뷰컨트롤러

뷰컨트롤러는 개발자가 개발한 앱의 내부구조의 기초이다 .
모든 앱은 최소 하나의 뷰컨트롤러를 가지고 있으며 거의 대부분의 앱은 여러개의 뷰컨트롤러를 가지고 있다 .
각각의 뷰컨트롤러는 개발자의 앱의 깔려있는 데이터와 인터페이스사이에 상호작용으로써 유저인터테이스의 부분들을 관리한다.
이런한 상호작용과 작용을 관리하는 것은 개발자가 만든 앱에서 매우 중요한 부분을 담당하고 있으며, 뷰컨트롤러들은 개발자가 할 거의 모든것들의 중심으로 자리 잡고 있다.
UIViewController class 는 개발자가 위에 올려놓은 뷰들을 관리하기 위해서 메서드와 프로퍼티로 정의되어 있으며 , 각종 사용자 이벤트를 다룬거나,또다른 뷰컨트롤러의 부터의 이동을 담당하거나, 개발자의 앱의 또다른 일부분을 조정하는 역할을 한다.
개발자는 UIViewController 를 subclass를 두거나 혹은 그위에 또다른 하나의 subclass를 둘수도 있고, 앱에서 필요한 기능이나 동작에 필요한 개발자의 커스텀 코드를 추가할 수도 있다 .

여기에는 두가지 뷰컨트롤러의 유형이 있다.

1. 컨텐츠 뷰컨트롤러는 개발자의 기능등의 한부분을 없앨 수도 있으며 또한 뷰컨트롤러의 주된기능은 삭제되기전에 있던 기능들을 생성하는 것이다.
2. Container ViewController 는 또다른 뷰컨트롤러(차일드 뷰컨트트롤러라고 알려진)들로부터의 정보를 가지고 있다. 그리고 navigation 또는 present방식으로 그 뷰컨트롤러를 다르게 화면에 보여준다.

대부분의 앱들은 2가지 유형이 혼합된 형태이다 .

뷰의 관리

뷰컨트롤러의 가장 중요한 역할은 뷰의 계층을 관리 하기 위한 것이다. 모든 뷰컨트롤러들은 하나의 뷰컨트롤러의 기능의 모든것을 위에 올릴 수 있는 루트뷰를 가지고 있는데 , 루트뷰의가 있으므로써 개발자는 원하는 내용 루트뷰에 올려서 사용자의 화면에 나타낼 수 있다.

뷰컨트롤러의 기능은 자기자신 뷰컨트롤러에 의해서 모든뷰들을 관리하는 것이다. 컨테이너 뷰컨트롤러는 하나또는 그이상의 차일드뷰컨트롤러으로부터 루트뷰위에 추가한 자신의 뷰들을 관리한다.
컨테이너는 차일드 뷰의 기능까지는 관리하지못한다.
이것은 오직 루트뷰만 관리 할 수 있으며 크기나 뷰를 대체하는 것은 컨테이너뷰컨트롤러의 디자인에 따라서 결정되어진다.

데이터 관리

한 뷰컨트롤러가 관리하는 뷰들과 앱의 데이터 사이의 중개역할을 한다. UIViewController의 메서드들이나 프로퍼티들은 개발자의 앱에서 시각적으로 보여지는 부분을 관리하도록 한다.
개발자가 UIViewController를 상속받을때상속받은것은 거의 변수들은 추가하거나, 상속받은 뷰컨트롤러에서 데이터는 관리하기위해 필요할 것이다.

개발자가 만든 변수들은 그 뷰컨트롤러가 데이터 참조주소를 가지거나, 그데이터를 보여줄 뷰들에 관계를 생성한다. 데이터를 뒤쪽으로나 앞쪽으로 이동하게하는것이 뷰컨트롤러의 역할이기도 하다.

