등장배경
-getter/setter 를 없애기위해서

왜? 변수들 마다 getter와setter를 만들다 보니 코드가 길어져 가독성이 불편하고 일일이 다 만들어주기에 귀찮았던 것. 그래서 이 귀찮은일을 컴파일러가 대신해주도록 한 것이다.

일단 Property는 @interface 와 @end 사이에 정의된다.

@property id name;

프로퍼티 Setter 사용여부 설정

- readwrite
setter/getter 모두 생성 (default)

- readonly
getter만 생성

Setter 함수를 위한 속성

Setter는 외부의 값을 저장하는 메소드이다. Objective-C는 레퍼런스 카운트에 따라서 객체의 라이프 사이클을 결정하기 때문에 중요한 옵션이다.

- strong
입력되는 변수의 소유권을 가져온다. referece count를 1증가시켜서
내가 소유하고 있는 동안에는 release되지 못하도록 한다.
이전의 retail 옵션과 동일한 기능, iOS 5.0이상 부터는 strong으로
변경되었다.

- retain
strong과 같이 소유권을 가져온다는 의미해서 러퍼런스 카우트를 증가
시킨다.

- week
week은 strong과 갈이 iOS5 이상 OS X v10.7에서 추가된 것으로
객체의 레퍼런스를 증가시키지 않는다. 따라서 언제가 객체가 릴리즈
되어서 메모리에서 사라질 수 있다. 이렇게 사라지면 자동으로 nil로
설정된다. (이전 버전에는 같은 목적으로 assign을 사용했다.)

- copy
주어진 객체를 복사하는 것으로 strong의 경우 같은 객체를 레퍼런스
카운드를 사용해서 소유권을 가져오는 것이라면 copy는 아예 같은 값을
같는 객체를 새로 생성하는 것이다.

- assign
기본 값으로 값을 복사하는 것이다. week과 유사하나 약간 다른데
week는 객체를 다룬다. 하지만 assign은 스칼라값 (NSInteger와 같은)
을 복사한다는 의미이다.

3.4 쓰레드 세이프 이거나 아니거나

- nonatomic
setter가 쓰레드 세이프하면 쓰레딩 환경에서 아무 걱정 없이 안전에 하게
사용할 수 있겠지만 이 세상에 공짜가 없듯이 여기에서도 비용이 발생한다.
성능이다. 간단한 명령이지만 이것이 반복되어서 사용되다면 문제가 될것이다.
그리고 쓰레드 세이프여야 필요가 꼭 있는 코드를 작성할 일도 많지 않다.
여러가지 이유에서 이 옵션은 많이 사용이 된다.

- atomic
setter가 쓰레드 세이프하도록 설정 (기본 값)

@property (nonatomic, copy) NSString *value;