**[WebSocket断开原因分析，再也不怕为什么又断开了](https://segmentfault.com/a/1190000014582485)**

[[](https://segmentfault.com/t/chrome) chrome](https://segmentfault.com/t/chrome)[socket.io](https://segmentfault.com/t/socket.io)[javascript](https://segmentfault.com/t/javascript)[websocket](https://segmentfault.com/t/websocket)

发布于 2018-04-25

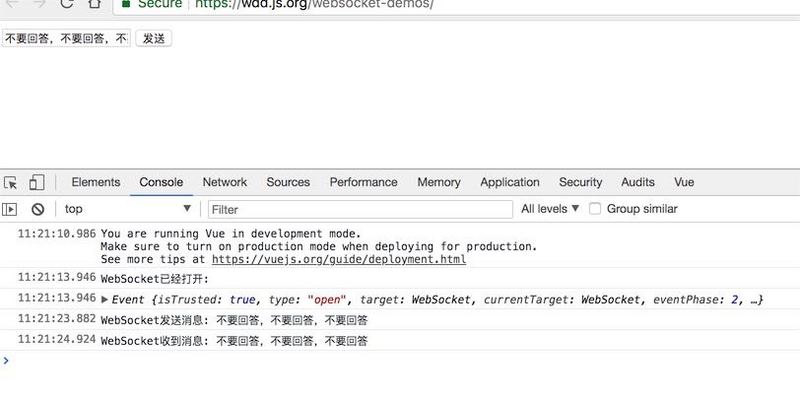


阅读原文：[https://wdd.js.org/websocket-...](https://wdd.js.org/websocket-close-reasons.html" \t "_blank)

**1. 把错误打印出来**

WebSocket断开的原因有很多，最好在WebSocket断开时，将错误打印出来。

在线demo地址：[https://wdd.js.org/websocket-...](https://wdd.js.org/websocket-demos/" \t "_blank)



ws.onclose = **function** (e) {

console.log('websocket 断开: ' + e.code + ' ' + e.reason + ' ' + e.wasClean)

console.log(e)

}

如果你想自己玩玩WebSocket, 但是你又不想自己部署一个WebSocket服务器，你可以使用ws = new WebSocket('wss://echo.websocket.org/'), 你向echo.websocket.org发送消息，它会回复你同样的消息。

**2. 重要信息错误状态码**

WebSocket断开时，会触发CloseEvent, CloseEvent会在连接关闭时发送给使用 WebSockets 的客户端. 它在 WebSocket 对象的 onclose 事件监听器中使用。CloseEvent的code字段表示了WebSocket断开的原因。可以从该字段中分析断开的原因。

CloseEvent有三个字段需要注意, 通过分析这三个字段，一般就可以找到断开原因

* CloseEvent.code: code是错误码，是整数类型
* CloseEvent.reason: reason是断开原因，是字符串
* CloseEvent.wasClean: wasClean表示是否正常断开，是布尔值。一般异常断开时，该值为false



**3. 关闭状态码表**

一般来说1006的错误码出现的情况比较常见，该错误码一般出现在断网时。

| **状态码** | **名称** | **描述** |
| --- | --- | --- |
| 0–999 |  | 保留段, 未使用. |
| 1000 | CLOSE\_NORMAL | 正常关闭; 无论为何目的而创建, 该链接都已成功完成任务. |
| 1001 | CLOSE\_GOING\_AWAY | 终端离开, 可能因为服务端错误, 也可能因为浏览器正从打开连接的页面跳转离开. |
| 1002 | CLOSE\_PROTOCOL\_ERROR | 由于协议错误而中断连接. |
| 1003 | CLOSE\_UNSUPPORTED | 由于接收到不允许的数据类型而断开连接 (如仅接收文本数据的终端接收到了二进制数据). |
| 1004 |  | 保留. 其意义可能会在未来定义. |
| 1005 | CLOSE\_NO\_STATUS | 保留. 表示没有收到预期的状态码. |
| 1006 | CLOSE\_ABNORMAL | 保留. 用于期望收到状态码时连接非正常关闭 (也就是说, 没有发送关闭帧). |
| 1007 | Unsupported Data | 由于收到了格式不符的数据而断开连接 (如文本消息中包含了非 UTF-8 数据). |
| 1008 | Policy Violation | 由于收到不符合约定的数据而断开连接. 这是一个通用状态码, 用于不适合使用 1003 和 1009 状态码的场景. |
| 1009 | CLOSE\_TOO\_LARGE | 由于收到过大的数据帧而断开连接. |
| 1010 | Missing Extension | 客户端期望服务器商定一个或多个拓展, 但服务器没有处理, 因此客户端断开连接. |
| 1011 | Internal Error | 客户端由于遇到没有预料的情况阻止其完成请求, 因此服务端断开连接. |
| 1012 | Service Restart | 服务器由于重启而断开连接. |
| 1013 | Try Again Later | 服务器由于临时原因断开连接, 如服务器过载因此断开一部分客户端连接. |
| 1014 |  | 由 WebSocket标准保留以便未来使用. |
| 1015 | TLS Handshake | 保留. 表示连接由于无法完成 TLS 握手而关闭 (例如无法验证服务器证书). |
| 1016–1999 |  | 由 WebSocket标准保留以便未来使用. |
| 2000–2999 |  | 由 WebSocket拓展保留使用. |
| 3000–3999 |  | 可以由库或框架使用.? 不应由应用使用. 可以在 IANA 注册, 先到先得. |
| 4000–4999 |  | 可以由应用使用. |

**4. 其他注意事项**

如果你的服务所在的域是HTTPS的，那么使用的WebSocket协议也必须是wss, 而不能是ws

**5. 如何在老IE上使用原生WebSocket？**

[web-socket-js](https://github.com/gimite/web-socket-js)是基于flash的技术，只需要引入两个js文件和一个swf文件，就可以让浏览器用于几乎原生的WebSocket接口。另外，web-socket-js还是需要在ws服务端843端口做一个flash安全策略文件的服务。

我自己曾经基于stompjs和web-socket-js，做WebSocket兼容到IE5, 当然了stompjs在低版本的IE上有兼容性问题, 而且stompjs已经不再维护了，你可以使用我fork的一个版本，地址是：[https://github.com/wangduandu...](https://github.com/wangduanduan/stomp-websocket/blob/master/lib/stomp.js" \t "_blank)

主要是老版本IE在正则表达式行为方面有点异常。

*// fix ie8, ie9, RegExp not normal problem*

*// in chrome the frames length will be 2, but in ie8, ie9, it well be 1*

*// by wdd 20180321*

**if** (frames.length === 1) {

**frames**.push('')

}

**6. 参考**

* [CloseEvent](https://developer.mozilla.org/zh-CN/docs/Web/API/CloseEvent)
* [getting the reason why websockets closed with close code 1006](https://stackoverflow.com/questions/19304157/getting-the-reason-why-websockets-closed-with-close-code-1006)
* [Defined Status Codes](https://tools.ietf.org/html/rfc6455#section-7.4.1)
* [Upgrading HTTP to WebSocket](http://enterprisewebbook.com/ch8_websockets.html)