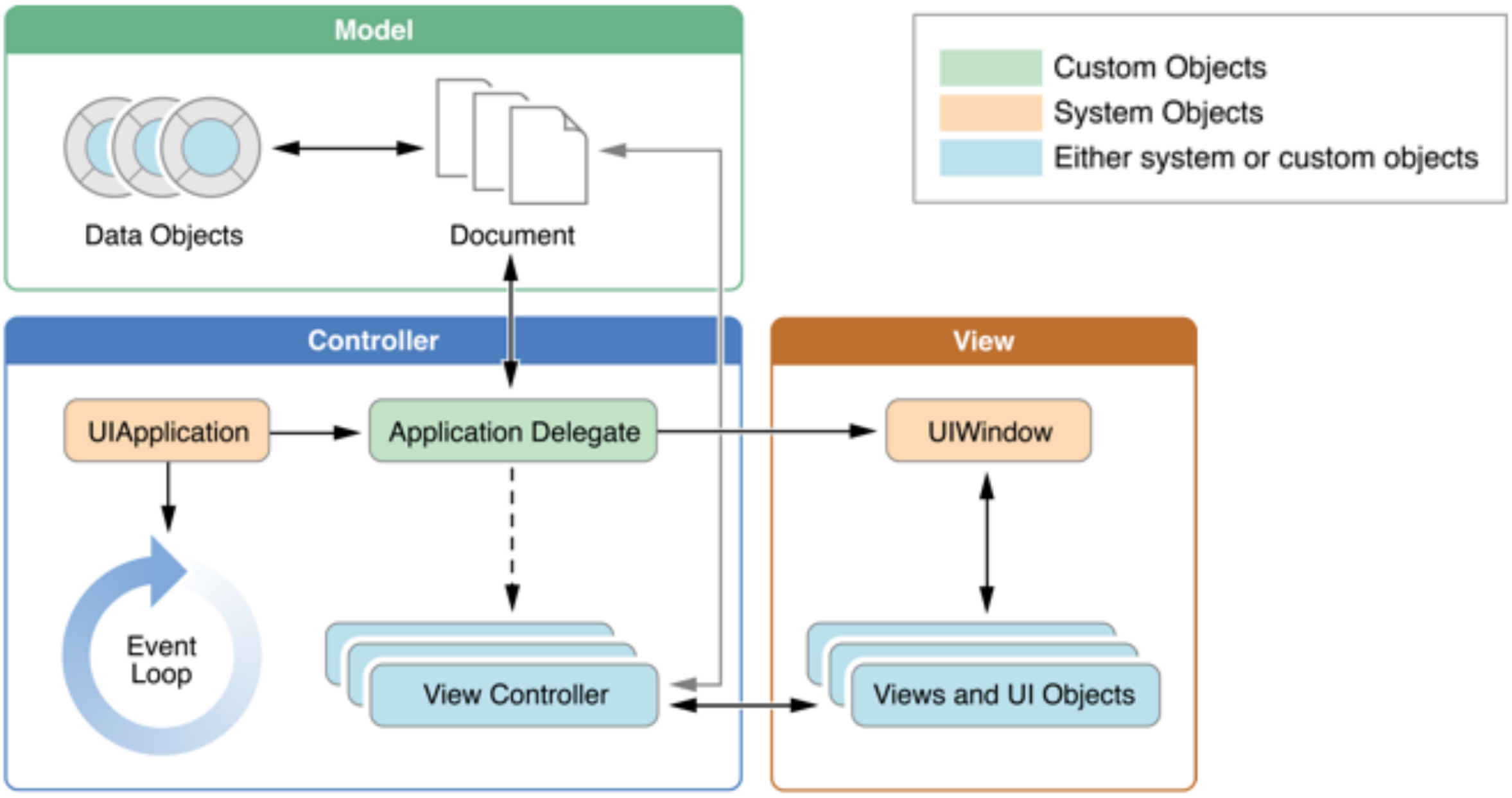
Fast campus

The App Life Cycle

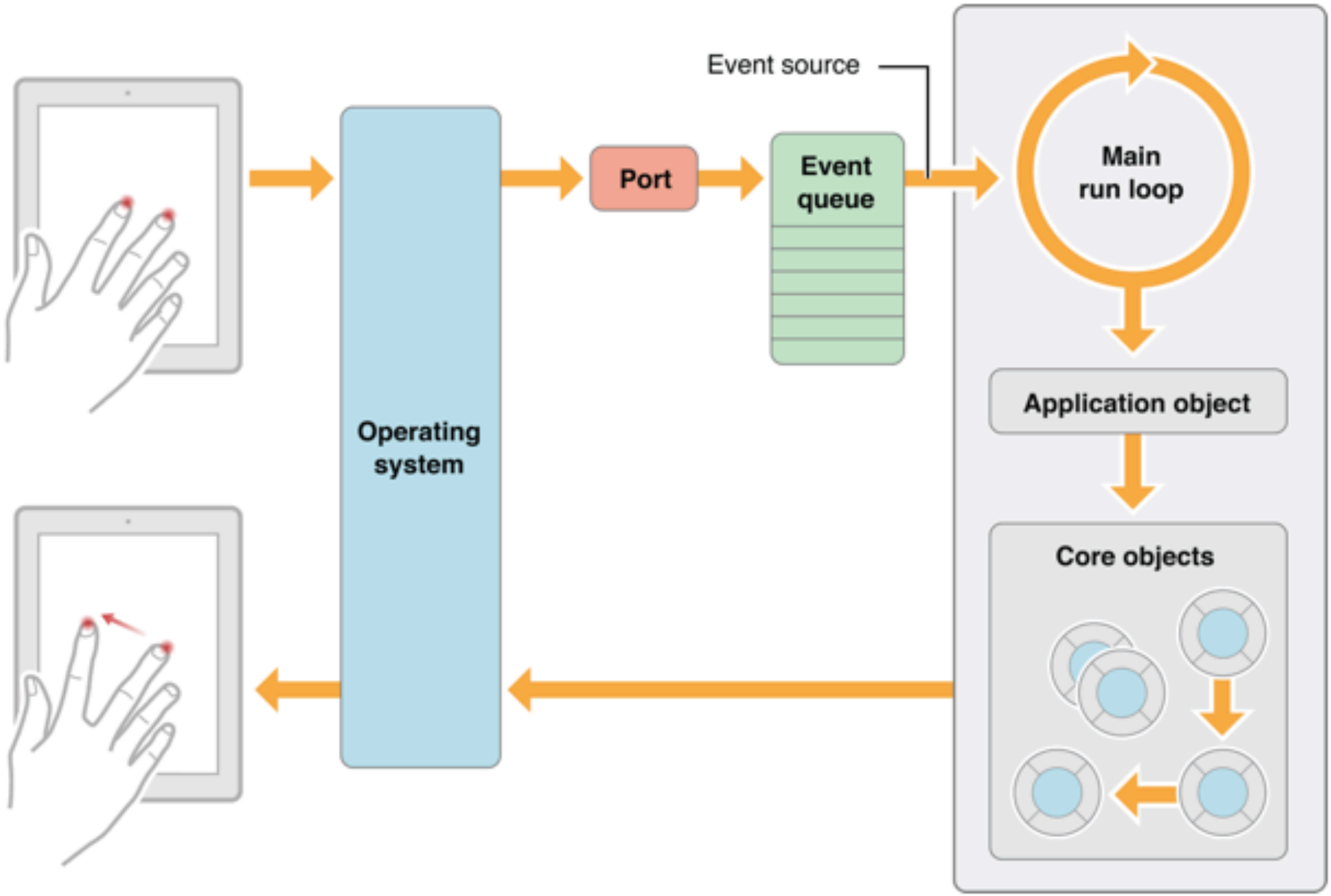
Giftbot

The Structure of an App

During startup, the UIApplicationMain function sets up several key objects and starts the app running. At the heart of every iOS app is the UIApplication object



