Chapter02. iOS앱의 구조

1. 앱이 실행되는 과정

1. main() 함수 실행 (해당 파일은 이제 확인 불가)

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


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

2. UIApplicationMain() 함수 호출 (@UIApplicataionMain)

```
import UIKit

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?


    func application(_ application: UIApplication,
didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        // Override point for customization after application launch.
        return true
    }
}
```

3. 앱의 본체에 해당하는 UIApplication 객체를 생성

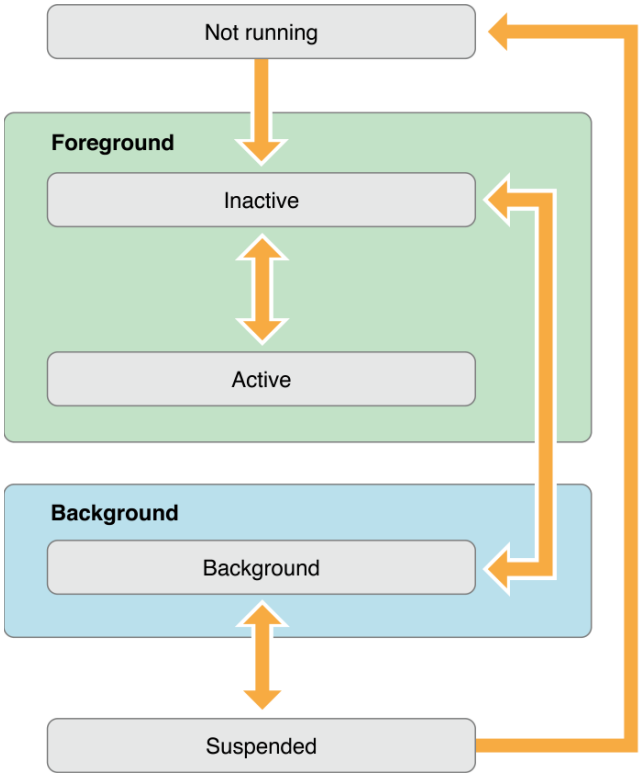
4. UIApplication 객체는 Info.plist파일을 바탕으로 앱에 필요한 데이터와 객체를 로드

5. AppDelegate 객체를 생성하고 UIApplication 객체와 연결

6. 이벤트 루프를 만드는 등 실행에 필요한 준비를 진행

7. 실행 완료 직전, 앱 델리게이트의 application(_:didFinishLaunchingWithOptions:) 메서드를 호출

2. 앱의 라이프 사이클(Life Cycle, 생명 주기)



application(_:willFinishLaunchingWithOptions)
- 앱이 구동되어 필요한 초기 실행 과정이 완료되기 직전에 호출되는 메서드

application(_:didFinishLaunchingWithOptions)
- 앱이 사용자에게 화면으로 표시되기 직전에 호출되는 메서드
- 앱이 실행된 후에 진행할 커스터마이징이나 초기화를 위한 코드를 여기에 작성

