作者 CoderAO (/users/31b10e251fd8) 2015.07.18 01:28*
写了28901字，被950人关注，获得了658个喜欢
(/users/31b10e251fd8)

使用NSURLSession

写此文时突发灵感作诗一首, 而后置顶, 欢迎品鉴.

> 有的程序员老了,还没听过NSURLSession
> 有的程序员还嫩,没用过NSURLConnection
> 有的程序员很单纯,他只知道AFN.

NSURLConnection在iOS9被宣布弃用,NSURLSession从13年发展到现在,终于迎来了它独步江湖的时代.NSURLSession是苹果在iOS7后为HTTP数据传输提供的一系列接口,比NSURLConnection强大,坑少,好用.今天从使用的角度介绍下.

除了 `NSURLSession` ,文中还会频繁地出现 `NSURLSessionConfiguration` 和 `NSURLSessionTask` 两个类.先认识一下,混个脸熟吧.

使用NSURLSession,拢共分两步:

- 第一步 通过NSURLSession的实例创建task
- 第二部 执行task

既然两步里面都出现了task,就先说说它吧.
NSURLSessionTask可以简单理解为任务:如数据请求任务,下载任务,上传任务and so on.
我们使用的是他的子类们:

- NSURLSessionTask(抽象类)
    - NSURLSessionDataTask

        - NSURLSessionUploadTask

    - NSURLSessionDownloadTask

从这几个子类的名字就可以大概猜出他们的作用了.接下来我们就从不同类型的任务出发,来使用session.

NSURLSessionDataTask

字面上看是和数据相关的任务,但其实dataTask完全可以胜任downloadTask和uploadTask的工作.这可能也是我们使用最多的task种类.

简单GET请求

如果请求的数据比较简单,也不需要对返回的数据做一些复杂的操作.那么我们可以使用带block

```
// 快捷方式获得session对象
NSURLSession *session = [NSURLSession sharedSession];
NSURL *url = [NSURL URLWithString:@"http://www.daka.com/login?username=daka&pwd=123"
// 通过URL初始化task,在block内部可以直接对返回的数据进行处理
NSURLSessionTask *task = [session dataTaskWithURL:url
                                completionHandler:^(NSData *data, NSURLResponse *resp
    NSLog(@"%@", [NSJSONSerialization JSONObjectWithData:data options:kNilOptions er
}];

// 启动任务
[task resume];
```

Tips:

- 所有类型的task都要调用resume方法才会开始进行请求.

简单POST请求

POST和GET的区别就在于request,所以使用session的POST请求和GET过程是一样的,区别就在于对request的处理.

```
NSURL *url = [NSURL URLWithString:@"http://www.daka.com/login"];
NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url];
request.HTTPMethod = @"POST";
request.HTTPBody = [@"username=daka&pwd=123" dataUsingEncoding:NSUTF8StringEncoding]

NSURLSession *session = [NSURLSession sharedSession];
// 由于要先对request先行处理,我们通过request初始化task
NSURLSessionTask *task = [session dataTaskWithRequest:request
                                    completionHandler:^(NSData *data, NSURLResponse *
[task resume];
```

NSURLSessionDataDelegate代理方法

NSURLSession提供了block方式处理返回数据的简便方式,但如果想要在接收数据过程中做进一步的处理,仍然可以调用相关的协议方法.NSURLSession的代理方法和NSURLConnection有些类似,都是分为接收响应、接收数据、请求完成几个阶段.

```
// 使用代理方法需要设置代理,但是session的delegate属性是只读的,要想设置代理只能通过这种方式创建ses
NSURLSession *session = [NSURLSession sessionWithConfiguration:[NSURLSessionConfigura
                                                      delegate:self
                                                 delegateQueue:[[NSOperationQueue al

// 创建任务(因为要使用代理方法,就不需要block方式的初始化了)
NSURLSessionDataTask *task = [session dataTaskWithRequest:[NSURLRequest requestWithU

// 启动任务
[task resume];

//对应的代理方法如下:

// 1.接收到服务器的响应
- (void)URLSession:(NSURLSession *)session dataTask:(NSURLSessionDataTask *)dataTask
    // 允许处理服务器的响应，才会继续接收服务器返回的数据
    completionHandler(NSURLSessionResponseAllow);
}

// 2.接收到服务器的数据（可能调用多次）
- (void)URLSession:(NSURLSession *)session dataTask:(NSURLSessionDataTask *)dataTask
    // 处理每次接收的数据
}

// 3.请求成功或者失败（如果失败，error有值）
- (void)URLSession:(NSURLSession *)session task:(NSURLSessionTask *)task didCompleteW
    // 请求完成,成功或者失败的处理
}
```