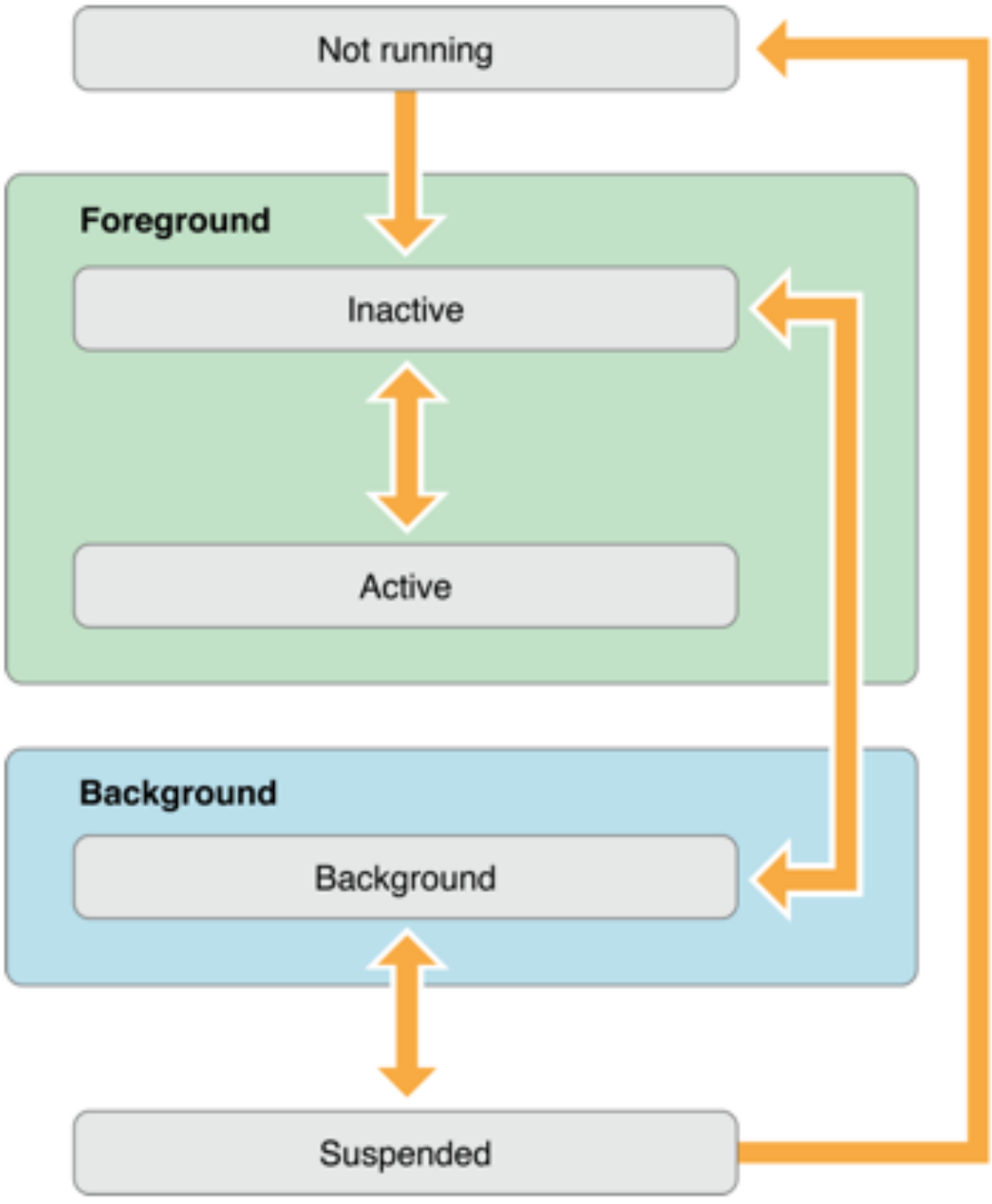
The Main Run Loop



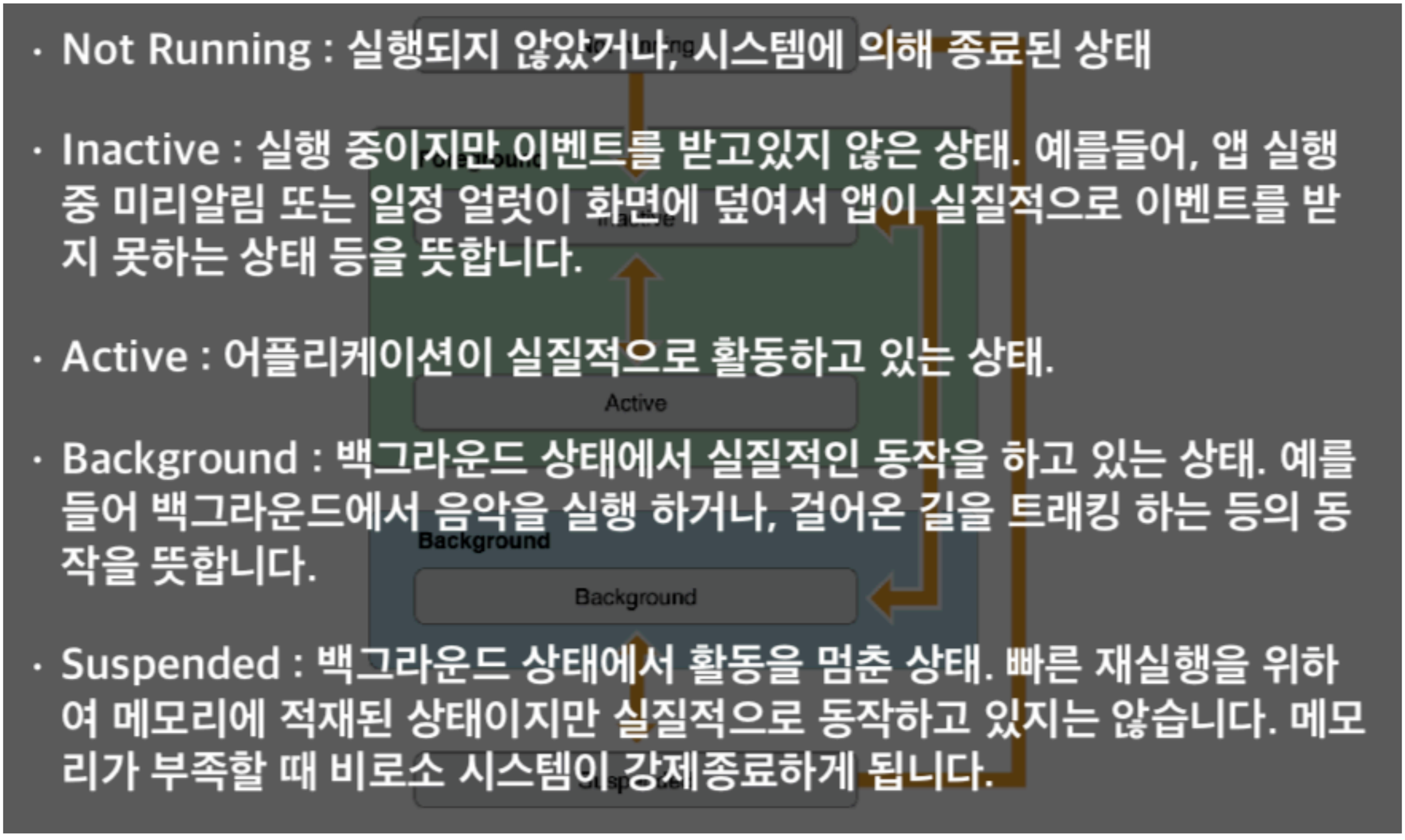
Common types of events

Event type	Delivered to...	Notes
Touch	The view object in which the event occurred	Views are responder objects. Any touch events not handled by the view are forwarded down the responder chain for processing.
Remote control Shake motion events	First responder object	Remote control events are for controlling media playback and are generated by headphones and other accessories.
Accelerometer Magnetometer Gyroscope	The object you designate	Events related to the accelerometer, magnetometer, and gyroscope hardware are delivered to the object you designate.
Location	The object you designate	You register to receive location events using the Core Location framework. For more information about using Core Location, see *Location and Maps Programming Guide*.
Redraw	The view that needs the update	Redraw events do not involve an event object but are simply calls to the view to draw itself. The drawing architecture for iOS is described in *Drawing and Printing Guide for iOS*.

State changes in an iOS app



State changes in an iOS app



- Not Running : 실행되지 않았거나, 시스템에 의해 종료된 상태
- Inactive : 실행 중이지만 이벤트를 받고있지 않은 상태. 예를들어, 앱 실행 중 미리알림 또는 일정 얼럿이 화면에 덮여서 앱이 실질적으로 이벤트를 받지 못하는 상태 등을 뜻합니다.
- Active : 어플리케이션이 실질적으로 활동하고 있는 상태.
- Background : 백그라운드 상태에서 실질적인 동작을 하고 있는 상태. 예를들어 백그라운드에서 음악을 실행 하거나, 걸어온 길을 트래킹 하는 등의 동작을 뜻합니다.
- Suspended : 백그라운드 상태에서 활동을 멈춘 상태. 빠른 재실행을 위하여 메모리에 적재된 상태이지만 실질적으로 동작하고 있지는 않습니다. 메모리가 부족할 때 비로소 시스템이 강제종료하게 됩니다.