applicationDidBecomeActive(_:)
- 실행된 앱이 포그라운드에 표시될 때 호출되는 메서드

applicationWillResignActive(_:)
- activie -> inactive 상태로 바뀔때 호출되는 메서드

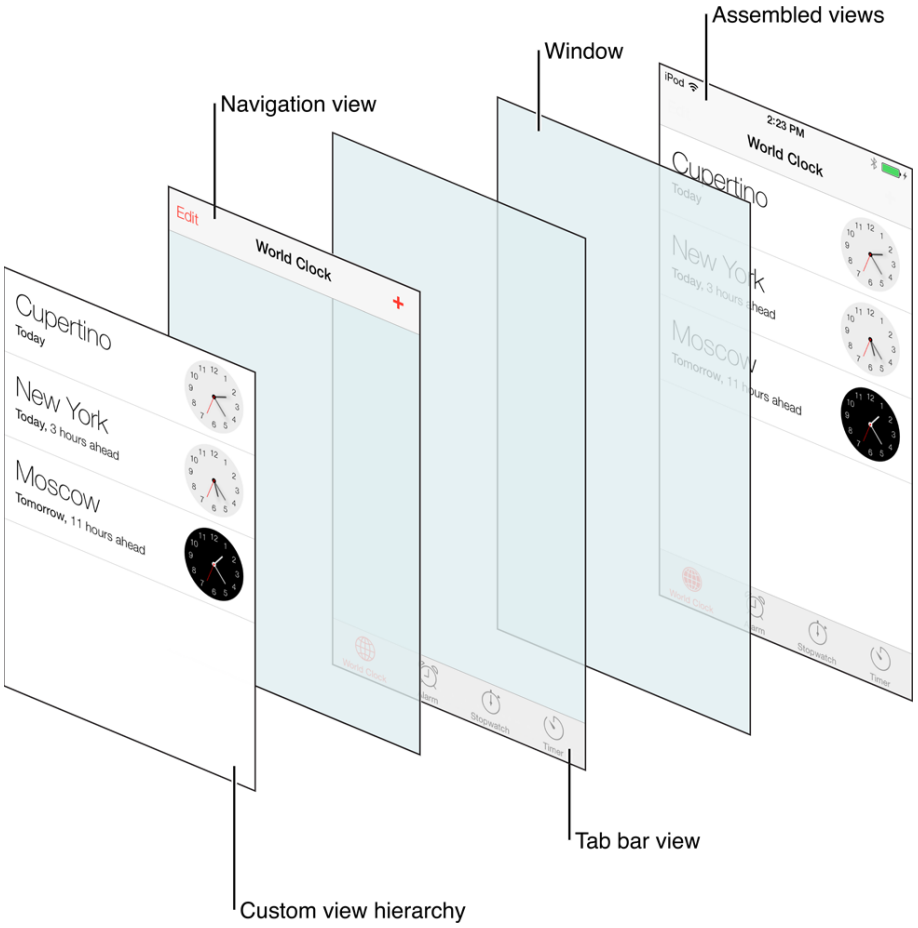
applicationDidEnterBackground(_:)
- 앱이 백그라운드 상태에 진입했을 때 호출되는 메서드

applicationWillEnterForeground:
- 앱이 백그라운드 상태에서 포그라운드로 진입했을 때 호출되는 메서드

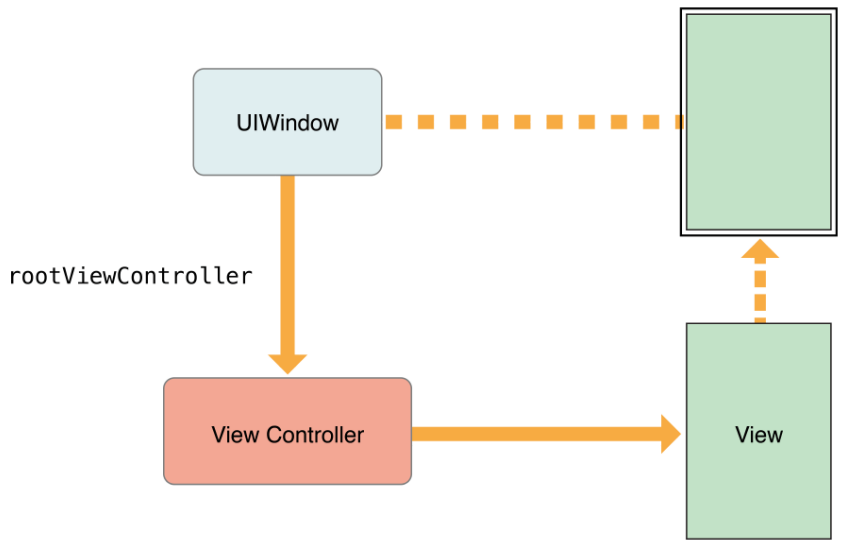
applicationWillTerminate(_:)

- 앱이 종료되기 직전에 호출되는 메서드

3. 앱의 인터페이스와 구성 요소



UIScreen : 기기에 연결되는 물리적인 화면을 정의하는 객체
UIWindow : 화면 그리기 지원 도구를 제공하는 객체
UIView : 그리기를 수행할 객체 세트



