FUNCTION CALL BY REFERENCE IN OBJECTIVE-C

The **call by reference** method of passing arguments to a function copies the address of an argument into the formal parameter. Inside the function, the address is used to access the actual argument used in the call. This means that changes made to the parameter affect the passed argument.

To pass the value by reference, argument pointers are passed to the functions just like any other value. So accordingly, you need to declare the function parameters as pointer types as in the following function **swap**, which exchanges the values of the two integer variables pointed to by its arguments.

```
/* function definition to swap the values */
- (void)swap:(int *)num1 andNum2:(int *)num2
{
   int temp;

   temp = *num1; /* save the value of num1 */
   *num1 = *num2;    /* put num2 into num1 */
   *num2 = temp; /* put temp into num2 */

   return;
}
```

To check more details about Objective-C - Pointers, you can check Objective-C - Pointers chapter.

For now, let us call the function **swap** by passing values by reference as in the following example:

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

@interface SampleClass:NSObject
/* method declaration */
- (void)swap:(int *)num1 andNum2:(int *)num2;
@end

@implementation SampleClass

- (void)swap:(int *)num1 andNum2:(int *)num2
{
   int temp;

   temp = *num1; /* save the value of num1 */
   *num1 = *num2;    /* put num2 into num1 */
   *num2 = temp; /* put temp into num2 */

   return;

}

@end

int main ()
{
   /* local variable definition */
   int a = 100;
   int b = 200;

   SampleClass *sampleClass = [[SampleClass alloc]init];

   NSLog(@"Before swap, value of a : %d\n", a );
   NSLog(@"Before swap, value of b : %d\n", b );

   /* calling a function to swap the values */
   [sampleClass swap:&a andNum2:&b];
```

```
    NSLog(@"After swap, value of a : %d\n", a );
    NSLog(@"After swap, value of b : %d\n", b );

    return 0;
}
```

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

```
2013-09-09 12:27:17.716 demo[6721] Before swap, value of a : 100
2013-09-09 12:27:17.716 demo[6721] Before swap, value of b : 200
2013-09-09 12:27:17.716 demo[6721] After swap, value of a : 200
2013-09-09 12:27:17.716 demo[6721] After swap, value of b : 100
```

Which shows that the change has reflected outside of the function as well unlike call by value where changes does not reflect outside of the function.