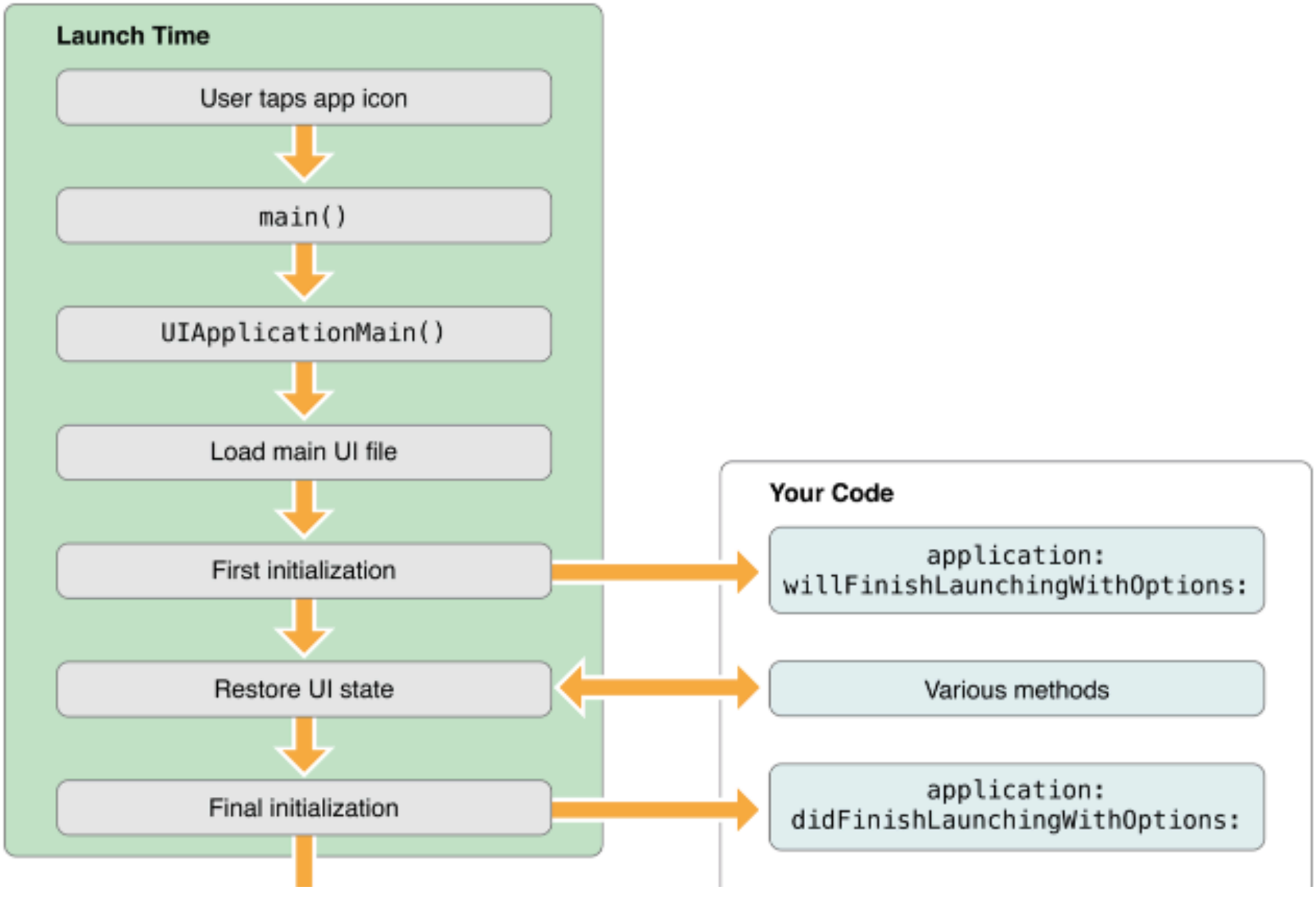
Execution States for Apps

- `application:willFinishLaunchingWithOptions:`
  - 어플리케이션이 최초 실행될 때 호출되는 메소드
- `application:didFinishLaunchingWithOptions:`
  - 어플리케이션이 실행된 직후 사용자의 화면에 보여지기 직전에 호출.
- `applicationDidBecomeActive:`
  - 어플리케이션이 Active 상태로 전환된 직후 호출.
- `applicationWillResignActive:`
  - 어플리케이션이 Inactive 상태로 전환되기 직전 호출
- `applicationDidEnterBackground:`
  - 어플리케이션이 백그라운드 상태로 전환된 직후 호출.
- `applicationWillEnterForeground:`
  - 어플리케이션이 Active 상태가 되기 직전에, 화면에 보여지기 직전의 시점에 호출.
- `applicationWillTerminate:`
  - 어플리케이션이 종료되기 직전에 호출.