4. 뷰 컨트롤러

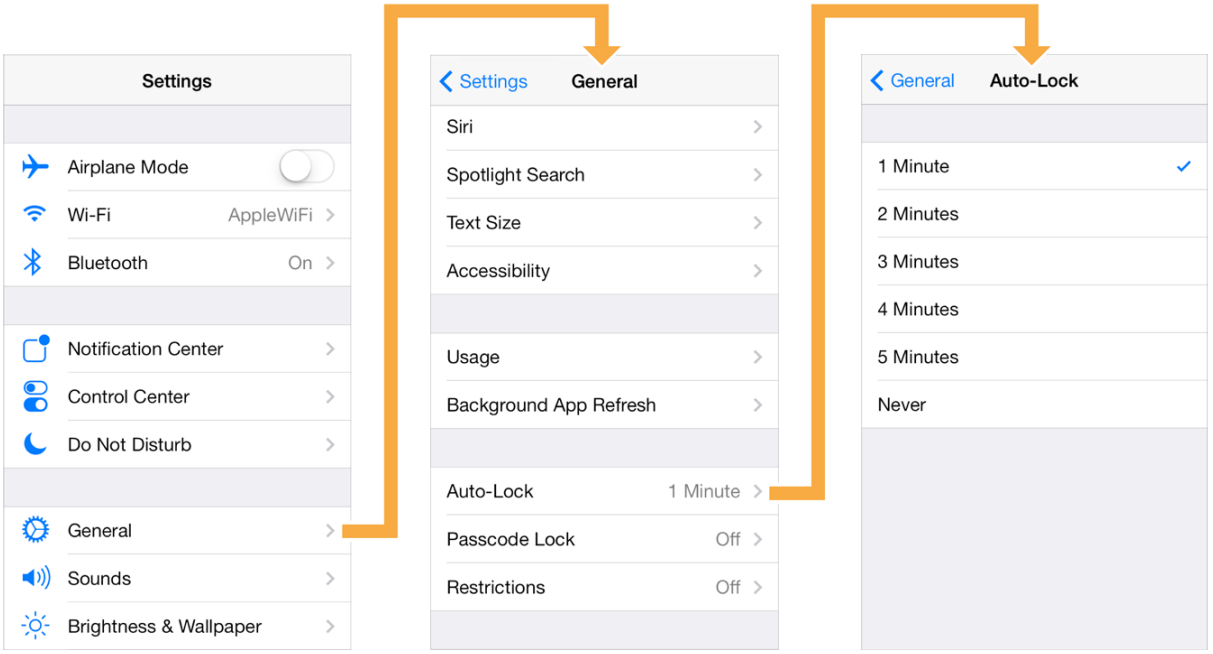
- 앱의 근간을 이루는 개체로, 모든 앱은 최소한 하나 이상의 뷰 컨트롤러로 구성됨

View Controller

- iOS에서 가장 기본이 되는 컨트롤러
- 내부에 뷰를 포함하므로 원하는 대로 화면을 직접 구성하고 컨트롤들을 배치할 수 있음.

