IOS 뷰와 윈도우의 구조

일단 ios에서 모든 보여지는 컴포넌트들은 뷰단위로 이루어진 다고 볼 수있다 .
또한 애플리케이션에서 유저의 반응과 반을 다루게 되는 것이며 유저의 터치나 여러가지는 동작을 뷰가 처음 받게 된다고 볼 수 있다 .
UIkit과 다른 시스템프레임워크는 개발자가 약간의 변경이나 그아예변경을 안해도 사용할 수 있도록 제공한다.
하지만 개발자 스스로 기본적으로 제공되는 뷰가 아니라 자신만의 뷰들을 사용하고자 할때 이 또한 개발자입장에서 만들 수 있도록 허락하고 있다 .

개발자가 시스템에서 기본적으로 제공하는 뷰를 사용할 지는 몰라도 UIView 와 UIWindow의해서 제공되어진 이 뷰의 하부구조를 이해하고 있어야 한다 .
이 두클래스들은 매우복잡한 기능들인데 왜나면 여기엔 레이아웃과 뷰들이 보여질때의 모습을 관리하여야 하기 때문이다.
이기능들이 어떻게 동작하는지 이해하는것은 개발자가 만든 뷰가 적절하게 애플리케이션 내에서 반응하도록하는데 매우 중요한 부분이다 .

뷰 구조의 기초들
개발자가 원하는 시각전인 모든것들은 대부분 뷰객체 또는 인스턴스로 이루어져 있다.(UIView Class) 이 뷰 객체는 사각형의 영역으로 되어있는데 화면위에 그려지며, 또한 사용자의 터치 를 그지역의 에서 받을 수 있다.
뷰는 자기자신뿐만아니라 다른 뷰의 부모가 될 수도 있고, 그 위치에서 위에 올려지있는 뷰의 사이즈를 조정 할 수도 있다.

그 UIView클래스는 대부분 동작될때 다른뷰 사이에서 부모뷰 로써 위에 올려진 다른뷰들을 관리한다 .
하지만 개발자는 이또한 기본적이 뷰모뷰와 그 위엥 올려진 자식뷰들의 기본적인 상호작용를 커스터마이징을 할 수도 있다 .

뷰들은 코어애니메이션 계층 결합되어 있는데 이것은 뷰들의 보여질때 애니메이션과 렌더링 떄문이다 .

코어애니메이션 객체를 사용하는것은 성능면에서 중요한 영향을 미친다 . 가급적이면 뷰의 객체에서 그리는코드는 불려지고 또한
코드가 불려질때 코어애니메이션의해서 결과가 저장장치에 남겨 지고 후에 다시 사용 되어지기도 한다.
이미 렌더링되어진것을 재사용한다는것은 그림을그리는 주기에서 보면 새롭게 갱신된 뷰들이 있을때 발생한다 .
이 뷰의 재사용은 존재하는 뷰가 컨트롤되어지는 곳에 애니메이션들이 발생하는 동안 매우 중요하다.
그러므로 재사용은 새로운 뷰를 만드는것보다 메모리적으로 덜 부담스럽다.

뷰의 계층구조들과 하위뷰 관리

뷰는 자신의 뷰를 보여주기위해서 다른 뷰들의 컨테이너의 역할을 하기도 한다. 한 뷰가 여러개의 뷰들을 포함 할 수 있고 이럴때 하나의 뷰와 여러개의 뷰는 부모뷰와 자식뷰라는 관계가 생긴다. 자식뷰는 자기가 하위뷰라는 것도 알고 있으며 부모뷰는 자기가 슈퍼뷰라는 것을 알고 있는 상태가 된다 .
이러한 관계가 생성되는것은 개발자의 애플리케이션에서 앱의 행동과 시각적인 모습 둘다 내포하고 있다 .

시각적으로 자식뷰들은 부모뷰의 모든부분이나 아니면 약간에서 왜곡을 보이는데 예를 들면 하위뷰는 전체적으로 불투명하다면 그 영역은 하위뷰가 부모뷰가 그 아래에서 영역을 차지하고 있더라도 자식뷰가 가리게 되어진다. 만약 그 자식뷰가 부분적으로 투명하다면 자식뷰와 부모뷰 겹쳐진부분은 화면에 섞여서 보여지게 된다. 각각 부모뷰는 그들의 자식뷰들을 배열이나 그 저장하는데 순서에 따라 각각의 자식뷰의 보여지는것에 대한 배열로써 가지고 있게 된다. 만약 두개의 자식뷰가 겹쳐진다면 서로 그것은 마지막에 추가된거나 마지막으로 배열에 마지막으로 움직인것 이 나타나게 된다 .