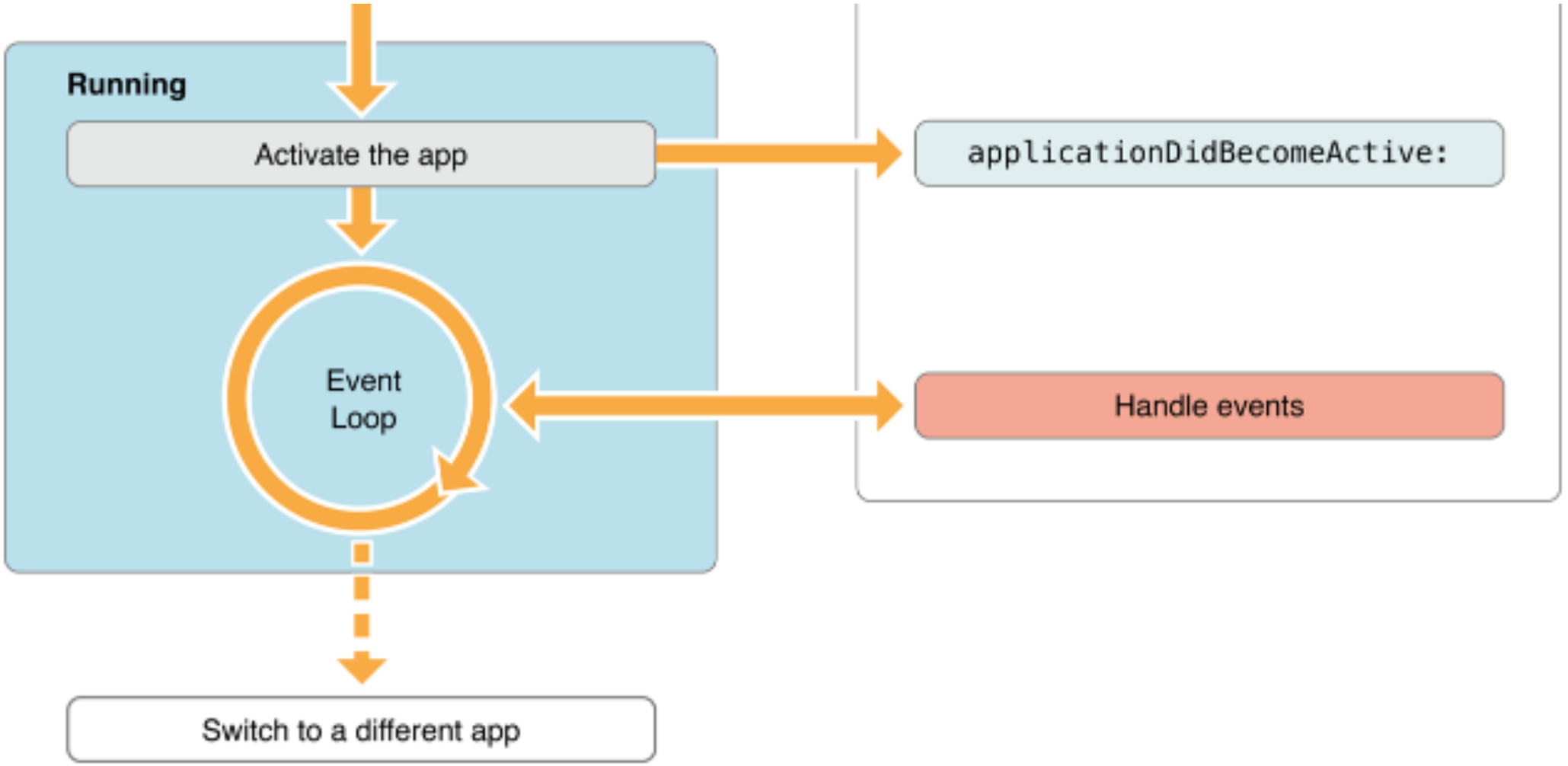
Execution States for Apps

상태가 변경될 때마다 호출되는 메서드 직접 확인해보기

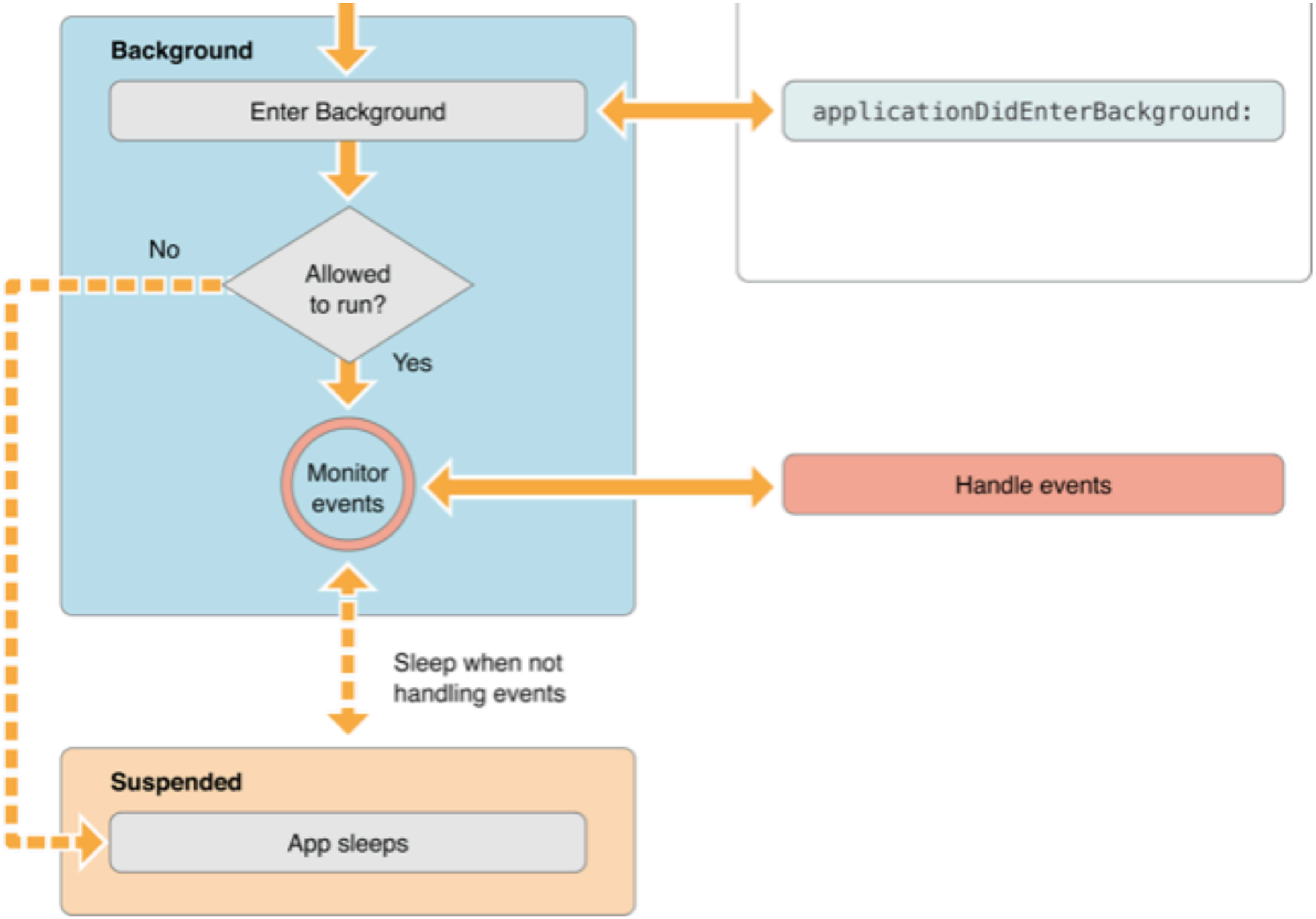
Launch Time


Launch Time
User taps app icon
main()
UIApplicationMain()
Load main UI file
First initialization
Restore UI state
Final initialization
Your Code
application:
willFinishLaunchingWithOptions:
Various methods
application:
didFinishLaunchingWithOptions:

