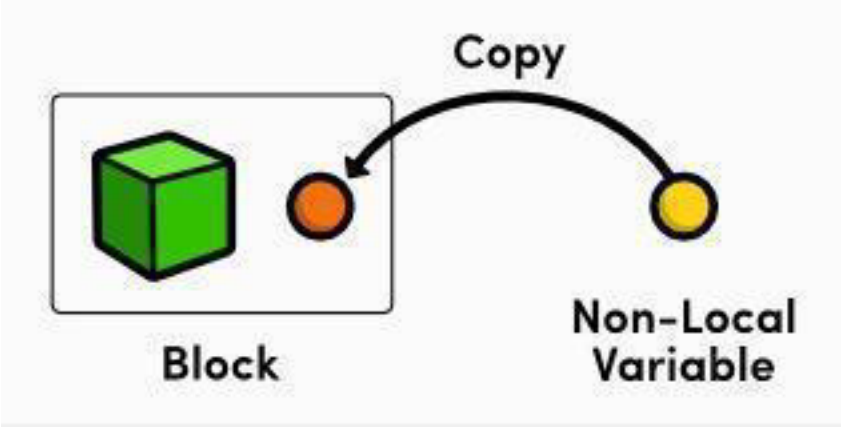
返回目录:全网各大厂 iOS 面试题-题集大全

一个 int 变量被__block 修饰与否的区别？

没有修饰，被 block 捕获，是值拷贝。
使用__block 修饰,会生成一个结构体，复制 int 的引用地址。达到修改数据。

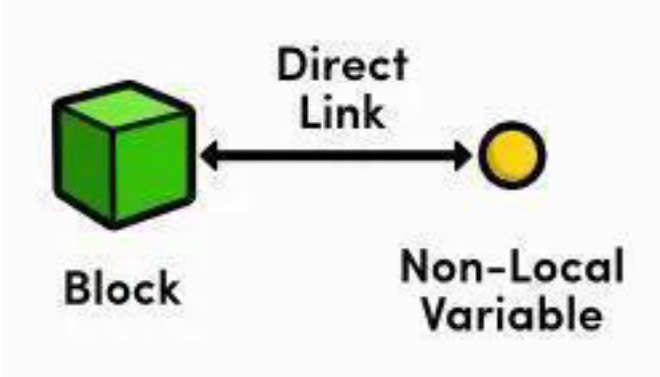
1、block 截获自动变量（局部变量）值

对于 block 外的变量引用，block 默认是将其复制到其数据结构中来实现访问的。也就是说 block 的自动变量截获只针对 block 内部使用的自动变量, 不使用则不截获, 因为截获的自动变量会存储于 block 的结构体内部, 会导致 block 体积变大。特别要注意的是默认情况下 block 只能访问不能修改局部变量的值。



2、 __block 修饰的外部变量

对于用 __block 修饰的外部变量引用，block 是复制其引用地址来实现访问的。block 可以修改__block 修饰的外部变量的值。



3、Block 的存储域及 copy 操作

先来思考一下：Block 是存储在栈上还是堆上呢？ 其实，block 有三种类型：

- 全局块(_NSConcreteGlobalBlock)
- 栈块(_NSConcreteStackBlock)
- 堆块(_NSConcreteMallocBlock)

全局块存在于全局内存中, 相当于单例. 栈块存在于栈内存中, 超出其作用域则马上被销毁 堆块存在于堆内存中, 是一个带引用计数的对象, 需要自行管理其内存 简而言之，存储在栈中的 Block 就是栈块、存储在堆中的就是堆块、既不在栈中也不在堆中的块就是全局块。

遇到一个 Block，我们怎么这个 Block 的存储位置呢？

（1）Block 不访问外界变量（包括栈中和堆中的变量）

Block 既不在栈又不在堆中，在代码段中，ARC 和 MRC 下都是如此。此时为全局块。

（2）Block 访问外界变量

MRC 环境下：访问外界变量的 Block 默认存储栈中。 ARC 环境下：访问外界变量的 Block 默认存储在堆中（实际是放在栈区，然后 ARC 情况下自动又拷贝到堆区），自动释放。

4、防止 Block 循环引用 Block 循环引用的情况： 某个类将 block 作为自己的属性变量，然后该类在 block 的方法体里面又使用了该类本身，如下：

```
self.someBlock = ^(Type var){
    [self dosomething];
};
```

解决办法：

（1）ARC 下：使用 __weak

```
__weak typeof(self) weakSelf = self;
self.someBlock = ^(Type var){
   [weakSelf dosomething];
};
```

（2）MRC 下：使用 __block

```
__block typeof(self) blockSelf = self;
self.someBlock = ^(Type var){
   [blockSelf dosomething];
};
```

值得注意的是，在 ARC 下，使用 __block 也有可能带来的循环引用，如下：

```
// 循环引用 self -> _attributBlock -> tmp -> self
typedef void (^Block)();
```

```
@interface TestObj : NSObject
{
    Block _attributBlock;
}
@end

@implementation TestObj
- (id)init {
    self = [super init];
    __block id tmp = self;
    self.attributBlock = ^{
        NSLog(@"Self = %@",tmp);
        tmp = nil;
   };
}

- (void)execBlock {
    self.attributBlock();
}
@end

// 使用类
id obj = [[TestObj alloc] init];
[obj execBlock]; // 如果不调用此方法，tmp 永远不会置 nil，内存泄露会一直在
```

5、有时候我们经常也会被问到 block 为什么 常使用 copy 关键字？

block 使用 copy 是从 MRC 遗留下来的“传统”,在 MRC 中,方法内部的 block 是在栈区的,使用 copy 可以把它放到堆区.在 ARC 中写不写都行：对于 block 使用 copy 还是 strong 效果是一样的，但写上 copy 也无伤大雅，还能时刻提醒我们：编译器自动对 block 进行了 copy 操作。 如果不写 copy ，该类的调用者有可能会忘记或者根本不知道“编译器会自动对 block 进行了 copy 操作”

返回目录:全网各大厂 iOS 面试题-题集大全
