参考：<http://blog.csdn.net/huawei_mcu/article/details/14056025>

浅析轮询（polling）和推送（long-polling）

实时 Web 应用的窘境

Web 应用的信息交互过程通常是客户端通过浏览器发出一个请求，服务器端接收和审核完请求后进行处理并返回结果给客户端，然后客户端浏览器将信息呈现出来，这种机制对于信息变化不是特别频繁的应用尚能相安无事，但是对于那些实时要求比较高的应用来说，比如说在线游戏、在线证券、设备监控、新闻在线播报、RSS 订阅推送等等，当客户端浏览器准备呈现这些信息的时候，这些信息在服务器端可能已经过时了。所以保持客户端和服务器端的信息同步是实时 Web 应用的关键要素，对 Web 开发人员来说也是一个难题。在 WebSocket 规范出来之前，开发人员想实现这些实时的 Web 应用，不得不采用一些折衷的方案，其中最常用的就是轮询 (Polling) 和 Comet 技术，而 Comet 技术实际上是轮询技术的改进，又可细分为两种实现方式，一种是长轮询机制，一种称为流技术。下面我们简单介绍一下这几种技术：

1. 轮询：

这是最早的一种实现实时 Web 应用的方案。客户端以一定的时间间隔向服务端发出请求，以频繁请求的方式来保持客户端和服务器端的同步。这种同步方案的最大问题是，当客户端以固定频率向服务器发起请求的时候，服务器端的数据可能并没有更新，这样会带来很多无谓的网络传输，所以这是一种非常低效的实时方案。

轮询(Polling)是指不管服务器端有没有更新，客户端(通常是指浏览器)都定时的发送请求进行查询，轮询的结果可能是服务器端有新的更新过来，也可能什么也没有，只是返回个空的信息。不管结果如何，客户端处理完后到下一个定时时间点将继续下一轮的轮询。

1. 推送或叫长连接(Long-Polling)的服务其客户端是不做轮询的，客户端在发起一次请求后立即挂起，一直到服务器端有更新的时候，服务器才会主动推送信息到客户端。 在服务器端有更新并推送信息过来之前这个周期内，客户端不会有新的多余的请求发生，服务器端对此客户端也啥都不用干，只保留最基本的连接信息，一旦服务器有更新将推送给客户端，客户端将相应的做出处理，处理完后再重新发起下一轮请求。分为长轮询和流2种：

长轮询：

长轮询是对定时轮询的改进和提高，目地是为了降低无效的网络传输。当服务器端没有数据更新的时候，连接会保持一段时间周期直到数据或状态改变或者时间过期，通过这种机制来减少无效的客户端和服务器间的交互。当然，如果服务端的数据变更非常频繁的话，这种机制和定时轮询比较起来没有本质上的性能的提高。

流：

流技术方案通常就是在客户端的页面使用一个隐藏的窗口向服务端发出一个长连接的请求。服务器端接到这个请求后作出回应并不断更新连接状态以保证客户端和服务器端的连接不过期。通过这种机制可以将服务器端的信息源源不断地推向客户端。这种机制在用户体验上有一点问题，需要针对不同的浏览器设计不同的方案来改进用户体验，同时这种机制在并发比较大的情况下，对服务器端的资源是一个极大的考验。

举个例子说明下就很清楚了：

1. 轮询模式，假设是客户端每2秒轮询一次，那么客户端每2秒就会发送一次请求，相应的服务器端每2秒就要响应这个客户端的一次请求。而实际上服务器端可能1秒钟后就有更新，也可能1分钟后才有更新。对于1秒钟就有更新的，客户端至少会有1秒钟的延时；而1分钟后才有更新的，只有最后一次查询有意义，这一分钟内的轮询其实都是没有必要的，服务器端和客户端均有资源的浪费。
2. 推送模式，客户端发送一次请求后马上挂起等待服务器端响应，可能1秒，也可能10秒钟，也可能1分钟。如果服务器端是1秒就有更新，那么到1秒钟时客户端马上就收到更新了，如果是1分钟才有更新，那么整个一分钟客户端也只请求一次，服务器也只会相应一次，这个跟轮询的区别是不是已经很清楚了。

综合这几种方案，您会发现这些目前我们所使用的所谓的实时技术并不是真正的实时技术，它们只是在用 Ajax 方式来模拟实时的效果，在每次客户端和服务器端交互的时候都是一次 HTTP 的请求和应答的过程，而每一次的 HTTP 请求和应答都带有完整的 HTTP 头信息，这就增加了每次传输的数据量，而且这些方案中客户端和服务器端的编程实现都比较复杂，在实际的应用中，为了模拟比较真实的实时效果，开发人员往往需要构造两个 HTTP 连接来模拟客户端和服务器之间的双向通讯，一个连接用来处理客户端到服务器端的数据传输，一个连接用来处理服务器端到客户端的数据传输，这不可避免地增加了编程实现的复杂度，也增加了服务器端的负载，制约了应用系统的扩展性。