프로퍼티
- 객체 외부에서 접근할 수 있는 인스턴스의 속성
  - 객체 외부에서 접근할 수 있는 객체 내부의 변수/값
- 객체지향 패러다임의 캡슐화에 관련된 설계
- 인스턴스 변수란?
  - 인스턴스 속성(외부에서 접근할 수 없다.)
  - 인스턴스 변수에 자동적으로 setter와 getter를 생성해주는 것이 property
- 프로퍼티에 접근할때는 method로만 한다. = 객체지향의 목적
- 프로퍼티 생성시 3가지의 종류가 생성이된다
  - 프로퍼티 (@property nsstring *value)
  - 내부변수 (_value)
  - getter (- (nsstring *)value;)
  - setter (- (void)setValue:(nsstring *)newvalue;)

객체 외부에서 내부의 변수에 접근하려면...
- 접근자(getter)와 설정자(setter) 메소드가 필요

프로퍼티는...
- 내부변수 / 접근자(getter) / 설정자(setter)를 모두 자동 생성!

```
//숫자
@property (weak, nonatomic) IBOutlet UILabel *changelabel;
@property NSInteger count;

//성, 이름
@property (weak, nonatomic) IBOutlet UITextField *lastnametext;
@property (weak, nonatomic) IBOutlet UITextField *firstnametext;

//프로퍼티 설정
@property NSString *firstName;
@property NSString *lastName;
@property (readonly) NSString *fullName; //읽기전용 readonly
```

위에 생성된 숫자 프로퍼티를 이용

```
//버튼을 누를경우 1씩 증가
- (IBAction)pushbutton:(id)sender
{
    self.count ++;
    [self.changelabel setText:[NSString stringWithFormat:@"%ld", self.count]];

//    다른방법
//    static NSInteger count = 0;
//    count++;
//    self.changelabel.text = [NSString stringWithFormat:@"%ld", count];
}

//버튼을 누를경우 1씩 감소
- (IBAction)minusbutton:(id)sender
{
    self.count --;
    [self.changelabel setText:[NSString stringWithFormat:@"%ld", self.count]];
}

//버튼을 누를경우 2를 곱함
- (IBAction)doublebutton:(id)sender
{
    self.count = self.count * 2;
    [self.changelabel setText:[NSString stringWithFormat:@"%ld", self.count]];
}

//count라는 프로퍼티를 setter로 사용
- (void)setCount:(NSInteger)count
{
    //_count 내부변수에 언더바를 사용
    _count = count;
    self.changelabel.text = [NSString stringWithFormat:@"%ld", self.count];
}
//------------------------------------------------------------------------------
```

위에 생성된 이름 프로퍼티를 이용

```
//성, 이름을 합친다.
- (IBAction)addname:(id)sender
{
    self.lastName = self.lastnametext.text;
    self.firstName = self.firstnametext.text;

    NSLog(@"%@", self.fullName);
}

//fullname
- (NSString *)fullName
{
    return [NSString stringWithFormat:@"%@%@", _lastName, _firstName];
}
//------------------------------------------------------------------------------

//setter가 getter를 통해 바뀌는것
- (void)setFirstName:(NSString *)firstName
{
    _firstName = firstName;
    self.fullName = [firstName stringByAppendingString:_lastName];
    //stringByAppendingString: 스트링끼리 묶어준다.
    //fullName프로퍼티가 readOnly로 되어있어서 setter로 사용하지못함
}

- (void)setLastName:(NSString *)lastName
{
    _lastName = lastName;
    self.fullName = [NSString stringWithFormat:@"%@%@",_firstName,_lastName];
    [_lastName stringByAppendingString:_firstName];
}
```

Key Value Observing = 키값 관찰자
- 특정 객체의 값이 변화하는 것을 감지하고자 할 때 사용
- 미리 감지하고자 하는 key값을 등록해 두면 KVO 메소드를 통해 변화되는 것을 감지
- 그 감지를 통해 여러가지 동작들을 줄수가 있다.
- 프로퍼티의 접근자 / 설정자를 오버라이드 하지 않아도 값의 변화를 파악할 수 있다.
- 프로퍼티의 접근자 / 설정자를 오버라이드 할 수 없는 상황에도 해당 객체의 특정 값의 변화를 파악할 수 있다.

예) 음악 재생시간이나, 다운로드 등 진행상황을 확인 할수있다.

Key Value Observing 관련 메소드

```
- (void)viewDidLoad {
    [super viewDidLoad];

    //변경하는값을 볼수가 있다(관찰자)
    //observer = 관찰자, keypath = 지켜볼 변수이름, option 4가지, context = nil
    [self addObserver:self
           forKeyPath:@"count"
              options:NSKeyValueObservingOptionNew |
              NSKeyValueObservingOptionInitial |
              NSKeyValueObservingOptionPrior |
              NSKeyValueObservingOptionOld
              context:nil];
}
//------------------------------------------------------------------------------
//값을 지켜보는 메서드
- (void)observeValueForKeyPath:(NSString *)keyPath
                      ofObject:(id)object
                        change:(NSDictionary<NSKeyValueChangeKey,id> *)change
                       context:(void *)context
{
    //kind, new, old, prior
    NSLog(@"keypath : %@\n infodic : %@", keyPath, change);
}

//사용 후 dealloc이 필요하다, 옵저버
- (void)dealloc
{
    [self removeObserver:self forKeyPath:@"count"];
}
//------------------------------------------------------------------------------
```

참조링크
https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/KeyValueObserving/KeyValueObserving.html