**Http2的应用**

**HTTP2.0新特性**

二进制分帧

首部压缩

多路复用

服务器推送

请求优先级

**HTTP/1.1存在许多问题 :**

1,单路连接 请求低效2,HTTP只允许由客户端主动发起请求,意味着缺少预加载的功能3,HTTP头冗余,每次发送HTTP请求都有一堆重复的header value

**HTTP/2 有什么改进 :**

1,一个TCP连接能发起多个HTTP请求2,实现完全多路复用, 意味着一个新的请求不用等到上一个请求得到响应之后再发出3,有请求优先级, 所以能把重要的资源优先提供给拥有更高权限的客户端.4,使用二进制分帧, 使得数据的处理和解析速度更快.5,使用 HPACK头压缩技术6,使用一个静态表和一个动态表7,静态表包含最常用的HTTP信息头, 并且不可修改

8,包含在静态表中的信息头可以动态添加到动态表中, 动态表中的信息头通过指针引用静态表中的信息9,HTTP/2自动支持HTTPS, 意味着HTTP/2是安全的

10,支持服务器自推送

**总之:**多路复用, 请求优先级优化,支持服务器推送技术,压缩HTTP头

**HTTP/2多路复用是如何解决队首阻塞的问题的**

在HTTP/1.1的时候曾试过采用HTTP pipelining (HTTP 管道/流水线 技术)能实现同一TCP连接中不用等待旧请求的响应就可以发送新请求. 但是HTTP pipelining有个致命的缺点 : HTTP响应仍然是按照请求的顺序依次收;

HTTP/2 多路复用+请求优先级, 发送的时候还是依次发送请求, 但是与此同时我们同时得到了回复, 同时, 更高优先级的请求我们得到以及发送给客户端的速度更快;

**HTTP/2 Multiplexing,服务器自推送**

在HTTP/1.1时代, 我们没有服务器自推送, 只能发一个请求, 得到响应之后再发第二个请求... 延迟可想而知

HTTP/1.1 没有服务器自推送,而在HTTP/2时代, 我们有了服务器自推送, 服务器会把相关联的数据全部push给我们 ,性能马上强了一大截, 我们再也不用像以前那样苦苦等待了. 并且这功能已经内嵌在NSURLSession中, 我们不需要写任何一行代码来支持其实现.

**iOS中适配HTTP/2**

NSURLSession已经自动支持HTTP/2, 客户端不需要额外写任何代码. 只需要一台支持HTTP/2通信的服务器即可. 要是没有怎么办, 也没关系,服务器支持HTTP/2   
NSURLSession自动使用HTTP/2进行通信 ,服务器不支持HTTP/2 ,NSURLSession自动使用HTTP/1.1或其他更优的协议

我们一句代码, 判断都不用写,.

**NSURLSession replace NSURLConnection,NSURLSession的新功能**

cookie 共享

增加sessionCookieGroup以实现app与其扩展(例如通知中心等等)之间cookie的共享

以前我们使用NSInputStream/NSOutputStream来进行一些非HTTP的连接, 例如利用TCP连接一台远程的服务器等等, 现在我们有了NSURLSessionStreamTask让我们更简单地实现以上功能.

**NSURLSessionStreamTask的特性** :

更轻松地使用TCP进行通信

替代NSInputStream/NSOutputStream, 提供更优的API

异步读写API

内置强大的支持功能, 能自动通过HTTP代理, 连接一个远程服务器.轻松转换成NSStream

**NSURLSessionTaskMetrics**

对发送请求/DNS查询/TLS握手/请求响应等各种环节时间上的统计. 更易于我们检测, 分析我们的请求缓慢到底是发生在哪个环节, 并对此进行优化提升我们APP的性能.

NSURLSessionTaskDelegate代理中新增一个方法- (void)URLSession:(NSURLSession \*)session task:(NSURLSessionTask \*)task didFinishCollectingMetrics:(NSURLSessionTaskMetrics \*)metrics, 当收集完成的时候就会调用该方法.,我们只需要实现代理的这个方法就能在这里做统计, 输出等等操作.

**iOS引入使用LZMA压缩算法**

之所以选择LZMA算法进行压缩处理，原因有以下几点：开源 ,iOS & Android平台均支持，可夸平台使用,使用广泛稳定，7zip即采用该算法及衍生算法,压缩效率较高,多线程支持

**利用系统默认支持的LZMA压缩算法**

Apple提供了一套通用的无损压缩算法，其中就支持LZMA、LZMA2压缩