Tips:

关键点在代码注释里面都有提及,重要的地方再强调一下:

- 如果要使用代理方法,需要设置代理,但从NSURLSession的头文件发现session的delegate属性是只读的.因此设置代理要通过session的初始化方法赋值: `sessionWithConfiguration:delegate:delegateQueue:` 其中:
    - configuration参数(文章开始提到的)需要传递一个配置,我们暂且使用默认的配置 `[NSURLSessionConfiguration defaultSessionConfiguration]` 就好(后面会说下这个配置是干嘛用的);
    - delegateQueue参数表示协议方法将会在哪个队列(NSOperationQueue)里面执行.

- NSURLSession 在接收到响应的时候要先对响应做允许处理: `completionHandler(NSURLSessionResponseAllow);`,才会继续接收服务器返回的数据,进入后面的代理方法.值得一提的是,如果在接收响应的时候需要对返回的参数进行处理(如获取响应头信息等),那么这些处理应该放在前面允许操作的前面.

NSURLSessionDownloadTask

文件下载可以使用NSURLSessionDownloadTask这个子类.

简单下载

NSURLSessionDownloadTask同样提供了通过NSURL和NSURLRequest两种方式来初始化并通过block进行回调的方法.下面以NSURL初始化为例:

```
SURLSession *session = [NSURLSession sharedSession];
NSURL *url = [NSURL URLWithString:@"http://www.daka.com/resources/image/icon.png"] ;
NSURLSessionDownloadTask *task = [session downloadTaskWithURL:url completionHandler:
    // location是沙盒中tmp文件夹下的一个临时url,文件下载后会存到这个位置,由于tmp中的文件随时可能
    NSString *path = [[NSSearchPathForDirectoriesInDomains(NSCachesDirectory, NSUserl
    // 剪切文件
    [[NSFileManager defaultManager] moveItemAtURL:location toURL:[NSURL fileURLWithP
}];
    // 启动任务
    [task resume];
```

Tips:

- 需要注意的就是需要将下载到tmp文件夹的文件转移到需要的目录.原因在代码中已经贴出.
- `response.suggestedFilename` 是从相应中取出文件在服务器上存储路径的最后部分,如数据在服务器的url为 `http://www.daka.com/resources/image/icon.png`,那么其suggestedFilename就是icon.png.

NSURLSessionDownloadDelegate代理方法

同样的,downloadTask也提供了配套的代理方法

```
// 每次写入调用(会调用多次)
- (void)URLSession:(NSURLSession *)session downloadTask:(NSURLSessionDownloadTask *)
// 可在这里通过已写入的长度和总长度算出下载进度
CGFloat progress = 1.0 * totalBytesWritten / totalBytesExpectedToWrite; NSLog(@"%f",
}

// 下载完成调用
- (void)URLSession:(NSURLSession *)session
      downloadTask:(NSURLSessionDownloadTask *)downloadTask
      didFinishDownloadingToURL:(NSURL *)location {
    // location还是一个临时路径,需要自己挪到需要的路径(caches下面)
    NSString *filePath = [[NSSearchPathForDirectoriesInDomains(NSCachesDirectory, NSU
    [[NSFileManager defaultManager] moveItemAtURL:location toURL:[NSURL fileURLWithPa
}

// 任务完成调用
- (void)URLSession:(NSURLSession *)session task:(NSURLSessionTask *)task didCompleteW

}
```

NSURLSessionUploadTask

在NSURLSession中,文件上传方式主要有以下两种:

```
NSURLSessionUploadTask *task =
[[NSURLSession sharedSession] uploadTaskWithRequest:request
                                           fromFile:fileName
                                  completionHandler:^(NSData *data, NSURLResponse *re
}];
```

和

```
 [self.session uploadTaskWithRequest:request
                            fromData:body
                   completionHandler:^(NSData *data, NSURLResponse *response, NSErro
 NSLog(@"-------%@", [NSJSONSerialization JSONObjectWithData:data options:kNilOption
 }];
```

处于安全性考虑,通常我们会使用POST方式进行文件上传,所以较多使用第二种方式.