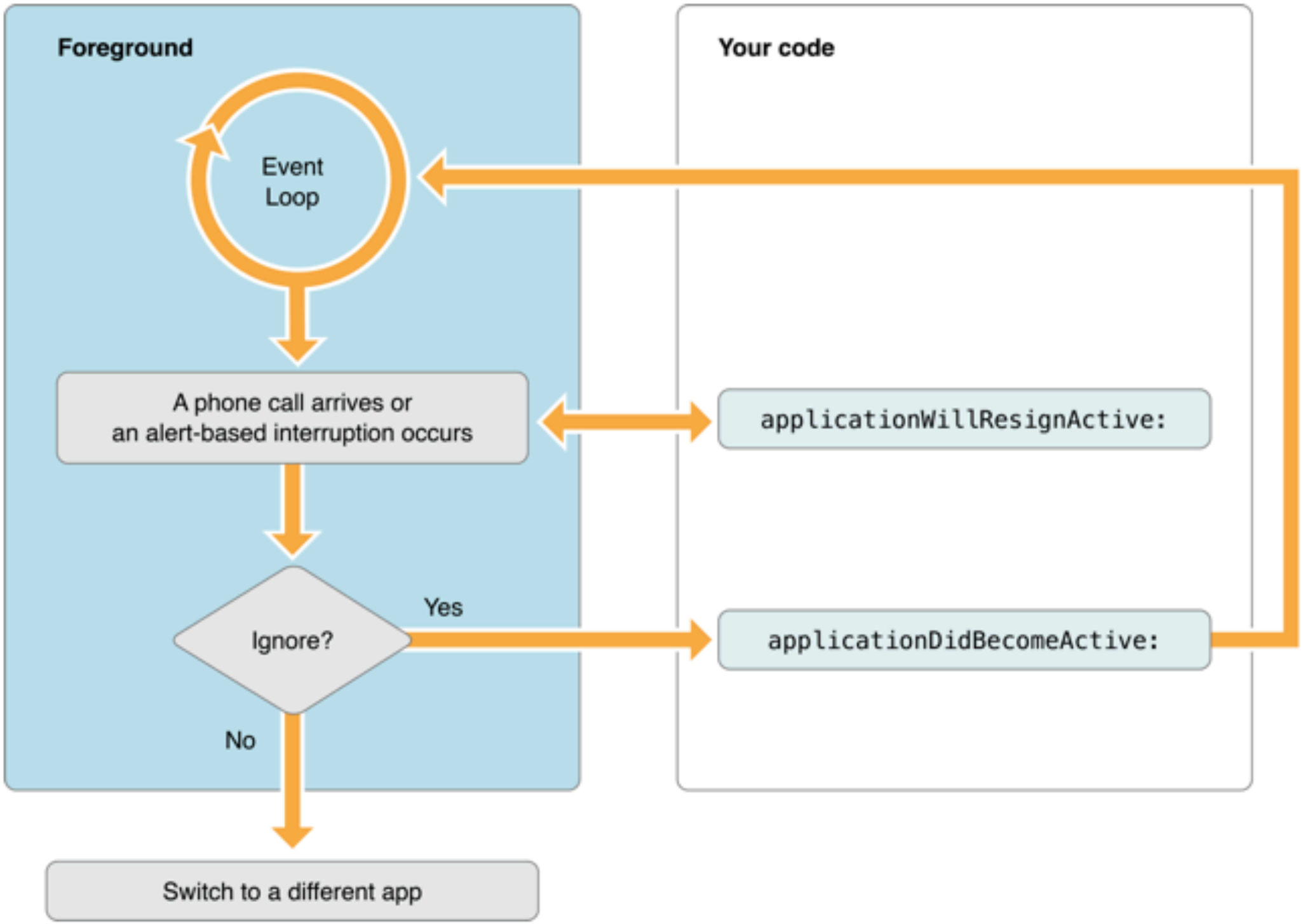
Into the foreground



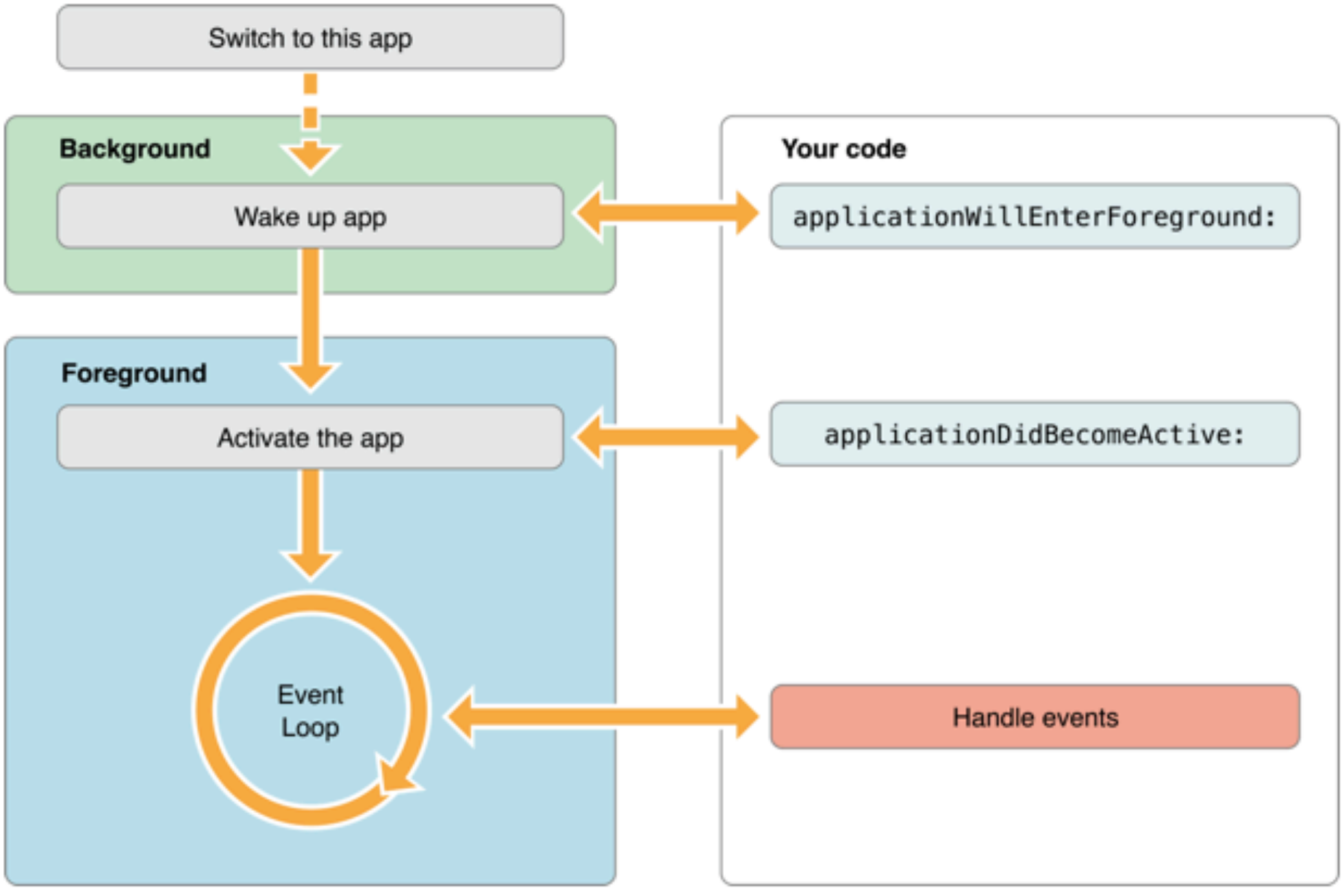
Into the background



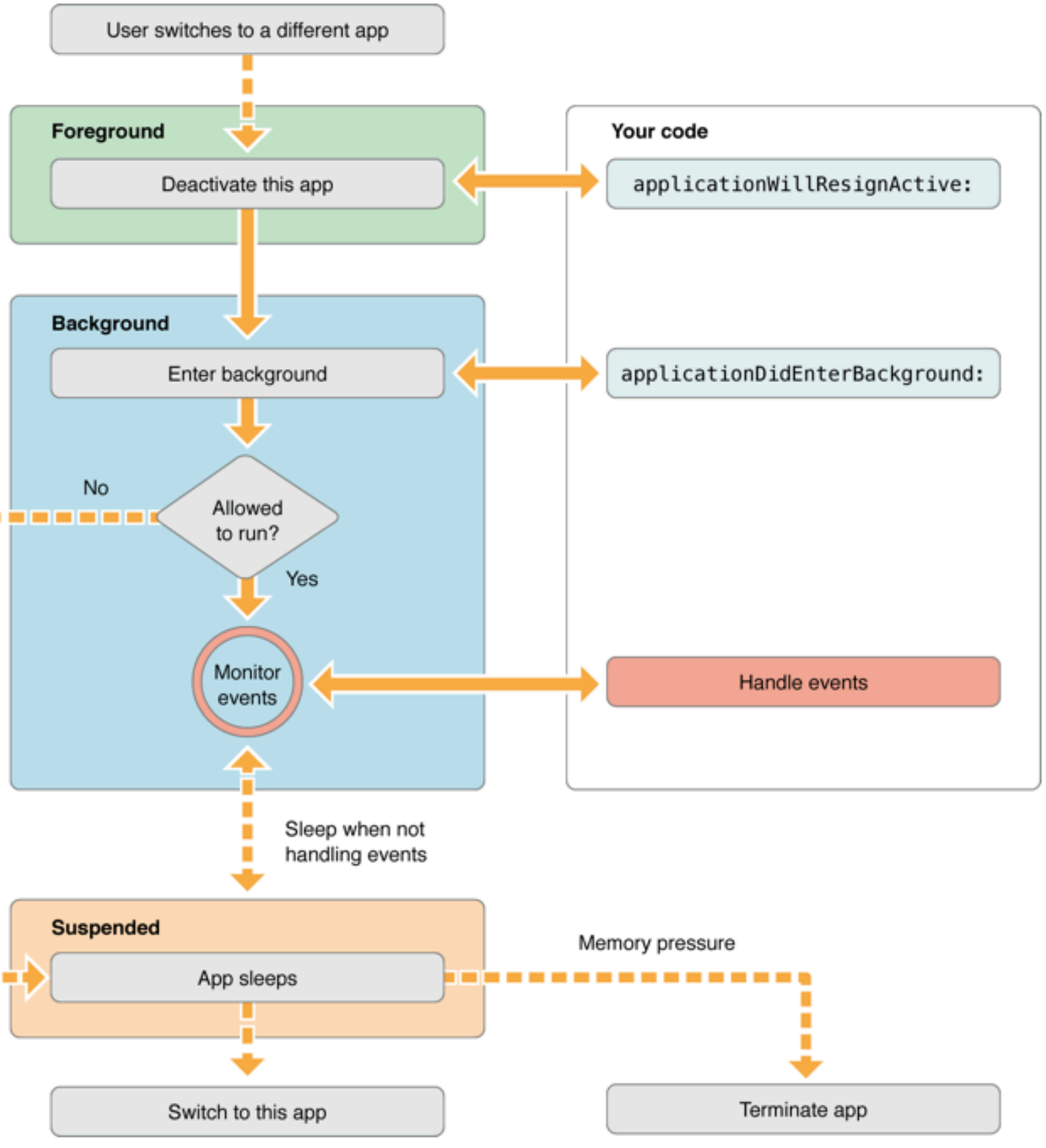
Handling alert-based interruptions



background to the foreground



The Background Transition Cycle



Background Modes for apps

Fast campus

Xcode background mode	UIBackgroundModes value	Description
Audio and AirPlay	`audio`	The app plays audible content to the user or records audio while in the background. (This content includes streaming audio or video content using AirPlay.) The user must grant permission for apps to use the microphone prior to the first use; for more information, see Supporting User Privacy.
Location updates	`location`	The app keeps users informed of their location, even while it is running in the background.
