### Websocket协议简介

Websocket是一个持久化的协议，相对于HTTP这种非持久的协议来说；

**HTTP协议**：

HTTP的生命周期通过 Request 来界定，也就是一个 Request 一个 Response ，那么在 HTTP1.0 中，这次HTTP请求就结束了。

在HTTP1.1中进行了改进，使得有一个keep-alive，也就是说，在一个HTTP连接中，可以发送多个Request，接收多个Response。但是请记住 Request = Response ， 在HTTP中永远是这样，也就是说一个request只能有一个response。而且这个response也是被动的，不能主动发起。

**Websocket协议：**

WebSocket是一种双向通信协议。在建立连接后，WebSocket服务器端和客户端都能主动向对方发送或接收数据，就像Socket一样；  
WebSocket需要像TCP一样，先建立连接，连接成功后才能相互通信。

WebSocket只需要经过一次HTTP请求，就可以做到源源不断的信息传送了（在程序设计中，这种设计叫做回调，即：你有信息了再来通知我，而不是我傻乎乎的每次跑来问你）

### 添加Websocket插件

在jmeter中测试websocket协议接口，需要先安装websocket插件和依赖包；

需添加的插件为：

JMeterWebSocketSampler-1.0.2-SNAPSHOT.jar

插件依赖的jar包为：

1、jetty-http-9.1.2.v20140210.jar

2、jetty-io-9.1.2.v20140210.jar

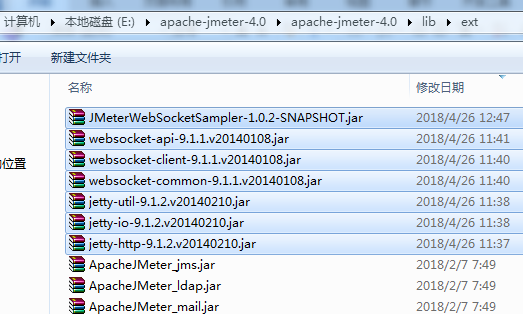
3、jetty-util-9.1.2.v20140210.jar

4、websocket-api-9.1.1.v20140108.jar

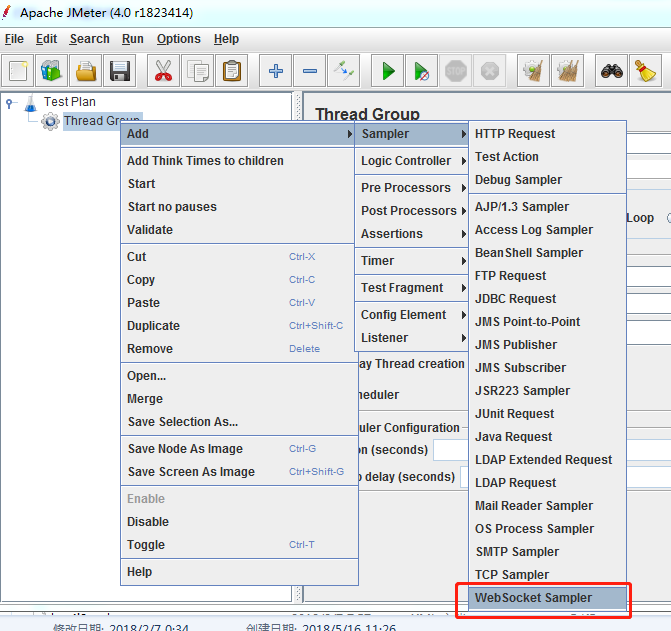
5、websocket-client-9.1.1.v20140108.jar

6、websocket-common-9.1.1.v20140108.jar

准备好这些jar包后，将它们拷贝到Jmeter目录的\lib\ext\下。

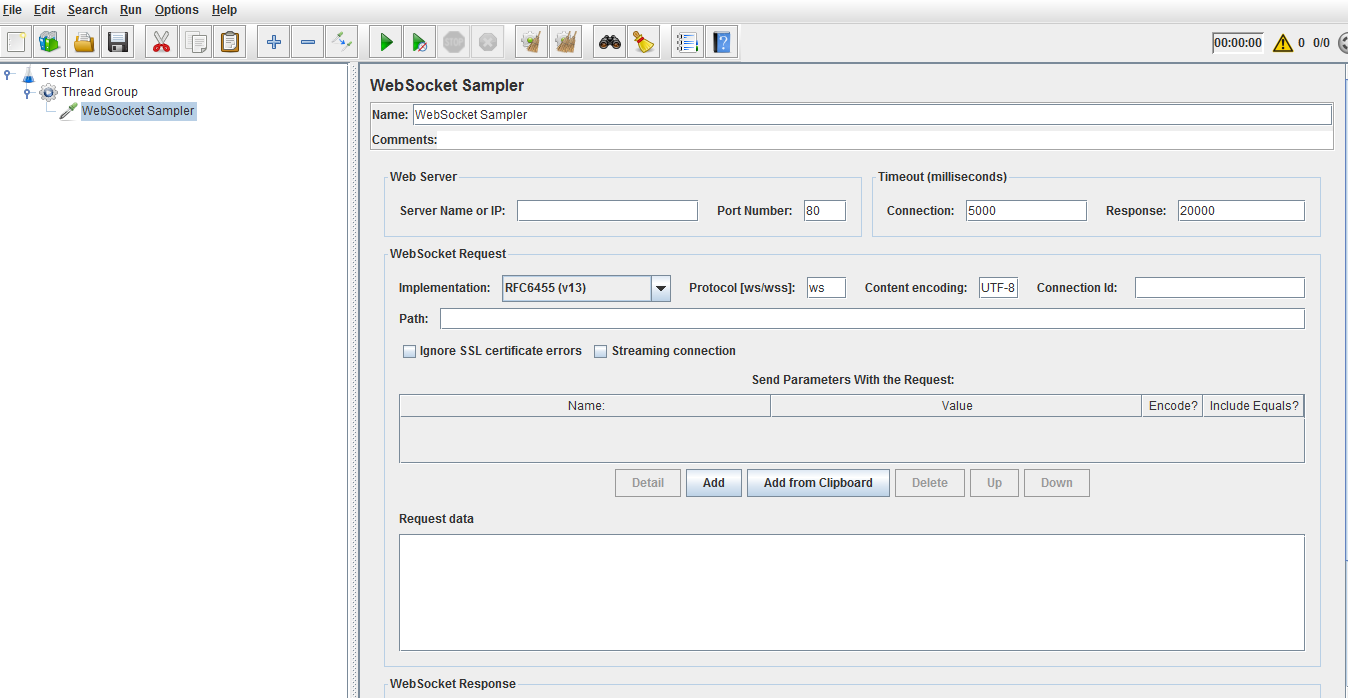


打开jmeter，在sampler下，可以查看到websocket sampler