이 부모뷰와 자식뷰 관계는 여러부분에서 뷰들에게 영향을 끼친다.
부모뷰의 크기를 변경하는 것은 연쇄효과로 다른 자식뷰의 위치나 크기의 변화 굉장히 많은 영향을 끼친다.
개발자가 부모뷰의 크기를 바꿀때 개발자는 부모뷰 위에 올라와 있는 자식뷰의 크기가 적절하게 구성 되는것에 대해서 제어를 해주어야만 한다 .
이 부분의 변화들은 부모뷰의 알파값이 변화하는것 또는 부모뷰의 좌표계에 수학적인 변환을 적용하는것에도 포함한다.

뷰의 계층에서 뷰들의 정렬 또한 어떻게 개발자가 만드는 애플리케이션의 이벤트의대해서 반응하느냐에 따라서 결정되어진다.
화면에 나타난 뷰를 사용자가 터치를 했을때 시스템은 이 터치에 대한 객체와 정보를 터치한 뷰에게 즉각적으로 보낸다 . 그러나 만약 뷰가 그 부분적인 터치이벤트에 대해서 다루지 않는다면 그터치이벤트는 그 뷰를 통과하여 그아래에 있는 뷰에게 터치에대한 객체와 정보를 전달하게 되고, 이것을 레스폰더 체인이라고 한다. 특정하게 뷰컨트롤러사이에서 정해놓은 뷰들은 이러한 레스폰터객체를 통과시킬 수있다. 만약 객체가 그 이벤트를 다루지 않느다면, 그터치이벤트는 이 애플리케이션의 객체에 다다를 것이고 그것은 일반적으로 버리는것과 같다.
자세한정보는 어떻게 뷰의 계층구조가 생성되는지 대한정보 를 보면알 수 있는데
https://developer.apple.com/library/content/documentation/WindowsViews/Conceptual/ViewPG_iPhoneOS/CreatingViews/CreatingViews.html#//apple_ref/doc/uid/TP40009503-CH5-SW47
여기들어가면 알 수가 있다 .

뷰의 그림그려지는 주기

UIView 클래스는 보여지는 뷰에 대해서 그림을 그리는 모델의 요청에 의해서 사용된다.
뷰가 첫번째로 화면에 나타나게 될 때 시스템은 뷰에게 그리라 고 요청한다. 시스템은 이것을 뷰의 시각적인 표현에대해서 스냅샷 형태로 찍는다. 만약 그뷰들의 내용이 변화가 아예 없다면 그뷰들의 그리는 코드는 아마도 다시 불리지 않는 것이다.
이 스냅샷이미지는 대부분의 뷰들의 동작을 포함하는곳에서 재사용되어지는데
뷰에 변화할게 있다면 개발자는 뷰에 변화가있다고 시스템에게 알려주어야 한다. 그러면 뷰 다시 그 뷰에 그리는 과정을 다시 실행 할 것이고 이 결과 또한 시스템에서 스냅샷 형태 다시 만들어 지게 된다 .

개발자가 만든 뷰의 내용이 변화할때, 개발자는 다시그릴 수없어 즉각적으로 게다가 개발자는 이 뷰의 setNeedDisplay 또는 setNeddDisplayInRect: 메서드를 사용해서 무효화시켜야한다.
이 메서드들은 시스템 다음에 그려질때 필요하거나 변화된 뷰의 내용을 시스템에게 알려주는 것인데, 시스템에서 다음 그려지는동작을 기다릴때 여러개의 뷰의 무효화 하거나 추가,또는 삭제를 개발자의 뷰의 계층구조에서 할 수 있도록 한다. 또한 뷰를 숨기는것, 리사이징하는것 그리고 한꺼번에 뷰들을 저장하는것 에대해서 말이다 . 개발자가 만든 모든변화는 동시에 처리되어진다.

같은시간에 뷰들의 내용이 렌더링된다면 , 실질적인 그리는 프로세스는 그뷰와 배치에 따라서 다르게 그려질 것이다 .
개발자가 뷰의 실직적인 모습을 구성하는데 사용할 수있는 인터페이스를 나타낸다.
개발자가 서브클래스로 만든 커스텀UIView에 대해서 개발자는 일반적으로 그뷰의 drawRect:메서드를 사용해서 오버라이드를 하거나, 개발자가 만든뷰의 내용을 그릴 메서드를 사용해서 서브래클래스를 만들어 커스텀을 하게 된다 .

이것에대한 또다른 방법은 뷰의 내용을 그리는 다른방법으로써 제공하는데 이러한 메서드로의 설정은 직접적인 내부계층을 동작하게 하는 것이다. 그러나 drawRect:메서드는 가장일반적인 방법이다.

https://developer.apple.com/reference/uikit/uiview?language=objc
요거공부