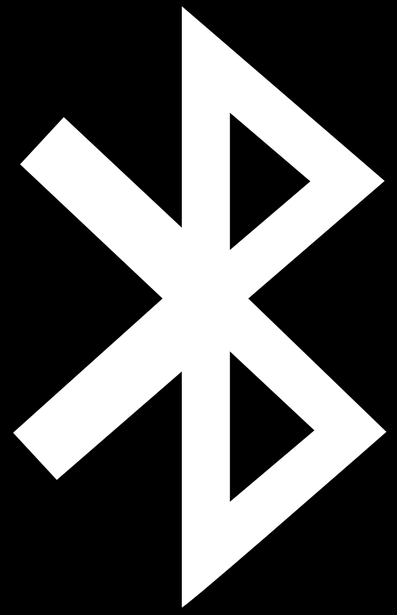
Form-factor friendly



HomeKit Bluetooth LE Accessories

Form-factor friendly

Low-power

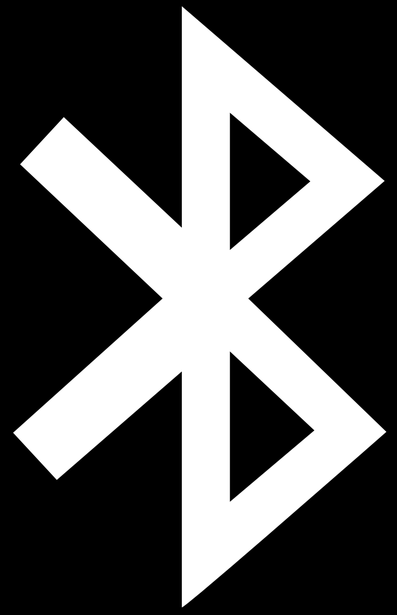


HomeKit Bluetooth LE Accessories

Form-factor friendly

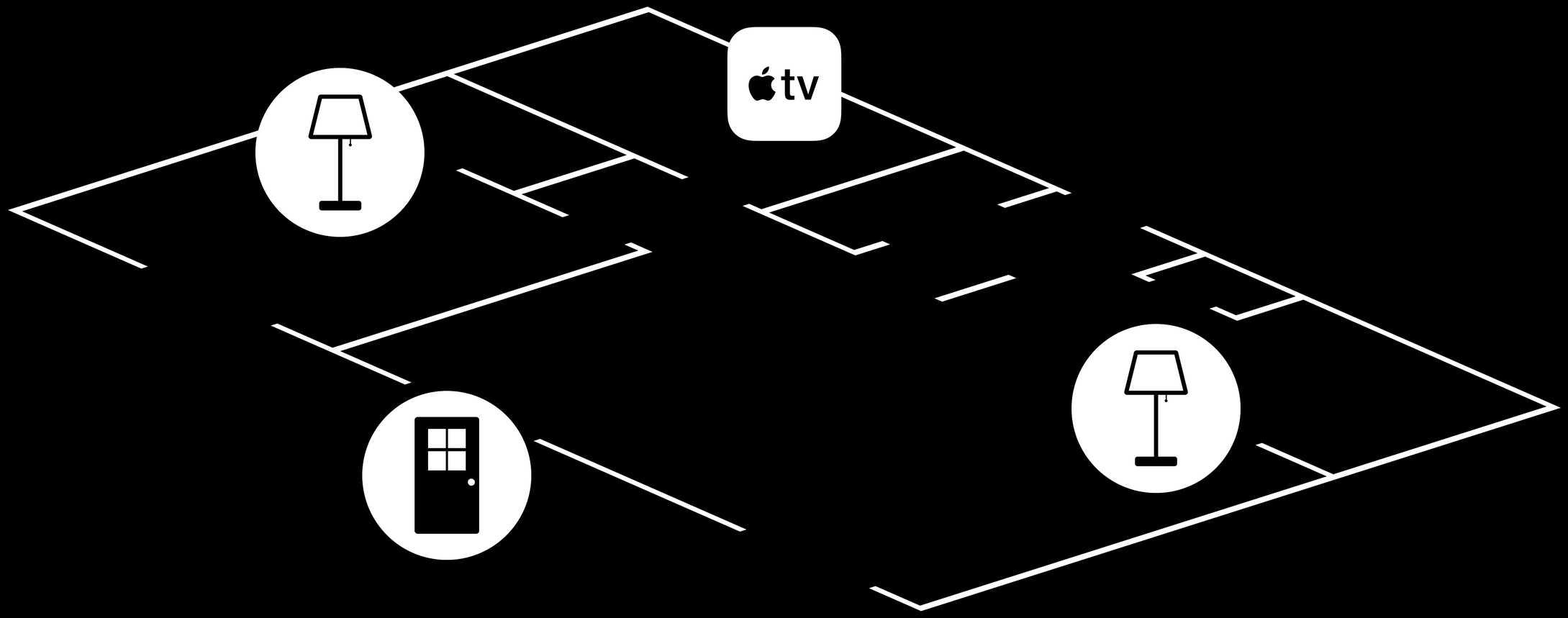
Low-power

Battery-operated



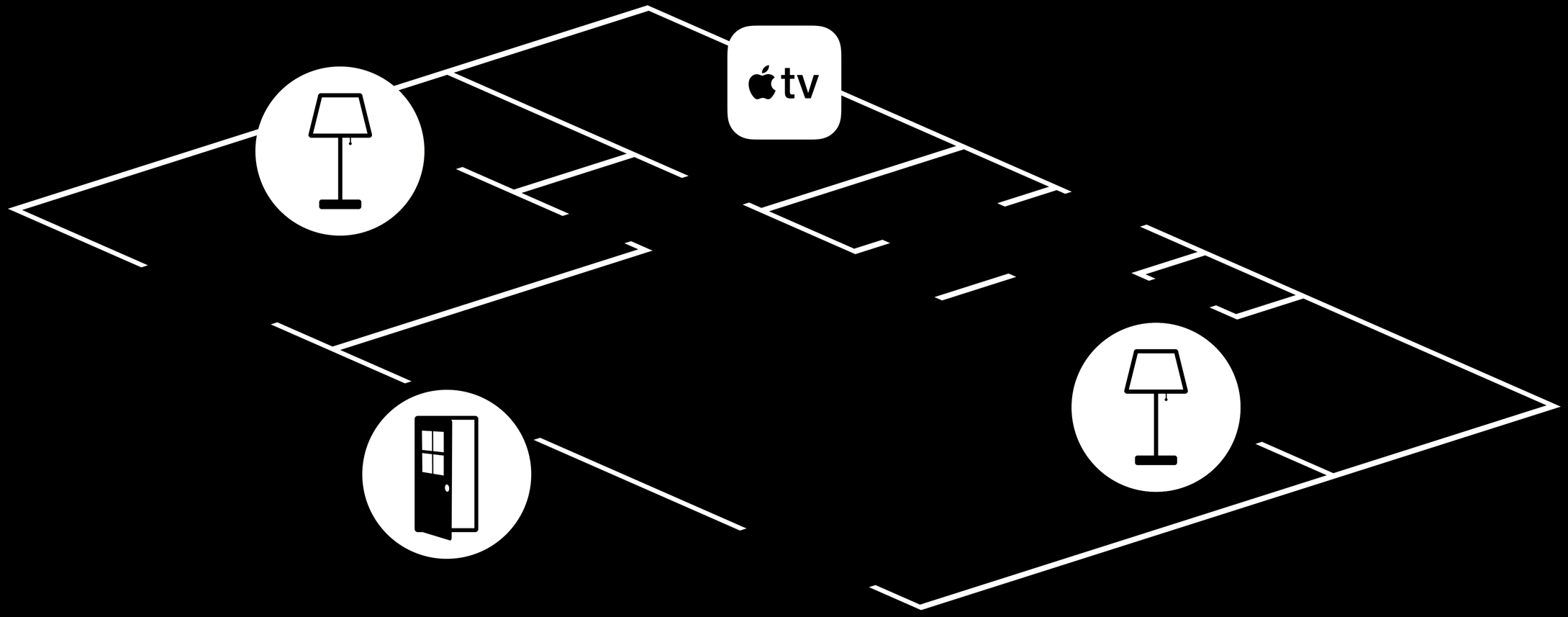
HomeKit Bluetooth LE Accessories