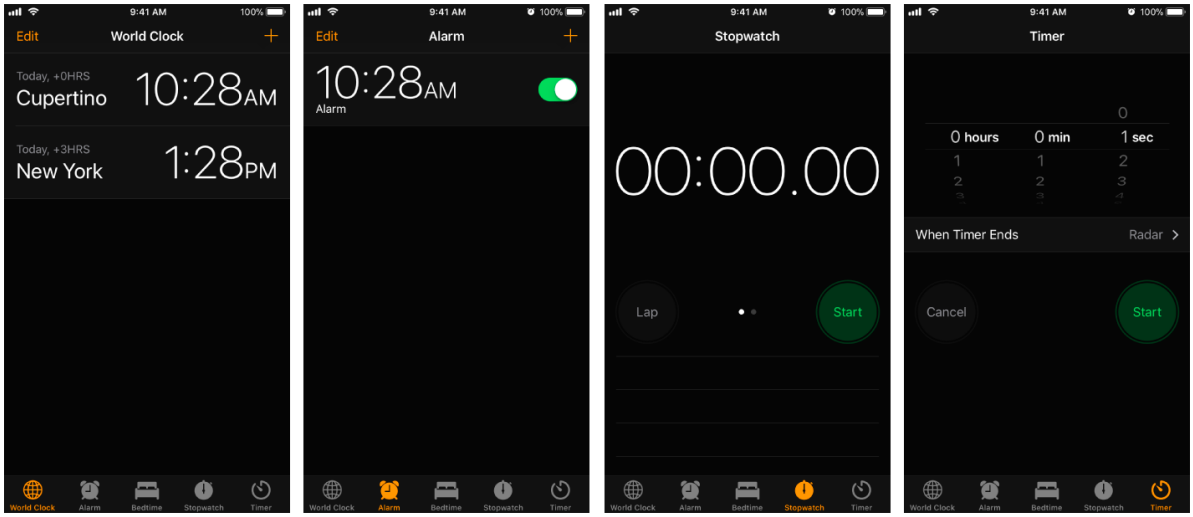
Navigation Controller

- 앱의 화면 이동에 대한 관리와 그에 연관된 처리를 담당해주는 컨트롤러

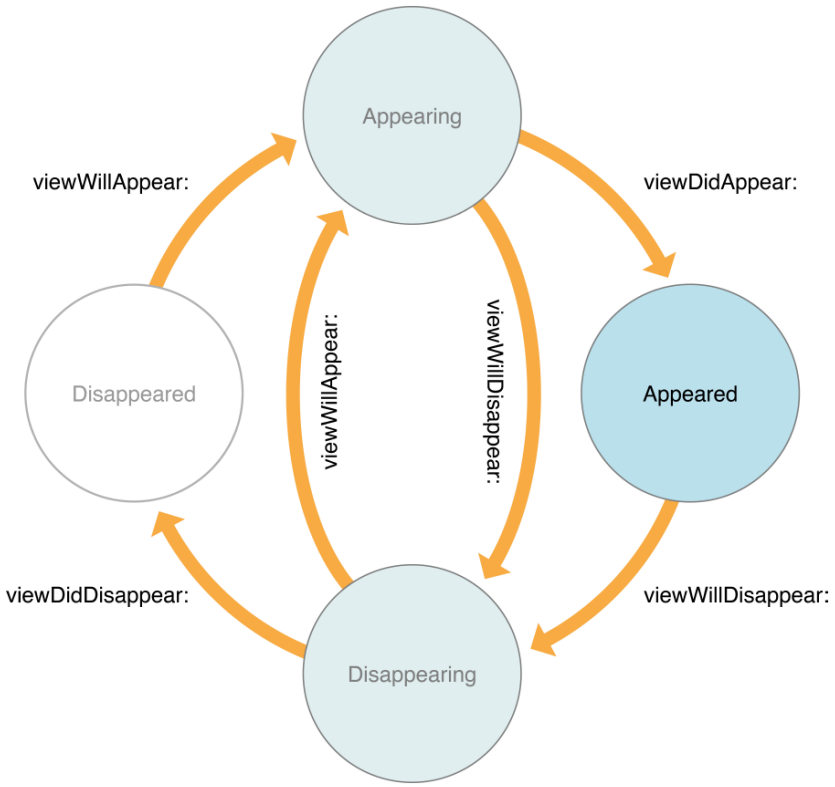


Tab Bar Controller

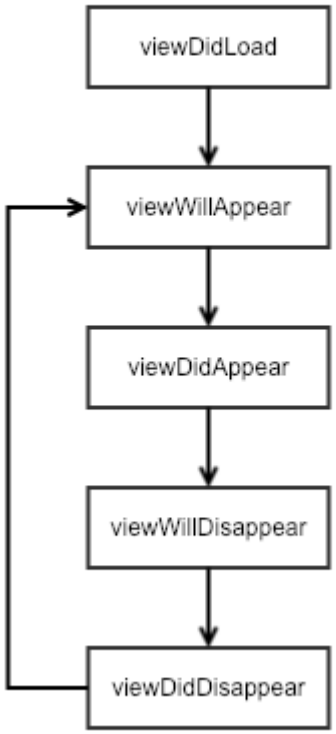
- 화면을 나타내는 여러 개의 탭이 있고, 탭을 터치하면 화면이 전환되는 형태의 앱을 만들고자 할 때 사용되는 컨트롤러



5. 뷰 컨트롤러의 라이프 사이클(Life Cycle, 생명 주기)



UIViewController Life Cycle



6. 로그 및 디버깅

	NSLog	print
TimeStamp	O	X
Only String	O (문자만 출력 가능)	X (자료형도 출력 가능)
Performance	X	O
Synchronization	O	X
Device Console	O	X

7. LLVM
po
Image lookup -address