UIDocument객체는 데이터를 뷰컨트롤러들로부터 개별적으로 관리 할 수 있는 방법 중 하나인데, 이 객체는 하나의 뷰컨트롤러 객체로써 어떻게 영구저장한것을 읽고,쓰고하는 방법을 알고 있다. 개발자가 상속을 할때 어떤 뷰컨트롤러에서 데이터를 통과시키거나 뽑아내는데 로직이나 메서드를 추가하여 데이터를 가공하고. 그뷰컨트롤러는 아마도 뷰들을 보다 쉽게 반영할 수있도록 데이터들을 복사하여 저장할 것이다. 그러나 document객체는 여전히 원래의 데이터를 가지고 있을 것이다.

유저작용

뷰컨트롤러는 responder object이면서 responder chain 을 받은 이벤트를 핸들하는 가능하다. 비록 그것들을 하기위해선, 뷰컨트롤러는 드물게 즉각적으로 터치이벤트를 다루어야한다. 게다가 뷰들은 보통 뷰자신이 터치를 받거나 그결과를 델리게이트로 연견되어지거나 해당 타겟 객체를 알려주기도 한다. 이것은 보통 뷰컨트롤이 하는 일이다. 뷰컨트롤러의 거의 대부분의 이벤트는 델리게이트머서드나 액션메서드를 통해서 관리되어진다.

추가적인것은 :액션메서드
https://developer.apple.com/library/content/featuredarticles/ViewControllerPGforiPhoneOS/DefiningYourSubclass.html#//apple_ref/doc/uid/TP40007457-CH7-SW11
을 찾아보면된다.
핸들링하는것은
https://developer.apple.com/library/content/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/Introduction/Introduction.html#//apple_ref/doc/ui

자원관리
한 뷰 콘트롤러 뷰에 대한 모든 책임과 그것이 만드는 객체를 가정한다. UIViewController class는 자동적으로 뷰 관리에 거의모든모든부분을 다루는데 예들들어서 UIkit은 자동적으로 그어떤 뷰와 관련된 더이상필요하지않은 자원에 대해서 자동적으로 메모리를 할당 해제한다. UIViewContoler class를 상속받은 뷰컨트롤러에서 개발자는 명시적으로 개발자가 생성한 어떠한 객체를 관리하는것에 대해서 담당한다.

이용할 수 있는 메모리가 줄어들때 UIkit은 앱에게 더이상 필요하지않은 자원을 할당해제하라고 요청한다. 이것을 부르는 것에 대한 한 방법으로는 didReceiveMemoryWarning이라는 뷰컨트롤로내부에 메서드가 있는데 이메서드는 더이상 필요하지 않거나 나중에 쉽게 생성되어질 수있는 객체들의 주소들을 지울 수 있는 메서드이다 . 예를 들어서 개발자가 캐쉬된 데이터를 지우기 위해서 사용한다. 이것은 메모리 할당하는것은 메모리의 상태를 개선하는 부분에서 굉장히 중요한일 이다.
앱들은 메모리가 너무 많이 소모되면 이것은 자동복구 시스템에 의해서 엡이 완전히 죽어버릴 수도 있다.

Adaptivity

뷰컨트롤러들은 위에 올라오는뷰들이 보여지는것에 대해서 내부에 환경에 맞춰서 책임을 져야한다. 모든 IOS앱은 IPad와 여러가지 사이즈아이폰에서 동작이가능할 것이다. 다른 뷰컨트롤러들과 뷰의 계층들이 각기다른 디바이스에서 다르게 제공하는 것보다는 이것은 변화된 요구상항에대한 뷰들을 싱글뷰컨트롤러에서 더욱 간단하게 될 수 있다.
ios 에서 뷰컨트롤러는 조잡한 변화와 잘 조합된 변화를 다루는것이 필요하다. 조잡한 변호는 뷰컨트롤러특성이변화 할때 발생할 수 있다. 변형는 디스플레이 규모의 전반적인 환경에 대한 특성을 말한다. 가장 중요한 변형은 두가지는 뷰컨트로럴가 수직과 수평으로 사이즈가 변경될 때이다. 이것은 얼마나많이 주어진 치수에대해서 공간을 차지할지 인지한다. 개발자는 뷰들의 레이아웃으 변경하는 것에대해서 size class를 이용해서 변화를 줄 수 있다.