/\*!

\* @function dispatch\_once

\*

\* @abstract

\* Execute a block once and only once.

\*

\* @param predicate

\* A pointer to a dispatch\_once\_t that is used to test whether the block has

\* completed or not.

\*

\* @param block

\* The block to execute once.

\*

\* @discussion

\* Always call dispatch\_once() before using or testing any variables that are

\* initialized by the block.

\*/

typedef long dispatch\_once\_t;

void

dispatch\_once(dispatch\_once\_t \*predicate,

DISPATCH\_NOESCAPE dispatch\_block\_t block);

**场景：项目中唯一的，且需要全局访问的对象，我们使用单例。**

#import <Foundation/Foundation.h>

@interface User : NSObject

@property (nonatomic, strong) NSString \*job;

+ (User \*)sharedInstance;

@end

#import "User.h"

static dispatch\_once\_t user\_onceToken;

@implementation User

+ (User \*)sharedInstance {

static User \*user = nil;

dispatch\_once(&user\_onceToken, ^{

user = [[User alloc]init];

});

return user;

}

@end

[User sharedInstance].job = @"iOS developer";

NSLog(@"%@====%@",[User sharedInstance],[User sharedInstance].job);//<User: 0x600000018450>====iOS developer

**场景：项目中，在退出时，需要把单例数据清空，该怎么做 ？？**

先简单的把单例对象设置 user = nil; 然而根本没用。  
单例的数据实际上是 static User \*user 这个静态变量保存着。  
方案一：直接把单例的所有属性一一清空数据。  
方案二：单例重新初始化，通过代码控制dispatch\_once的block。(我以前一直误以为是不是dispatch\_once\_t \*predicate，这个指针初始为空，执行一次后非空，这样子控制block是否执行。)

#import "User.h"

static dispatch\_once\_t user\_onceToken;

@implementation User

+ (User \*)sharedInstance {

static User \*user = nil;

NSLog(@"before: %ld--%p",user\_onceToken, &user\_onceToken);

dispatch\_once(&user\_onceToken, ^{

NSLog(@"after: %ld--%p",user\_onceToken, &user\_onceToken);

user = [[User alloc]init];

});

return user;

}

@end

调用单例的代码

[User sharedInstance].job = @"iOS developer";

NSLog(@"%@====%@",[User sharedInstance],[User sharedInstance].job);

输出结果

2018-12-02 07:37:09.284862+0800 dispatch\_once\_t-Demo[51477:2436486] before: 0--0x1001100f8

2018-12-02 07:37:09.285004+0800 dispatch\_once\_t-Demo[51477:2436486] after: 768--0x1001100f8

2018-12-02 07:37:09.285141+0800 dispatch\_once\_t-Demo[51477:2436486] before: -1--0x1001100f8

2018-12-02 07:37:09.285246+0800 dispatch\_once\_t-Demo[51477:2436486] before: -1--0x1001100f8

发现：  
第一次[User sharedInstance]，static dispatch\_once\_t user\_onceToken 初始值为0。  
然后进入了block，static dispatch\_once\_t user\_onceToken 变成了768。  
第二次调用[User sharedInstance]，static dispatch\_once\_t user\_onceToken 初始值为-1，没有进入block。  
第三次调用[User sharedInstance]，static dispatch\_once\_t user\_onceToken 初始值为-1，没有进入block。  
**而且从头到位静态变量指针没有发生变化。**  
**猜想：是不是修改static dispatch\_once\_t user\_onceToken;这个静态变量的值，就可以控制block是否执行。**

#import <Foundation/Foundation.h>

@interface User : NSObject

@property (nonatomic, strong) NSString \*job;

+ (User \*)sharedInstance;

- (void)deleteUser;

@end

static dispatch\_once\_t user\_onceToken;

@implementation User

+ (User \*)sharedInstance {

static User \*user = nil;

NSLog(@"before: %ld--%p",user\_onceToken, &user\_onceToken);

dispatch\_once(&user\_onceToken, ^{

NSLog(@"after: %ld--%p",user\_onceToken, &user\_onceToken);

user = [[User alloc]init];

});

return user;

}

- (void)deleteUser {

user\_onceToken = 0;

}

@end

调用单例

[User sharedInstance].job = @"iOS developer";

NSLog(@"%@====%@",[User sharedInstance],[User sharedInstance].job);

[[User sharedInstance] deleteUser];

NSLog(@"%@====%@",[User sharedInstance],[User sharedInstance].job);

输出结果

2018-12-02 07:50:46.558717+0800 dispatch\_once\_t-Demo[51640:2456936] before: 0--0x10e847100

2018-12-02 07:50:46.558891+0800 dispatch\_once\_t-Demo[51640:2456936] after: 768--0x10e847100

2018-12-02 07:50:46.559008+0800 dispatch\_once\_t-Demo[51640:2456936] before: -1--0x10e847100

2018-12-02 07:50:46.559091+0800 dispatch\_once\_t-Demo[51640:2456936] before: -1--0x10e847100

2018-12-02 07:50:46.559624+0800 dispatch\_once\_t-Demo[51640:2456936] <User: 0x600000012530>====iOS developer

2018-12-02 07:50:46.560110+0800 dispatch\_once\_t-Demo[51640:2456936] before: -1--0x10e847100

2018-12-02 07:50:46.560254+0800 dispatch\_once\_t-Demo[51640:2456936] before: 0--0x10e847100 //改为0

2018-12-02 07:50:46.560501+0800 dispatch\_once\_t-Demo[51640:2456936] after: 768--0x10e847100

2018-12-02 07:50:46.560983+0800 dispatch\_once\_t-Demo[51640:2456936] before: -1--0x10e847100

2018-12-02 07:50:46.561175+0800 dispatch\_once\_t-Demo[51640:2456936] <User: 0x604000007f00>====(null)

大功告成，只需要把dispatch\_once\_t user\_onceToken改成0，就重新初始化User了。

0人点赞

[iOS多线程](file:////nb/40295390)

作者：初灬终  
链接：https://www.jianshu.com/p/4f4b2708bbd6  
来源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。