

iOS多线程技术

iOS有三种多线程编程技术:

- NSThread
- NSOperation
- GCD

它们的抽象程度由低到高,越高的使用起来越简单。

NSThread

显示调用NSThread类

- 类方法

```
[NSThread detachNewThreadSelector:@selector(doSomething:)
toTarget:self withObject:@"hi"];
```

- 实例方法

```
NSThread *thread = [[NSThread alloc] initWithTarget:self
selector:@selector(doSomething:) object:@"hi"];
[thread start];
```

隐式调用

- 开启后台线程

```
[self performSelectorInBackground:@selector(doSomething:)
withObject:@"hi"];
```

- 在主线程中运行

```
[self performSelectorOnMainThread:@selector(doSomething:)
withObject:@"hi" waitUntilDone:YES];
```

- 在指定线程中执行,但该线程必须具备run loop

```
[self performSelector:@selector(doSomething:) onThread:thread
withObject:@"hi" waitUntilDone:YES];
```

常见NSThread的方法

```
+ (NSThread *)currentThread; //获得当前线程
+ (void)sleepForTimeInterval:(NSTimeInterval)ti; //线程休眠
+ (NSThread *)mainThread; //主线程,亦即UI线程了
- (BOOL)isMainThread;
+ (BOOL)isMainThread; //当前线程是否主线程
- (BOOL)isExecuting; //线程是否正在运行
- (BOOL)isFinished; //线程是否已结束
```

NSOperation

- NSInvocationOperation

```
// 创建一个队列
NSOperationQueue queue = [[NSOperationQueue alloc]init];

// 创建子任务, 定义子任务必须是NSOperation的子类
NSInvocationOperation operation = [[NSInvocationOperation alloc]
initWithTarget:self selector:@selector(doSomething:) object:@"hi"];

// 当把任务添加到队列后, 自动开启线程,
[queue addOperation:operation];
```

- NSBlockOperation

```
// 创建一个队列
NSOperationQueue queue = [[NSOperationQueue alloc]init];

// 创建NSBlockOperation对象
NSBlockOperation operation = [NSBlockOperation
blockOperationWithBlock:^(
    [self doSomething];
)];

// 加入队列
[queue addOperation:operation];
```