但是,NSURLSession并没有为我们提供比NSURLConnection更方便的文件上传方式.方法中 `body` 处的参数需要填写request的请求体(http协议规定格式的大长串).因为你有90%的可能性用了AFNetworking,即使是自己写的应该也是copy,所以代码就不贴了我们只说方法呵呵哒.

断点下载

NSURLSessionDownloadTask提供了与断点下载相关的几个方法:

```
// 使用这种方式取消下载可以得到将来用来恢复的数据,保存起来
[self.task cancelByProducingResumeData:^(NSData *resumeData) {
    self.resumeData = resumeData;
}];

// 由于下载失败导致的下载中断会进入此协议方法,也可以得到用来恢复的数据
- (void)URLSession:(NSURLSession *)session task:(NSURLSessionTask *)task didCompleteW
{
    // 保存恢复数据
    self.resumeData = error.userInfo[NSURLSessionDownloadTaskResumeData];
}

// 恢复下载时接过保存的恢复数据
self.task = [self.session downloadTaskWithResumeData:self.resumeData];
// 启动任务
[self.task resume];
```

以目前我对NSURLSession的理解这种断点下载只支持应用内断点,如果程序在下载过程中途关闭,则不能恢复下载.(暂时对NSURLSession理解还不全面,不敢妄下断论,如有不妥简友们可以沟通下)

其他

此外,task们自身有都拥有下面几个方法

```
- (void)suspend;
- (void)resume;
- (void)cancel;
```

suspend可以让当前的任务暂停
resume方法不仅可以启动任务,还可以唤醒suspend状态的任务
cancel方法可以取消当前的任务,你也可以向处于suspend状态的任务发送cancel消息,任

务如果被取消便不能再恢复到之前的状态.

NSURLSessionConfiguration

简单地说,就是session的配置信息.如:

```
NSURLSessionConfiguration *config = [NSURLSessionConfiguration defaultSessionConfigu
// 超时时间
config.timeoutIntervalForRequest = 10;
// 是否允许使用蜂窝网络(后台传输不适用)
config.allowsCellularAccess = YES;
// 还有很多可以设置的属性
```

有没有发现我们使用的Configuration都是默认配置:`[NSURLSessionConfiguration defaultSessionConfiguration]`,其实它的配置有三种类型:

```
+ (NSURLSessionConfiguration *)defaultSessionConfiguration;
+ (NSURLSessionConfiguration *)ephemeralSessionConfiguration;
+ (NSURLSessionConfiguration *)backgroundSessionConfigurationWithIdentifier:(NSStrin
```

表示了NSURLSession几种不同的工作模式.
默认的配置会将缓存存储在磁盘上,第二种瞬时会话模式不会创建持久性存储的缓存,第三种后台会话模式允许程序在后台进行上传下载工作.

除了支持任务的暂停和断点续传,我觉得NSURLSession之于NSURLConnection的最伟大的进步就是支持后台上传下载任务,这又是一个可以深入讨论的话题.但在这方面我还没有进行深入的研究,待后续了解之后另行开贴.

PS:AFNetWorking从2.0版本就有了基于NSURLSession的系列封装,感兴趣的童鞋自行前往了解.

¥ 打赏支持

♡ 喜欢 | 70

分享到微博 分享到微信
更多分享

13条评论 （ 按时间正序 · 按时间倒序 · 按喜欢排序 ） 添加新评论 (/sign_in)

小凡凡520 (/users/af956396d977)
2楼 · 2016.05.01 11:56 (/p/fafc67475c73/comments/2221894#comment-2221894)
(/users/af956396d977)
good mark

♡ 喜欢(0) 回复

iOS_愛OS (/users/6b0da6cb659c)
3楼 · 2016.06.08 11:34 (/p/fafc67475c73/comments/2685819#comment-2685819)
(/users/6b0da6cb659c)
漂亮

♡ 喜欢(0) 回复

NiuBaBa (/users/946bdd048d3f)
4楼 · 2016.06.14 20:36 (/p/fafc67475c73/comments/2756426#comment-2756426)
(/users/946bdd048d3f)
mark 请问代码的排版是怎么弄的?