Voice over IP	`voip`	The app provides the ability for the user to make phone calls using an Internet connection.
Newsstand downloads	`newsstand-content`	The app is a Newsstand app that downloads and processes magazine or newspaper content in the background.
External accessory communication	`external-accessory`	The app works with a hardware accessory that needs to deliver updates on a regular schedule through the External Accessory framework.
Uses Bluetooth LE accessories	`bluetooth-central`	The app works with a Bluetooth accessory that needs to deliver updates on a regular schedule through the Core Bluetooth framework.
Acts as a Bluetooth LE accessory	`bluetooth-peripheral`	The app supports Bluetooth communication in peripheral mode through the Core Bluetooth framework. Using this mode requires user authorization; for more information, see Supporting User Privacy.
Background fetch	`fetch`	The app regularly downloads and processes small amounts of content from the network.
Remote notifications	`remote-notification`	The app wants to start downloading content when a push notification arrives. Use this notification to minimize the delay in showing content related to the push notification.

Question

- 생성한 프로젝트의 AppDelegate 파일에서 코드를 작성해보세요.
- 코드는 어떤 순서로 호출되어 동작하고 있을까요?
- 실행되지 않는 코드가 있다면 왜 그럴까요?

Not allowed at the top level

Fast campus

```
print("Hello, Swift!")
```

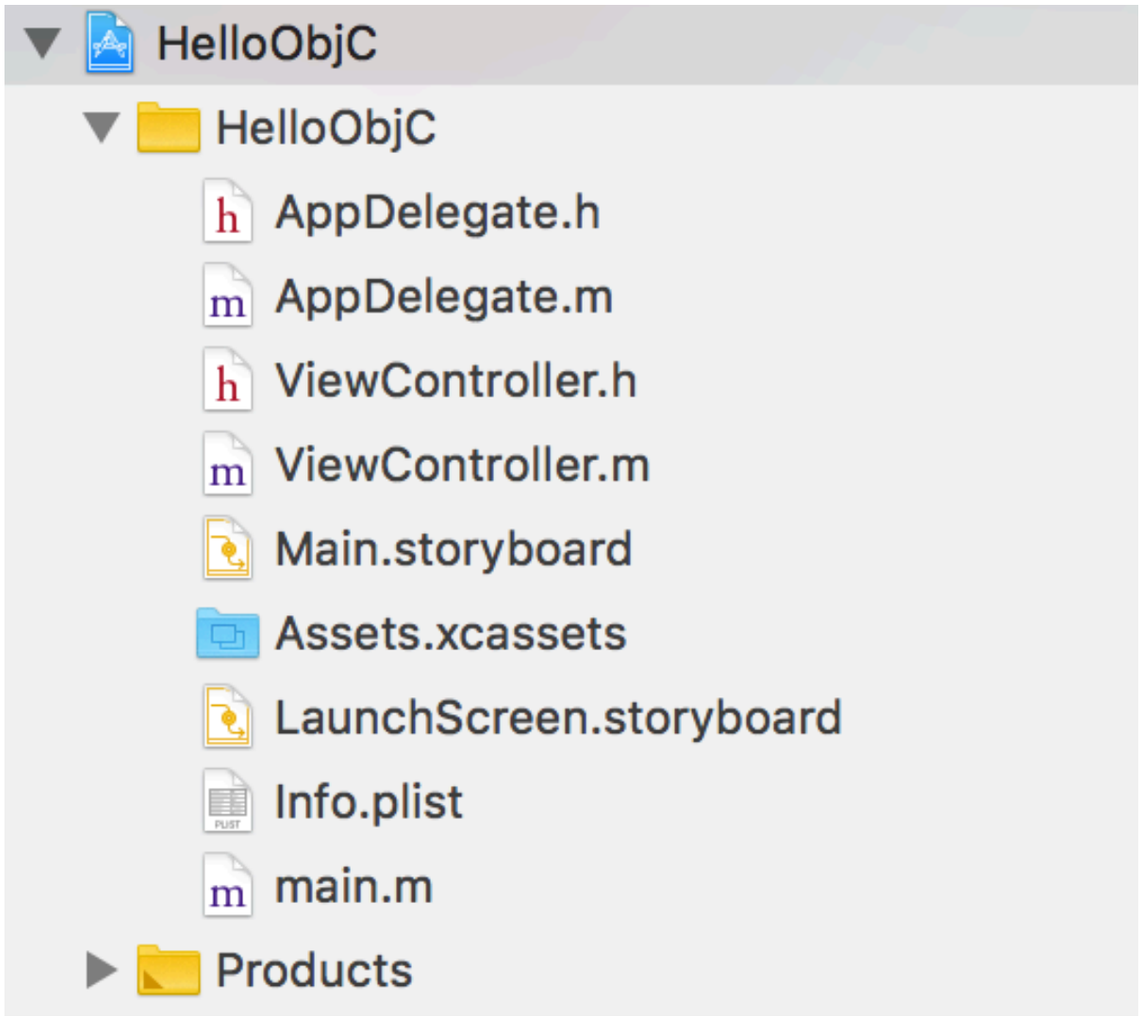
Expressions are not allowed at the top level

