#08. Block

전미정

Block

- C언어 문법을 이용하는 블록은 C언어의 함수와 유사한 모양을 지니는 Objective-C의 객체이다. 객체이기 때문에 data structures (NSArray, NAMutableArray, NSDictionary)에 포함될 수 있다.
- 블록은 항상 다른 구조체(메소드 또는 함수)내에 포함되며, 그 안에서 독립적인 역할을 수행한다. 블록은 외부환경(변수, 함수등)과 상호작용을 할 수 있다. 블록 내에서 일어나는 일은 보통 외부로 드러나지 않지만 특별한 방법을 사용하면 외부 변수의 값을 바꿀 수 있다.
- 블럭은 코드를 한층 더 깔끔하게 작성할 수 있도록 해주며, delegate method 역할을 대신 할 수 있다. Delegate의 역할인 ***callback***기능을 할 수 있는데, protocol, delegate method를 따로 작성하지 않고 직접적으로 (범위 내)변수의 접근 및 사용이 가능하다는 장점이 있다. 하지만 delegate처럼 모든 범위(여러 파일)에서 활동하지 못하기 때문에 delegate와 적절히 섞어서 사용하는 것이 좋다.
- 블록은 블록이 포함된 영역 내(메소드)에서 변수의 값을 캡쳐하는 능력이 있으며, 다른 언어의 lambdas, closures(Javascript)와 비슷한 기능을 수행한다.
- 블록은 리턴값과 파라미터를 가질 수 있는데, 이때 파라미터는 각기 다양한 종류의 type을 지닐 수 있다.
- Static/전역변수는 블록이 저장되는 메모리 영역(데이터영역/ 스택영역)과

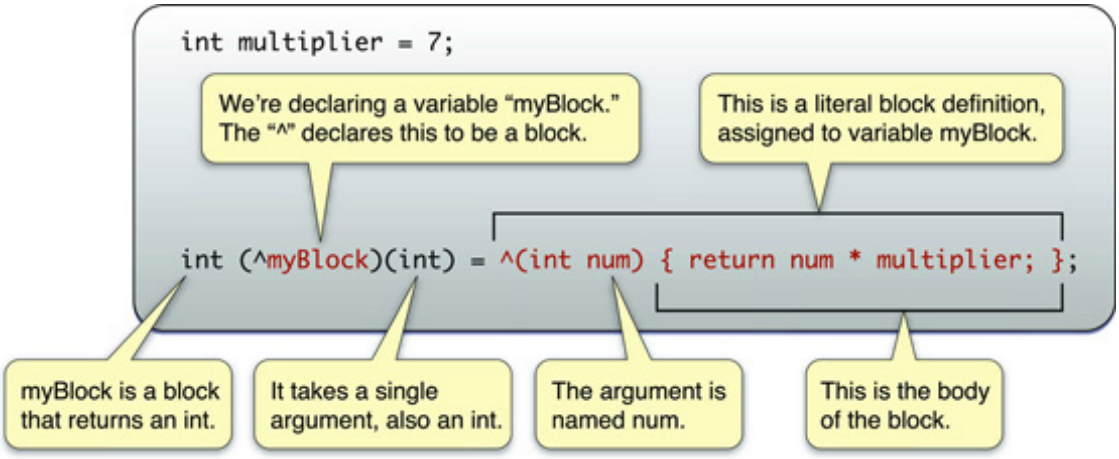
다르며, 프로그램 전체에서 사라지지 않는 값이므로 캡쳐 되지 않고 값을 그대로 가져온다. 값 형태/참조 형태에 상관없이 지역변수의 값만 캡쳐한다.

문법

- 블록의 문법은 Objective-C와 어울리지 않게(!!) 매우 혼란스러운데, 그 기본 문법은 C 언어의 함수와 매우 유사하다. 블록은 ^(caret)을 이용한다.

```
(int)myFunction(int parameterOne, int parameterTwo) {
  // code goes here
  return anInt;
}
```

```
int (^myFunction)(int, int) = ^(int parameterOne, int parameterTwo) {
  // do something here
  return anInt;
}
```



- 블록의 다중 사용을 위해서는 typedef를 사용한다.

```
typedef float (^MyBlockType)(int, float);
```

- 블록은 원래 외부 변수 값 현재 값을 캡처하는 기능을 지니므로, 블록 코드 내에서 한번 캡쳐된 후, 블록이 끝난 뒤 발생한 변수 값의 변화는 반영되지 않는다. 이미 캡쳐된 변수 값의 변화를 반영하기 위해서는 ***__block 변수***를 사용한다.

Completion Handler

실제 iOS개발시 블록이 가장 유용하게 사용되는 부분은 completion handeler이다. 하나의 메소드(함수)가 끝난시점에서 특정 행동을 할 수 있게 해준다(callback). 블록은 delegation과 protocol없이 간단하게 callback 기능을 구현할 수 있게 해준다. 블록은 메소드(함수) 내에서 마지막 파라미터로 들어가야 한다. 가장 많이 사용되는 곳이 modal view controller와 UIView animation 부분이다.

https://goo.gl/nR5igK (apple reference)

https://goo.gl/oq7Es5 (apple)

http://goo.gl/L3IuHh (appcoda)

https://goo.gl/WWxWsn (basic of block, syntax)