Yannick Loriot | Block Cheat Sheet

Definition

A **block** is an **inline anonymous function** which can **capture** the **variables** available **in its context** at the **run-time.** It is always created on the **stack.**

Declaration

```
// Define a block which return nothing and takes as only argument a NSString *
void (^block_name)(NSString *);
```

Typedef

```
// Doing the same thing using a typedef
typedef void (^MyBlockType)(NSString *);
MyBlockType block_name;
```

Creation

```
// Assigning this block to the block_name variable
block_name = ^ void (NSString *parameter) { /* body */ };
```

Call

```
// Call the block_name passing a NSString * as parameter
block_name(@"a string");
```

Passing a block

```
[foo aMethod: ^ BOOL () { return YES; }];

// Infered return type and skipped argument list
[foo aMethod: ^  { return YES; }];
```

__block storage type modifier

__block storage type modifier copy the address/reference of the variable instead of their value.

```
__block int x = 0; // Use the __block keyword to be able to modify it within the block

// Create a block to increment the given variable
void (^increment) () = ^ { x++; };

NSLog(@"%d", x); // "0"
increment();
NSLog(@"%d", x); // "1"
```

Copy / Release

Block_copy: Move a block on the heap.

Block_release: Release a block.

To avoid a **memory leak** you must always use a *Block_release* function with a *Block_copy* function.

Return a block example

```
typedef NSInteger (^PBlock) (NSInteger);

- (PBlock)blockRaisedToPower:(NSInteger)y {
    PBlock block = ^ NSInteger (NSInteger x) {
        return pow(x, y); // y closure
    };
    return [[block copy] autorelease]; // Move to the heap
}

- (void)test {
    PBlock square = [self blockRaisedToPower:2];
    PBlock cube = [self blockRaisedToPower:3];

    NSLog(@"%d", square(3));            // 9
    NSLog(@"%d", cube(3));              // 27
}
```

Callback example

```
// YLAudioPlayer.h
#import <AVFoundation/AVFoundation.h>

typedef void (^YLBlock)(BOOL);
@interface YLAudioPlayer : AVAudioPlayer {
    YLBlock block;
}
@property (nonatomic, copy) YLBlock block;

- (id)initWithURL:(NSURL *)url usingBlock:(YLBlock) block;

@end

// YLAudioPlayer.m
#import "YLAudioPlayer.h"
@implementation YLAudioPlayer
@syntesize block;

- (void)dealloc {
    // Release the block because it has been copied
    [block release], block = nil;
    [super dealloc];
}

- (id)initWithURL:(NSURL *)url usingBlock:(YLBlock) block {
    if ((self = [super initWithContentsOfURL:url error:nil]))  {
        self. block = block;
        self.delegate = self;
    }
    return self;
}

- (void)audioPlayerDidFinishPlaying:(AVAudioPlayer
*)player successfully:(BOOL)flag {
    // Call the block delegate
    block (flag);
}
```

Source: http://yannickloriot.com