`UIViewController` 클래스를 상속받은 클래스에서 부모의 Life Cycle Method(예: `viewDidLoad()`, `viewWillAppear(animated:)` 등)를 호출(`super.viewDidLoad()` 등)하는 이유에 대해 토의해 보고, 부모클래스의 메서드를 호출하지 않으면 발생할 수 있는 일에 대해 탐구해보세요.

토의내용

1. Gamja
    a. UIViewController의 원형 Life Cycle Method는 특별한 일을 하지는 않는 것 같다, 만약 Life Cycle Method 안에서 실행되어야 할 코드가 있다면, Life Cycle Method를 호출하기 직전에 실행시켜 버리면 될 것이기 때문에, 아마도 원형 Life Cycle Method는 빈 함수(Empty Function)일 것으로 추정된다.

    b. 그럼에도 Apple 공식 문서에서 호출을 해주어야 한다고 말하는 것은, 개발자가 Controller 간 상속 관계를 만들어 사용할 때, 상속된 부모 Controller Class의 초기화가 필요하기 때문이 Controller의 재사용성을 위한 것이 아닐까 싶다.
    (하지만, Controller 간의 상속 관계는 변화에 대응하기 힘들어 질 가능성이 높다는 점에서 Controller 간의 상속 관계를 남발하는 것은 좋지 않을 것 같다.)

2. NOIS
    a. UIViewController를 상속받은 ParentViewController. ParentViewController를 상속받은 ChildViewController가 있을 경우, Parent의 viewDidLoad 등에서 setup code가 있을 때, Child에서 super.viewDidLoad()를 호출하지 않으면 문제가 될것 같다. (예를 들어, parent에서 버튼 초기화를 하였는데 child에서 super.viewDidLoad를 호출하지 않으면 버튼이 초기화되지 않아 문제가 발생할 가능성이 높다.)

    b. viewDidLoad()와 같은 함수를 제거하여도 컴파일 시 Warning이나 Error 발생하지 않았다. 하지만 Life Cycle method의 내부 구현을 우리가 알 수 없고, Apple이 언제 이를 변경할 지 모르므로, super의 method를 override할 때에는 super.method()의 형식으로 쓰는게 객체지향 설계상 맞을 것이라고 생각한다.

3. Brix
    a. **loadView()**
        i. ViewDidLoad()메서드 보다 우선 실행된다.

b. **viewDidLoad()**
    i. ViewController 클래스가 생성될 때 실행된다. Low Memory와 같은 특별한 경우가 아니면 한번만 실행된다. 주로 초기화 할 때 사용하는 메서드이다.
c. **viewWillAppear()**
    i. 뷰가 화면에 나타나기 직전에 실행된다. 뷰 전환간에도 해당 ViewController 화면이 나타나기 직전에 실행되기 때문에 여러번 호출될 수 있다.
d. **viewDidAppear()**
    i. 뷰가 화면에 나타난 직 후 실행된다. 애니메이션 또는 API로 부터 정보를 받아와 화면에 업데이트 할 경우 이 메서드에 구현하는게 좋다.
e. **viewWillDisappear()**
    i. 뷰가 사라지기 직전에 실행된다.
f. **viewDidDisappear()**
    i. 뷰가 사라진 직 후 실행된다.

4. 성준성준
    a. viewDidLoad에서 정확히 무슨 작업을 하는지는 알 수 없지만 내부적으로 뷰를 셋업하는 과정에서 UIViewController가 어떤 작업을 수행하기 때문에 습관적으로라도 UIViewController를 상속받아 쓰는 경우라면 super.viewDidLoad를 호출하는 것이 미래를 보았을 때 안전하다고 생각한다.
    b. UIViewController에 있는 viewDidLoad가 쓰잘데기없는 function이라고 생각 하지만 객체지향적인 관점에서 봤을 때 디자인적으로 상속해주는 것이 깔끔하다고 생각한다.
5. BLU
   부모 UIViewController 와 ViewController 간 적용된 디자인 패턴은 템플레이트 패턴으로 이용한 올바른 패턴을 사용하는것이 좋습니다. iPhone의 바이너리가 빌드된 SDK를 감지하고 이를 기반으로 일부 OS동작을 수정한다는 것이기 때문입니다. UIViewController의 하위 클래스에서 해당 메서드 viewDidLoad 를 재정의하는 경우 발생하는 문제는 슈퍼 클래스 메서드를 호출하여 수퍼 클래스에서 설정 한 일부 항목을 설정해야 합니다. 만약 Apple이 코드를 변경하고 전화하지 않으면 앱이 다운되거나 제대로 작동하지 않을 수도 있습니다.

토의결과

- 부모의 변경에 대해 자식이 문제가 발생하지 않도록 super.viewDidLoad()와 같은 life cycle method를 호출하여야 한다.

- 애플 공식 문서에서는 몇몇 Life Cycle Method에 대해서는 다음 문구를 통해 반드시 super method 호출을 강요한다.

You can override this method to release any additional memory used by your view controller. If you do, your implementation of this method must call the super implementation at some point.

ex) didReceiveMemoryWarning, viewWillAppear, viewWillDisappear, viewDidAppear, viewDidDisappear