OBJECTIVE-C BLOCKS

An Objective-C class defines an object that combines data with related behavior. Sometimes, it makes sense just to represent a single task or unit of behavior, rather than a collection of methods.

Blocks are a language-level feature added to C, Objective-C and C++ which allow you to create distinct segments of code that can be passed around to methods or functions as if they were values. Blocks are Objective-C objects which means they can be added to collections like NSArray or NSDictionary. They also have the ability to capture values from the enclosing scope, making them similar to closures or lambdas in other programming languages

Simple Block declaration syntax

```
returntype (^blockName)(argumentType);
```

Simple block implementation

```
returntype (^blockName)(argumentType)= ^{
};
```

Here is a simple example

```
void (^simpleBlock)(void) = ^{
    NSLog(@"This is a block");
};
```

We can invoke the block using

```
simpleBlock();
```

Blocks Take Arguments and Return Values

Blocks can also take arguments and return values just like methods and functions.

Here is a simple example to implement and invoke a block with arguments and return values.

```
double (^multiplyTwoValues)(double, double) =
    ^(double firstValue, double secondValue) {
     return firstValue * secondValue;
    };
double result = multiplyTwoValues(2,4);
NSLog(@"The result is %f", result);
```

Blocks using type definitions

Here is a simple example using typedef in block. Please note this sample **doesn't work** on the **online compiler** for now. Use **XCode** to run the same.

```
#import <Foundation/Foundation.h>

typedef void (^CompletionBlock)();
@interface SampleClass:NSObject
- (void)performActionWithCompletion:(CompletionBlock)completionBlock;
@end

@implementation SampleClass

- (void)performActionWithCompletion:(CompletionBlock)completionBlock{

    NSLog(@"Action Performed");
    completionBlock();
}

@end

int main()
{
    /* my first program in Objective-C */
    SampleClass *sampleClass = [[SampleClass alloc]init];
    [sampleClass performActionWithCompletion:^{
        NSLog(@"Completion is called to intimate action is performed.");
    }];

    return 0;
}
```

Let us compile and execute it, it will produce the following result:

```
2013-09-10 08:13:57.155 demo[284:303] Action Performed
2013-09-10 08:13:57.157 demo[284:303] Completion is called to intimate action is
performed.
```

Blocks are used more in iOS applications and Mac OS X. So its more important to understand the usage of blocks.