### 三、Websocket sampler说明

在jmeter中，创建websocket sampler，界面如下；



针对界面中各选项说明如下：

1. WebServer  
     
   （1）Server Name or IP：WebSocket发送的目标服务器的地址或者名称  
     
   （2）Port Number：WebSocker服务器监听的端口号。（一般是HTTP 80端口，可以通过fiddler、WireShark数据包得到）  
     
   2、Timeout:   
     
   （1）Connection – 发送一个连接请求后，Jmeter等待连接完成的最长时间，单位是毫秒。  
   （2）Response - 对响应消息的最大等待时间。  
     
   3、WebSocket Request  
     
   （1）Implementation – 只支持RFC6455(v13) ，WebSocket协议标准的最新版。  
     
   （2）Protocol – 有ws与wss之分， ws前缀是WebSocket连接的辨别标识，wss前缀是WebSocket安全连接的辨别标识。根据自己的实际情况填写  
   （3）Streaming Connection – 选择这个TCP session要不要保持，如果勾上标识连接会一直存在，如果没有勾上，那么得到第一次响应后该链接就会被关闭。  
   （4）Request data：填入将要发送的请求，要跟开发沟通好，这个是什么格式的消息。  
     
   4.WebSocket Response  
     
   （1）Response Pattern – 采样器将等待含有该标识的消息并继续通信（或者直到timeout，该连接关闭）  
   （2）Close Connection Pattern – 如果服务器返回的消息含有这样的字符，就结束会话。  
   （3）Message Backlog – 定义服务器返回消息保留的最大长度。

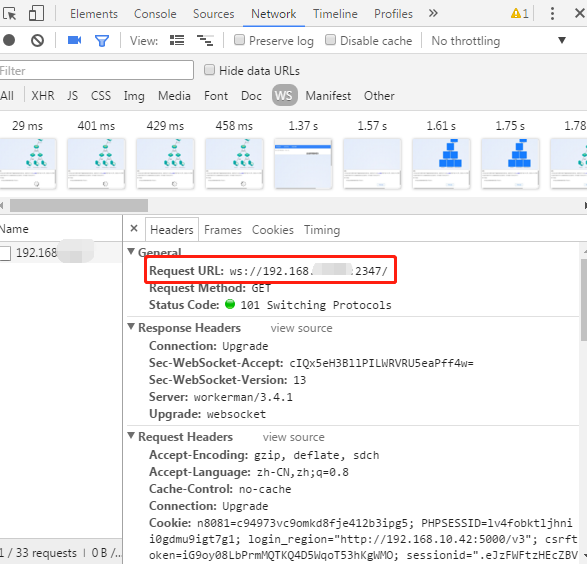
### 四、应用实例：

项目场景：

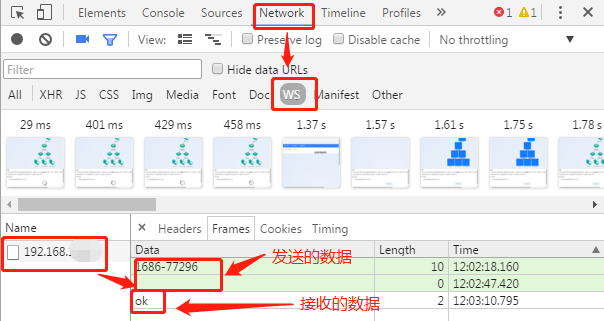
在某项目页面中点击某个按钮之后，会建立一个websocket长连接，实时监控返回的值；

