[介绍 2](#_Toc375160914)

[环境配置 2](#_Toc375160915)

[客户端 2](#_Toc375160916)

[服务器端 2](#_Toc375160917)

[基本操作 3](#_Toc375160918)

[已解决的问题 7](#_Toc375160919)

[1. 客户端找不到socket.io.js 7](#_Toc375160920)

[问题描述 7](#_Toc375160921)

[原因分析 7](#_Toc375160922)

[解决方案 7](#_Toc375160923)

[参考资料 8](#_Toc375160924)

# 介绍

[socket.io](http://socket.io/)是一个以实现跨浏览器、跨平台的实时应用为目的的项目。针对不同的浏览器版本或者不同客户端会做自动降级处理，选择合适的实现方式（websocket, long pull..），隐藏实现只暴露统一的接口。可以让应用只关注于业务层面上。

socket.io设计的目标是支持任何的浏览器，任何Mobile设备。目前支持主流的PC浏览器 (IE,Safari,Chrome,Firefox,Opera等)，Mobile浏览器(iphone Safari/ipad Safari/android WebKit/WebOS WebKit等)。socket.io基于node.js并简化了WebSocket API，统一了通信的API。它支持：WebSocket, Flash Socket, AJAX long-polling, AJAX multipart streaming, Forever IFrame, JSONP polling。

# 环境配置

## 客户端

在客户端的话，可以通过

**<script** src="/socket.io/socket.io.js"**></script>**

或者引用socket.io的CDN服务。

**<script** src="http://cdn.socket.io/stable/socket.io.js"**></script>**

## 服务器端

Node.js中安装：

npm install socket.io

安装之后就可以require模块来使用了:

**var** sio = require('socket.io');

# 基本操作

终止连接：

var socket = io.connect("http://localhost:8001");

…

socket.disconnect();

1. 启动服务器还是交给node，打开命令行窗口，定位到server.js所在文件夹，输入node server.js启动服务器。

    在上面的index.html中，注意这行：<script src="/socket.io/socket.io.js"></script>。如果不想使用本地的socket.io脚本，可以直接使用下面这个公开的脚本：

|  |
| --- |
| <script src="<http://cdn.socket.io/stable/socket.io.js>"></script> |

 此外需要注意这行：socket = io.connect(null)。这里的null代表连接本地服务，可以换成"localhost"，效果也是一样的。

2. 可以使用socket.io直接启动http服务。例如：

|  |
| --- |
| var io = require('socket.io').listen(80);    io.sockets.on('connection', function (socket) {      io.sockets.emit('this', { will: 'be received by everyone'});  }); |

 3. socket.io可以直接通过send方法发送消息，使用message事件接收消息，例如：

|  |
| --- |
| //server.js  var io = require('socket.io').listen(80);    io.sockets.on('connection', function (socket) {      socket.on('message', function () { });  });    //index.html  <script>  var socket = io.connect('<http://localhost/>');  socket.on('connect', function () {      socket.send('hi');      socket.on('message', function (msg) {          // my msg      });  });  </script> |

 4. 发送和处理数据

    两端可以互发事件，互发数据，相互通信。发送事件的代码为：socket.emit(action, data, function)，其中action为事件的名称，data为数据，function为回调函数;处理事件代码 为：socket.on(action,function)，如果emit发送的时候有数据data，则function中参数包含了这个数据。 socket.io除了发送和处理内置事件，如connect, disconnect, message。还允许发送和处理自定义事件，例如:

|  |
| --- |
| //服务端：  io.sockets.on('connection', function (socket) {      socket.emit('news', { hello: 'world' });      socket.on('my other event', function (data) {          console.log(data);      });  });    //客户端：  <script src="/socket.io/socket.io.js"></script>  <script>  var socket = io.connect('[http://localhost](http://localhost/)');  socket.on('news', function (data) {      console.log(data);      socket.emit('my other event', { my: 'data' });  });  </script> |

 5. 从上面可以看出来，发送数据的时候，send和emit是都可以使用的。只不过emit更是强化了自定义事件的处理。

6. 可以在服务端使用socket的get/set方法存储客服端的相关数据，例如：

|  |
| --- |
| //服务端  var io = require('socket.io').listen(80);    io.sockets.on('connection', function (socket) {      socket.on('set nickname', function (name) {          socket.set('nickname', name, function () { socket.emit('ready'); });      });        socket.on('msg', function () {          socket.get('nickname', function (err, name) {              console.log('Chat message by ', name);          });      });  });  //客户端  <script>  var socket = io.connect('[http://localhost](http://localhost/)');    socket.on('connect', function () {      socket.emit('set nickname', confirm('What is your nickname?'));      socket.on('ready', function () {          console.log('Connected !');          socket.emit('msg', confirm('What is your message?'));      });  });  </script> |

# 已解决的问题

## 客户端找不到socket.io.js

### 问题描述

我先从网上下载了一个socket.io.js，结果客户端io没有connect方法！这估计是因为下载的是0.6版本，而最新的是0.9版本。

官网上的示例是直接在客户端引入：

<script src="/socket.io/socket.io.js"></script>

但是这样却找不到socket.io.js！

### 原因分析

这是因为socket.io.js是包含在node.js下载的socket.io模块中的，它位于（如果监听的是8080端口）<http://localhost:8080/socket.io/socket.io.js>

### 解决方案

1. 硬编码为：<script src="<http://localhost:8080/socket.io/socket.io.js>"></script>

这样每次更改端口时，都需要更改src。且在部署时也很麻烦（如要手动改域名等）。

1. 在浏览器中打开<http://localhost:8080/socket.io/socket.io.js>文件，将该文件保存为静态文件socket.io.js，再在客户端引用。

# 参考资料

[官网](http://socket.io/#home)

待补充