Broadcast notifications



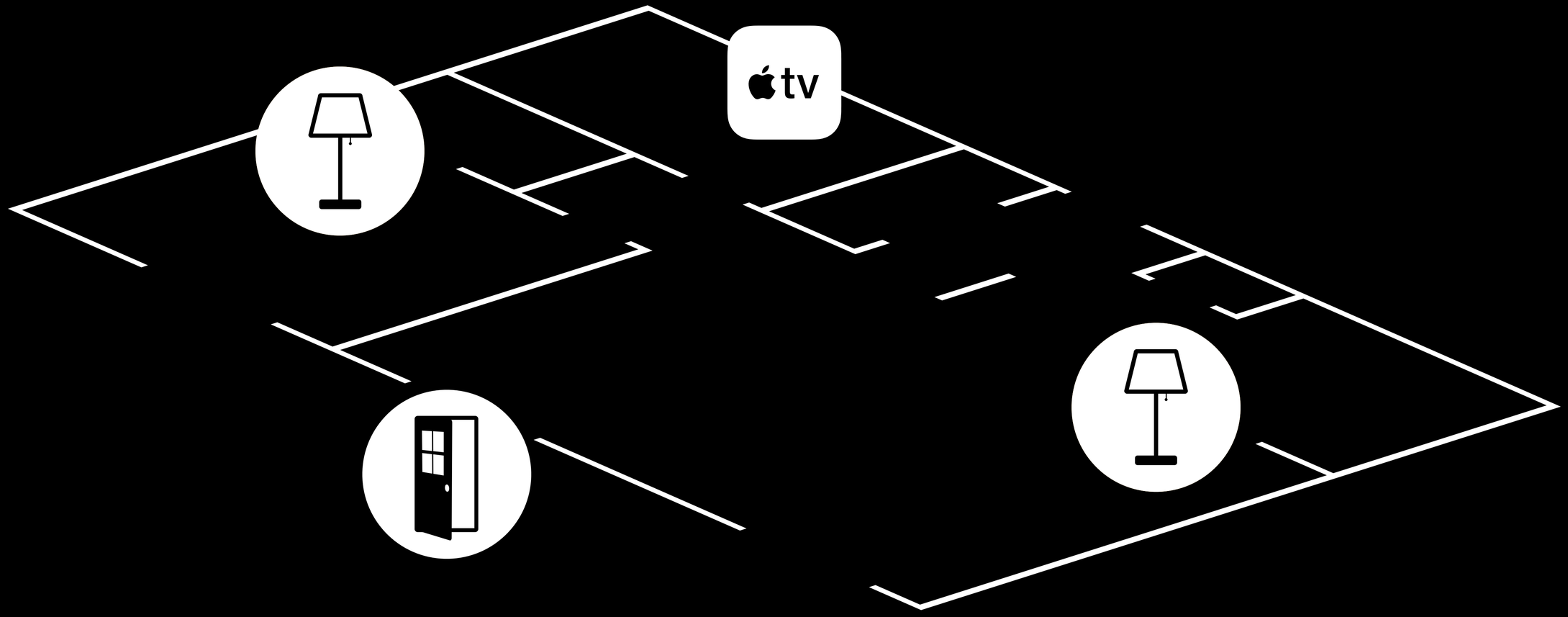
HomeKit Bluetooth LE Accessories

Broadcast notifications



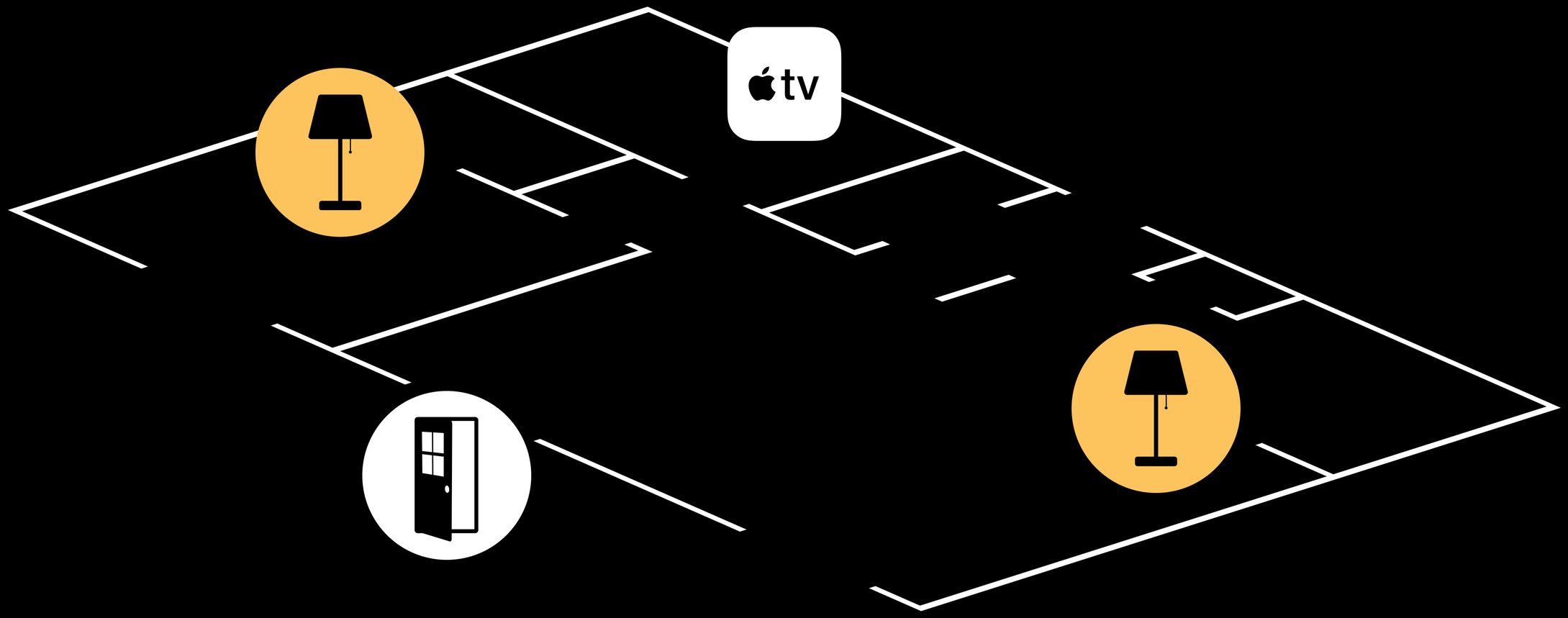
HomeKit Bluetooth LE Accessories

Broadcast notifications



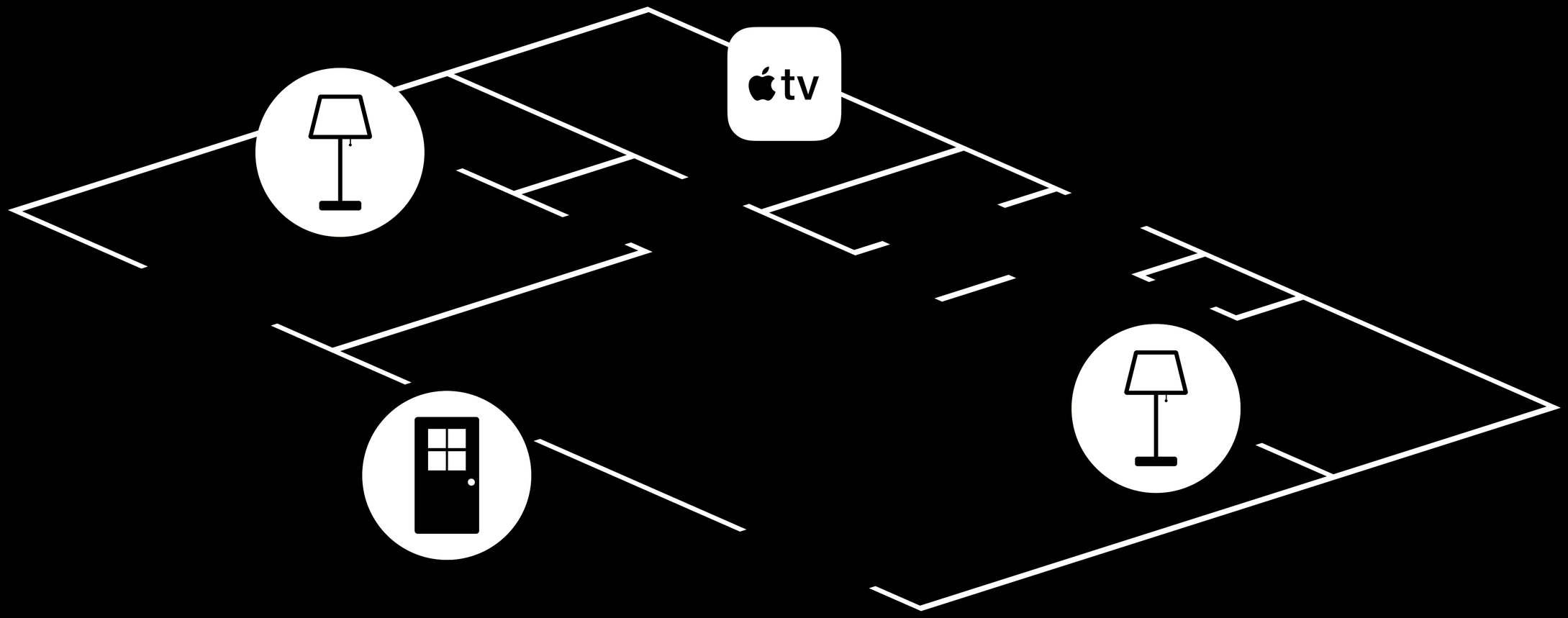
HomeKit Bluetooth LE Accessories

Broadcast notifications



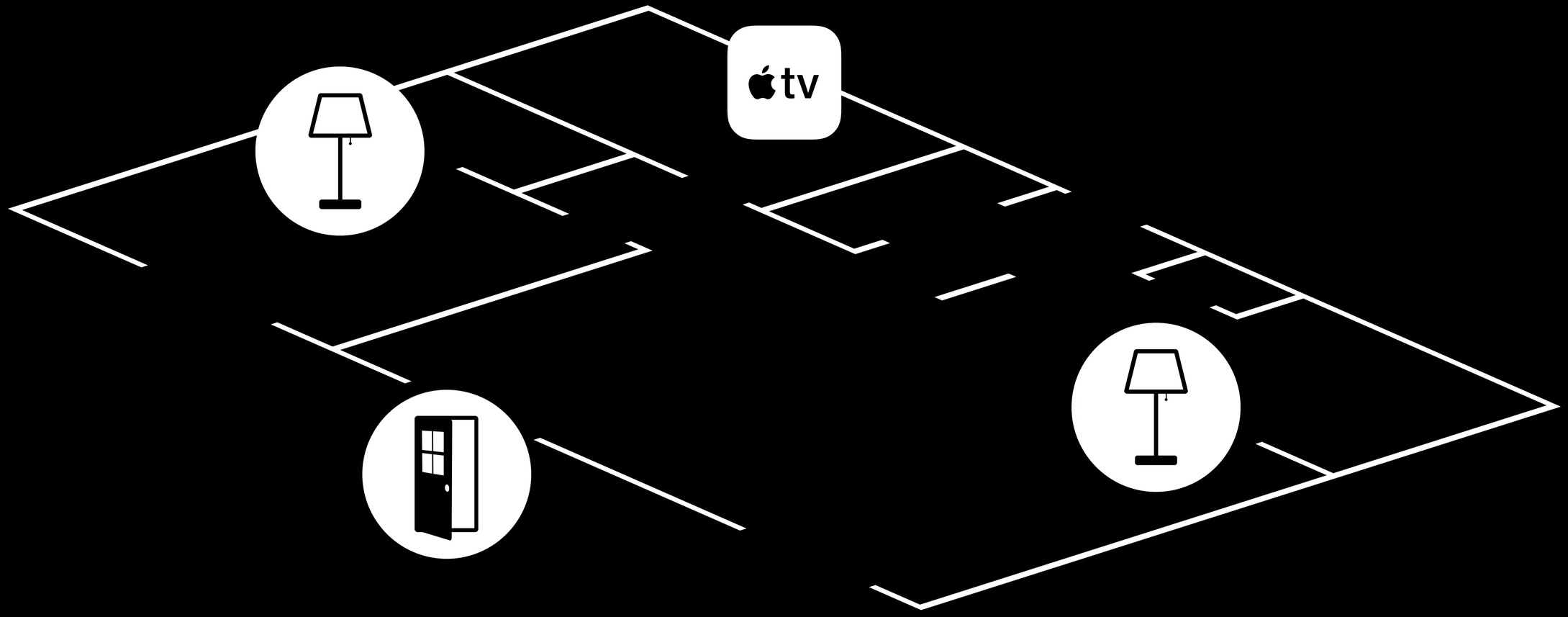
