线程

创建线程的方法	方法名
第一种创建线程的方法	(void)performSelectorInBackground:withObject:
第二种创建线程的方法	(void)detachNewThreadSelector:toTarget:withObject:
第三种创建一个线程的方法，使用初始化方法	NSThread *thread = [[NSThread alloc] initWithTarget:self selector:@selector(thrid) object:nil];
第四种创建线程的方法，使用 operation 的子类 **`NSInvocationOperation`**	`// 创建一个operation队列` NSOperationQueue *queue = [[NSOperationQueue alloc] init]; `// 实例化一个operation对象` *NSInvocationOperation* operation = [[NSInvocationOperation alloc] initWithTarget:self selector:@selector(four) object:nil]; `// 将operation对象添加进入队列` [queue addOperation:operation];
第五种创建线程的方法， 使用 operation的子类 **`NSBlockOperation`**	`// 实例化一个blockOperation对象` NSBlockOperation *block = [NSBlockOperation blockOperationWithBlock:^{ }]; `// 可以使用addBlockOperation 添加线程` [block addExecutionBlock:^{ }]; `// 需要手动开启` [block start];

GCD

主线程(main_queue)

- `每一个应用程序都只有一个主线程`
- 谁要在主线程上工作呢？在iOS的开发中， `所有的UI操作都必须在主线程上执行`

```
dispatch_queue_t q = dispatch_get_main_queue();
```

全局队列(global_queue)

- 为方便多线程的设计，供所有的app使用
- 全局队列不需要创建，两个队列的执行效果是一样的，全局队列没有名称，调试时无法确认准队列

```
dispatch_queue_t queue =
dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0);
// 第一个参数是优先级,使用默认的优先级 第二个是标记，现在用不到，是未来使用的
```

串行队列(SERIAL)

- 就相当于跑步，一个接一个，保持队行
- 既可以保证效率，又能够容易实现
- 创建异步任务，使用串行队列的异步任务非常非常有用
- dispatch_async 异步分发

最最核心的时把任务放到队列中，同步任务顺序执行，异步任务并发执行

```
// 在c语言中定义类型，绝大多数是_t 或者是ref
dispatch_queue_t queue =
dispatch_queue_create("cn.rocky.demo",DISPATCH_QUEUE_SERIAL);
```

并行队列(CONCURRENT)

- 类似于赛跑
- 特点是没有队形，执行顺序程序员不能控制！也不能控制新建线程的数量
- 并发执行任务，没有先后顺序

```
dispatch_queue_t queue = dispatch_queue_create("fggi",
DISPATCH_QUEUE_CONCURRENT);
```