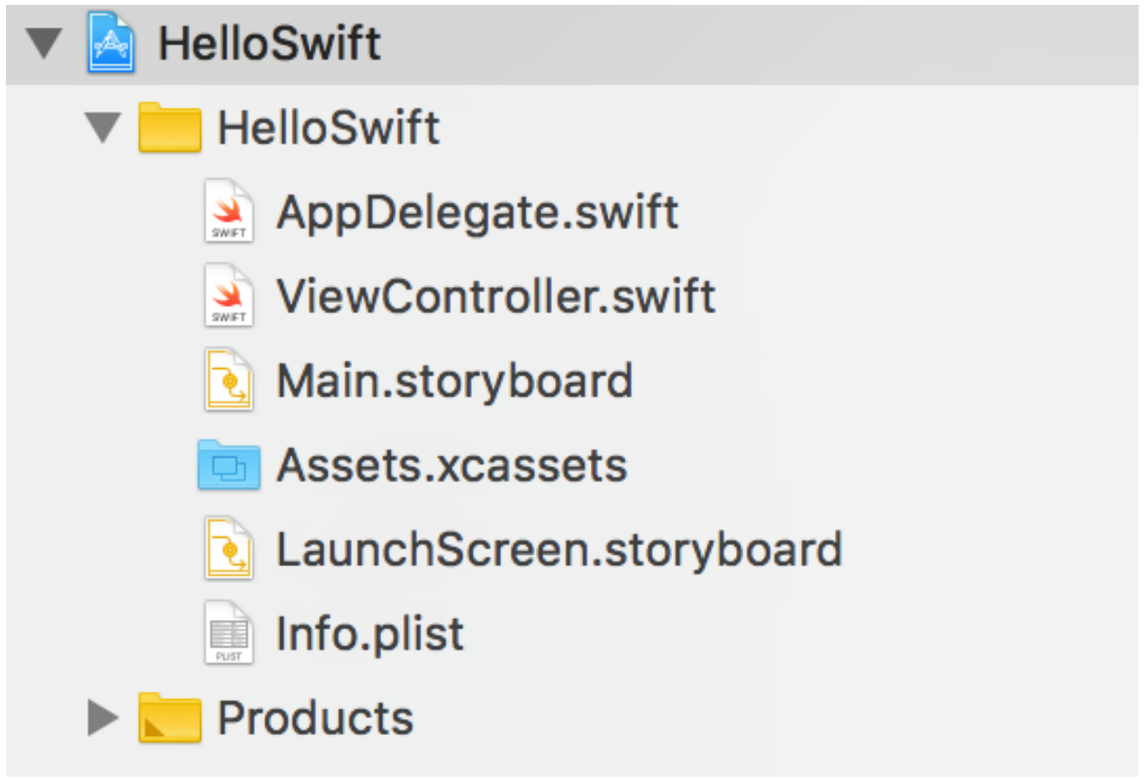
```
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

  var window: UIWindow?

}
```


```
print("Hello, Swift!")
```

Expressions are not allowed at the top level

Project files

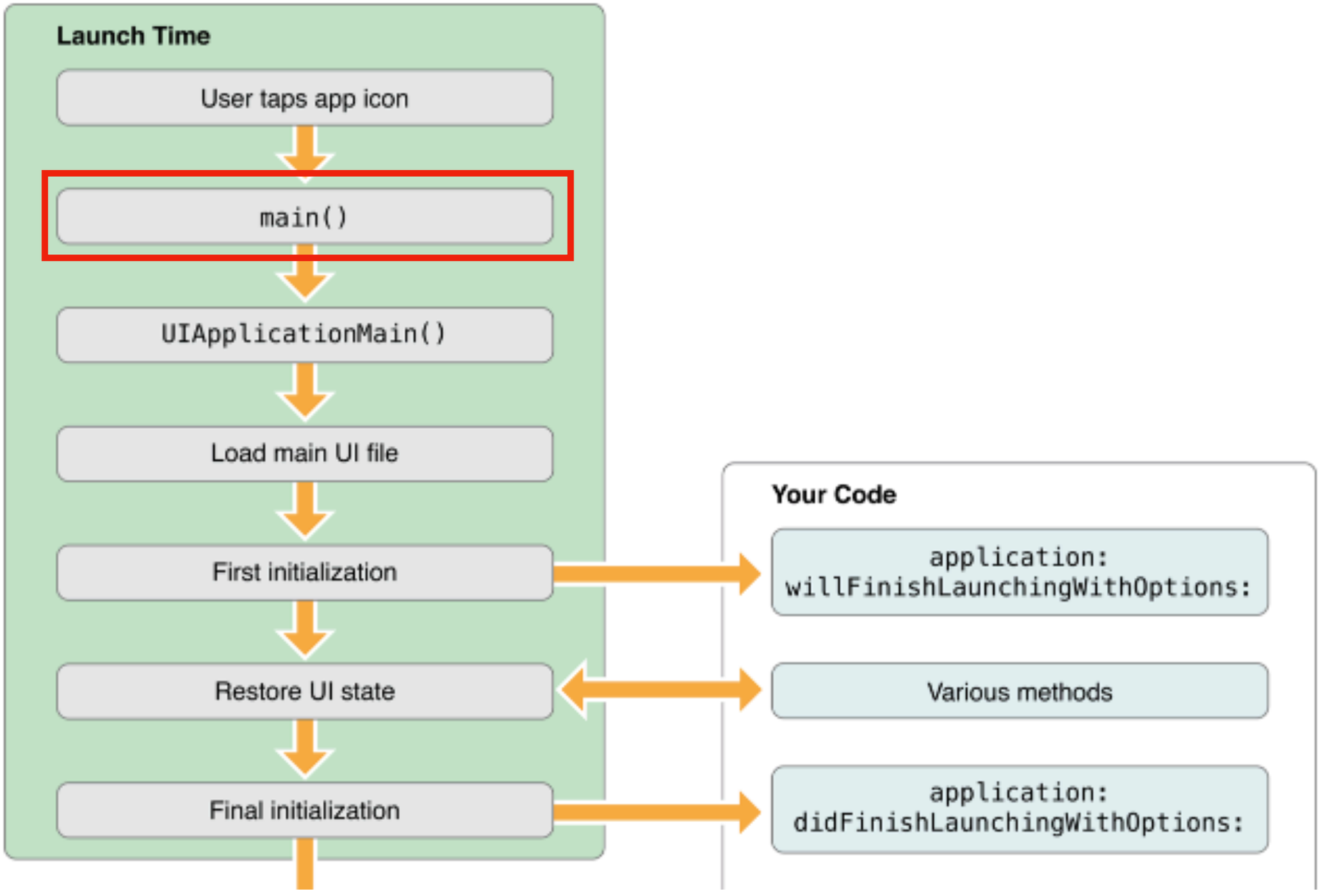




The Main Function (Obj-C)

```
#import <UIKit/UIKit.h>
#import "AppDelegate.h"

int main(int argc, char * argv[])
{
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([AppDelegate class]));
    }
}
```

Launch Time


Launch Time
User taps app icon
main()
UIApplicationMain()
Load main UI file
First initialization
Restore UI state
Final initialization
Your Code
application:
willFinishLaunchingWithOptions:
Various methods
application:
didFinishLaunchingWithOptions:

The Main Function (Swift)

Fast campus

```swift
import UIKit

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

  var window: UIWindow?

  // Code...
}
```

main.swift

Fast campus

The “main.swift” file can contain top-level code, and the order-dependent rules apply

```
import UIKit

UIApplicationMain(
  CommandLine.argc,
  UnsafeMutableRawPointer(CommandLine.unsafeArgv)
    .bindMemory(to: UnsafeMutablePointer<Int8>.self,
                capacity: Int(CommandLine.argc)),
  nil,
  NSStringFromClass(AppDelegate.self)
)
```