HomeKit Bluetooth LE Accessories

Broadcast notifications



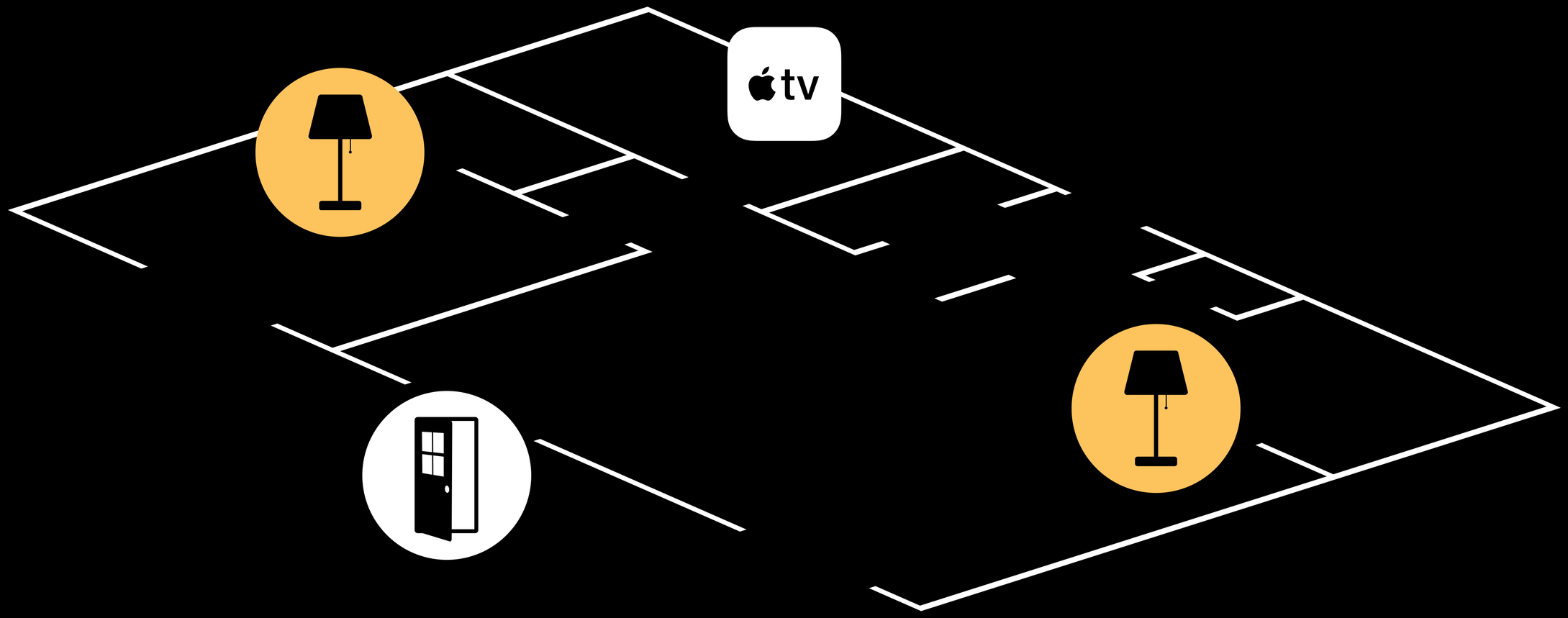
HomeKit Bluetooth LE Accessories

Broadcast notifications



HomeKit Bluetooth LE Accessories

Broadcast notifications



HomeKit Bluetooth LE Accessories

Broadcast notifications

NEW

HomeKit Bluetooth LE Accessories

Broadcast notifications

NEW

Secure-broadcast notifications

