WebSocket

概述

WebSocket是一种在单个TCP连接上进行全双工通信的协议。WebSocket使得客户端和服务器之间的数据交换变得更加简单，允许服务端主动向客户端推送数据。在WebSocket API中，浏览器和服务器只需要完成一次握手，两者之间就直接可以创建持久性的连接，并进行双向数据传输。

很多网站为了实现推送技术，所用的技术都是**轮询**。轮询是在特定的的时间间隔（如每1秒），由浏览器对服务器发出HTTP请求，然后由服务器返回最新的数据给客户端的浏览器。这种传统的模式带来很明显的缺点，即浏览器需要不断的向服务器发出请求，然而HTTP请求可能包含较长的头部，其中真正有效的数据可能只是很小的一部分，显然这样会浪费很多的带宽等资源。

WebSocket是基于Http协议的，或者说借用了Http协议来完成一部分握手，在握手阶段与Http是相同的。(HTTP connection upgrade request)

- 建立在 TCP 协议之上，服务器端的实现比较容易。
- 与 HTTP 协议有着良好的兼容性。默认端口也是80和443，并且握手阶段采用 HTTP 协议，因此握手时不容易屏蔽，能通过各种 HTTP 代理服务器。
- 数据格式比较轻量，性能开销小，通信高效。
- 可以发送文本，也可以发送二进制数据。
- 没有同源限制，客户端可以与任意服务器通信。
- 协议标识符是ws（如果加密，则为wss），服务器网址就是 URL。

对比

- **基于TCP协议**：HTTP协议目的是规定客户端和服务端数据传输的格式和数据交互行为，并不负责数据传输的细节。底层是基于TCP实现的。
- **无状态连接**：HTTP协议本身不对请求和响应之间的通信状态进行保存。
- **多次请求**：由于管道机制可实现一次TCP连接同时多个HTTP请求。客户端请求服务器时先响应HTML，再请求加载CSS,JS，图片等资源。
- **持久连接**：当TCP连接建立后，只要任意一端没有明确提出断开连接，则保持TCP连接状态。减少TCP重复建立和断开的开销及服务器端的负载。不过这个持久连接只会持续一分钟左右，并且通常情况下是浏览器控制的。不像Ws可以持续很久很久。
- **使用Cookie和Session机制管理状态**：为了实现保存通信状态，引入了Cookie和Session技术。比如用户登录网站跳转到其他页面能够保存用户状态，而不需要重新登录。
- **全明文传输**: 报文数据不加密（这一方面从某种意义上也方便了开发人员调试bug），敏感信息易泄露。可通过在代码上使用SSL/TLS协议改造系统，加密请求响应报文，需调试可将报文解密后保存到日志中进行问题排查。 内容编码: 由于某些报文的内容过大，因此在传输时，为了减少传输的时间，会采取一些压缩的措施。

Socket 是操作系统提供的对于传输层（TCP / UDP）抽象的接口，是一个编程概念，而 Websocket 与 HTTP 一样是一个成文的互联网协议。