并间隔30s发送心跳数据，保持连接；

通过浏览器的F12获取websocket的连接信息；在network->ws中，查看websocket的连接信息；



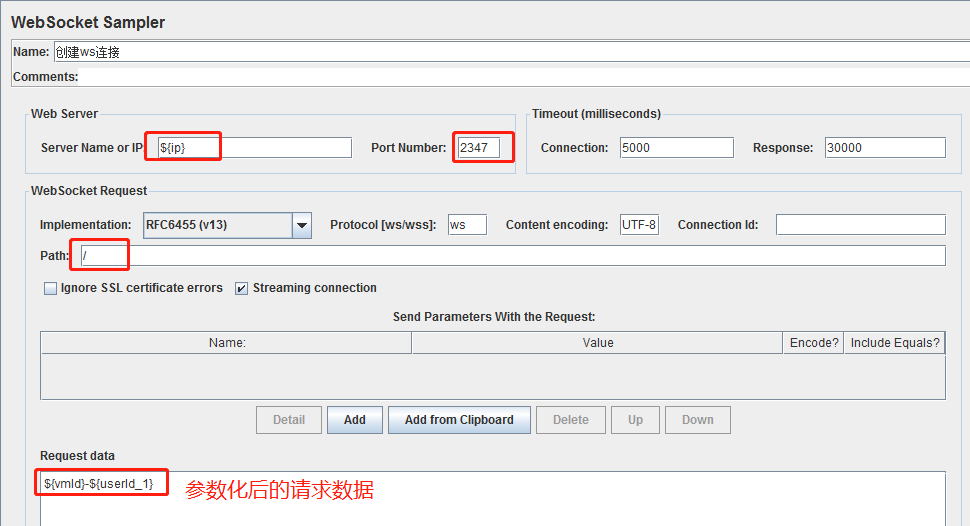
进入Frames标签页，可查看websocket通道发送和接收的数据



其中‘1686-77296’为发送的数据，‘ok’为接收到的响应数据

在发送‘1686-77296’数据之后，会间隔30s发送心跳连接数据，数据内容为空，直到接收到‘ok’后，该websocket通道连接关闭。

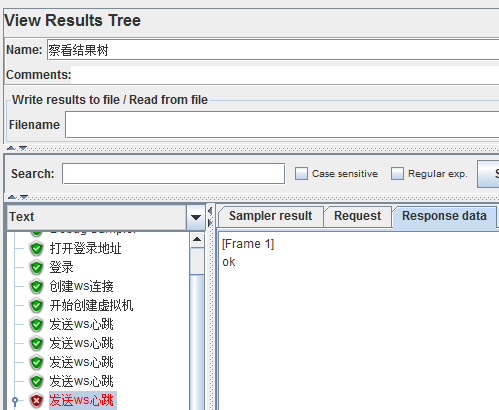
接下来，依据上述信息，在jmeter的websocket sampler中填入对应的值



并利用while 控制器，间隔30s循环发送心跳，直到接收到期望的返回值，跳出while循环；



查看结果树如下所示：



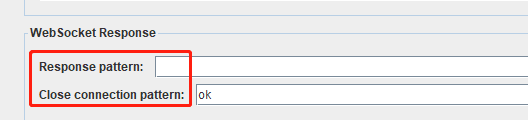
注意事项：

1. Response pattern和Close connection pattern项的设置

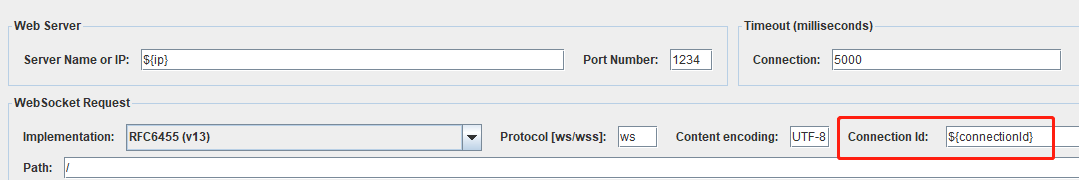
如果同时设置了Response pattern和Close connection pattern，那么，当返回值符合Response pattern设置值，则会直接结束该次通信，而不会执行下面的Close connection

项来关闭连接，即便返回值符合所设置的Close connection pattern

因此，若需要在接受到指定响应，就关闭连接的话，Response pattern应不设置值或设置为不符合返回值的字符，且设置Close connection pattern项为响应的返回值。



2、Connection Id为websocket连接的id号，当存在有多个连接时，必须要设置该选项，定义本次websocket所要连接的id



当进行性能测试，用户循环调用websocket请求时，也需要指定Connection Id，且每次循环应指定不同的Connection Id；

若不指定，就会出现以下问题：

1、若每次循环的websocket未关闭时，再次进行websocket请求，会出现请求repeat的返回值，这是因为，本次循环仍使用的是上一次循环的websocket连接，而上一个循环，已经发送过相同的数据了；

2、若每次循环的websocket均关闭时，会出现连接不上websocket的问题。