♡ 喜欢(1) 回复

annj (/users/4b6513334335)： @NiuBaBa (/users/946bdd048d3f) 简书设置里选择markDown编辑器，<pre>中间放代码</pre>
2016.07.15 10:38 (/p/fafc67475c73/comments/3126447#comment-3126447) 回复

NiuBaBa (/users/946bdd048d3f)： @annj (/users/4b6513334335) 谢谢了
2016.07.16 09:28 (/p/fafc67475c73/comments/3138839#comment-3138839) 回复

添加新回复

扣肉快快跑 (/users/836199c40552)
5楼 · 2016.06.24 13:05 (/p/fafc67475c73/comments/2871315#comment-2871315)
(/users/836199c40552)
省略的上传还要查看别资料了

♡ 喜欢(0) 回复

Oye0815 (/users/2e0cde7b84aa)
6楼 · 2016.07.12 09:48 (/p/fafc67475c73/comments/3087037#comment-3087037)
(/users/2e0cde7b84aa)
内容很完善了，学习了。但是代码有明显的错误，缺少了指针符合“*”，如下面的代码(NSURLSession)session ----> (NSURLSession *)session

```
// 1.接收到服务器的响应
- (void)URLSession:(NSURLSession )session dataTask:(NSURLSessionDataTask )dataTask didReceiveResponse:(NSURLResponse *)response completionHandler:(void (^)(NSURLSessionResponseDisposition))completionHandler {
// 允许处理服务器的响应，才会继续接收服务器返回的数据
completionHandler(NSURLSessionResponseAllow);
}

// 2.接收到服务器的数据（可能调用多次）
- (void)URLSession:(NSURLSession )session dataTask:(NSURLSessionDataTask )dataTask didReceiveData:(NSData *)data {
// 处理每次接收的数据
}

// 3.请求成功或者失败（如果失败，error有值）
- (void)URLSession:(NSURLSession )session task:(NSURLSessionTask )task didCompleteWithError:(NSError *)error {
// 请求完成,成功或者失败的处理
}
```

♡ 喜欢(0) 回复

> CoderAO (/users/31b10e251fd8)： @Oye0815 (/users/2e0cde7b84aa) 谢谢提醒,已经改过来了,之前都是在其他编辑器编辑然后复制过来,不知道复制过来的时候为什么有很多*就没掉了...这些代码都是Xcode里面敲的然后复制过去,理论上不会出现缺少*的情况.
> 2016.08.03 14:44 (/p/fafc67475c73/comments/3429876#comment-3429876) 回复
>
> 添加新回复

全民同學 (/users/a2d5d5edcee5)
7楼 · 2016.07.14 14:04 (/p/fafc67475c73/comments/3115071#comment-3115071)
(/users/a2d5d5edcee5)
"拢共"这个成用的好啊，生动形象

♡ 喜欢(0) 回复

cyhai (/users/24c16721bbdb)
8楼 · 2016.07.28 11:19 (/p/fafc67475c73/comments/3325015#comment-3325015)
(/users/24c16721bbdb)
写的不错，就是感觉有些粗心

♡ 喜欢(0) 回复

smallLabel (/users/68f1f79dd096)
9楼 · 2016.08.17 17:27 (/p/fafc67475c73/comments/3685843#comment-3685843)
(/users/68f1f79dd096)
看完了，写的挺好的，马了

♡ 喜欢(0) 回复

Xiaoye__220 (/users/b8e0c9b678f1)
10楼 · 2016.08.18 09:50 (/p/fafc67475c73/comments/3696212#comment-3696212)
(/users/b8e0c9b678f1)
好诗好诗

♡ 喜欢(0) 回复

我的大名叫小爱 (/users/e6868bf4a355)
11楼 · 2016.08.18 10:21 (/p/fafc67475c73/comments/3706187#comment-3706187)
(/users/e6868bf4a355)
你知道无线互联???

♡ 喜欢(0) 回复

登录后发表评论 (/sign_in)

被以下专题收入，发现更多相似内容：

iOS 开发 (/collection/2ffaa203eb6a)
7942篇文章 (/collection/2ffaa203eb6a) · 6830人关注
(/collection/2ffaa203eb6a)

+ 添加关注 (/sign_in)

iOS学习 (/collection/1332c736fe39)

学习从点滴开始！(PS: 拒绝部分投稿的文章仅仅是由于专题内已收录相关知识点的文章, 并非是投稿的文章技术含量不够好, 望谅解.)

4395篇文章 (/collection/1332c736fe39) · 5263人关注

添加关注 (/sign_in)

寒哥管理的技术专题 (/collection/5be41e88940c)

心情不好的时候问自己： 我为何这么屌 心情好的时候问自己： 为什么比我屌的这么多

2476篇文章 (/collection/5be41e88940c) · 3662人关注

添加关注 (/sign_in)