HomeKit Bluetooth LE Accessories

Broadcast notifications

NEW

Secure-broadcast notifications

Sub-second latencies

Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

Accessory Categories

Accessory Categories


Garage Doors

Thermostats


Sensors


Window Shades


Security


Humidifiers

Air Conditioners


Locks


Air Purifiers


Lights


Outlets


Fans


Cameras


Doorbells

Accessory Categories


Garage Doors

Thermostats


Sensors


Window Shades


Security


Humidifiers

Air Conditioners


Locks


Air Purifiers


Lights


Outlets


Fans


Cameras


Doorbells


Sprinklers


Faucets

Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

Accessory Authentication

Accessory Authentication

Quality

Accessory Authentication

Quality

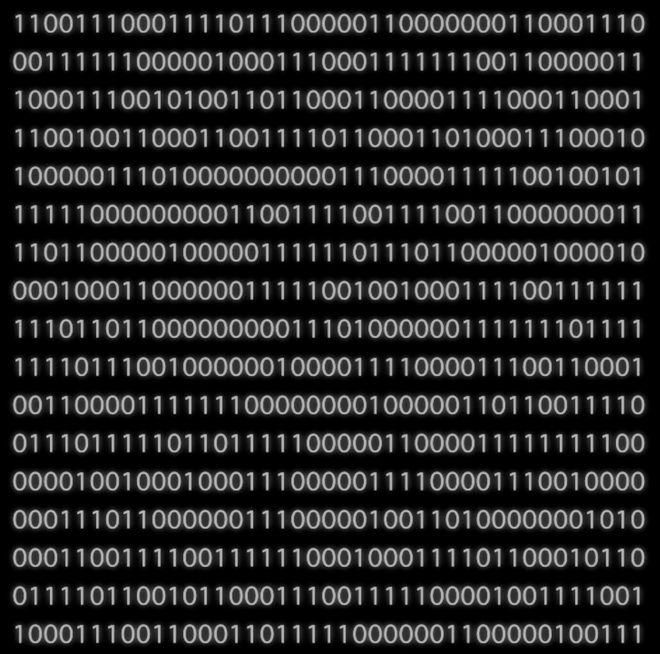
Trusted

Accessory Authentication

Quality

Trusted

Hardware-based authentication



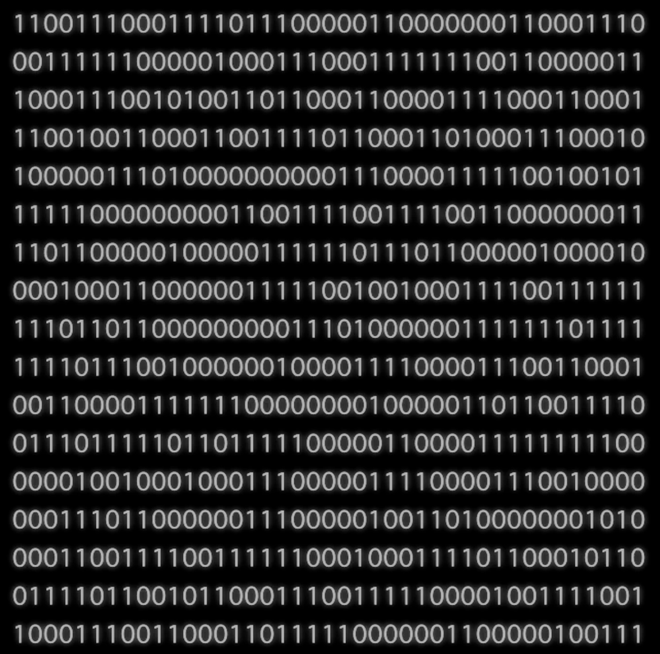
Accessory Authentication

NEW

Accessory Authentication

NEW

Software-based authentication

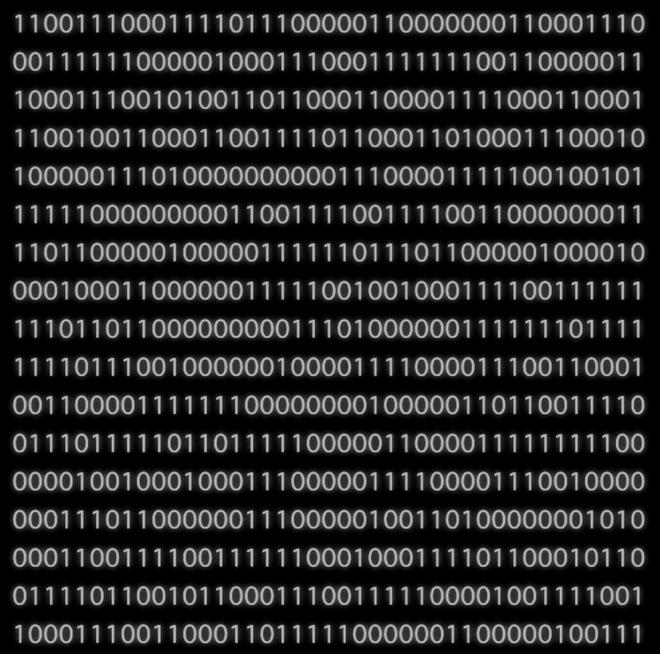


Accessory Authentication

NEW

Software-based authentication

Enables HomeKit on shipping accessories



Specification

Protocol Enhancements

Categories

Authentication

Self-Certification

Self-Certification

Self-Certification

Improved work flow

- HomeKit Certification Assistant
- Increase audit capacity

Self-Certification

Improved work flow

- HomeKit Certification Assistant
- Increase audit capacity

Apple Authorized Labs

- United States
- China
- UK

Summary

Event Triggers

HomeKit Specifications

Protocol Enhancements

New Accessory Categories

Self-Certification Process

More Information

https://developer.apple.com/wwdc17/705

Related Sessions

What’s New in Core Bluetooth	Grand Ballroom B	Thursday 11:00AM

Labs

HomeKit Lab 1	Technology Lab J	Wed 10:00AM–11:30AM
HomeKit Lab 2	Technology Lab J	Thu 4:10PM–5:40PM


WWDC17