添加一个请求头

response.addHeader(**"Access-Control-Allow-Origin"**, **"\*"**);

\*可以改成指定域名

表示指定域名可以跨域请求

在HTML5中有一种新的跨域方式，即设置“Access-Control-Allow-Origin”可以指定允许跨域访问的域名。   
Node.js中可以这样写

app.all(‘\*’, function(req, res, next) {   
res.header(“Access-Control-Allow-Origin”, ‘[https://www.google.com](https://www.google.com/)‘);   
res.header(‘Access-Control-Allow-Methods’, ‘POST, GET’);   
res.header(‘Access-Control-Allow-Headers’, ‘X-Requested-With’);   
res.header(‘Access-Control-Allow-Headers’, ‘Content-Type’);   
next();   
});   
但在实际使用中，可能需要设置多个域名。   
在网上查询资料，有一种写法

Access-Control-Allow-Origin: https://www.google.com,https://www.baidu.com   
改成相应的代码，但是始终提示问题，只允许一个值。

于是换种方式，查资料发现，许多是先自己判断域名是否是允许的，如果是再设置允许跨域访问。那么代码应该改成如下形式

app.all(‘\*’, function(req, res, next) {   
if( req.headers.origin == ‘[https://www.google.com](https://www.google.com/)’ || req.headers.origin == ‘[https://www.baidu.com](https://www.baidu.com/)’ ){   
res.header(“Access-Control-Allow-Origin”, req.headers.origin);   
res.header(‘Access-Control-Allow-Methods’, ‘POST, GET’);   
res.header(‘Access-Control-Allow-Headers’, ‘X-Requested-With’);   
res.header(‘Access-Control-Allow-Headers’, ‘Content-Type’);   
}   
next();   
});   
使用时可以把允许访问的域名写成一个数组，然后JS写一个比较字符串是否在数组内的函数，这样使用就比较方便了。

其中，Access-Control-Allow-Origin就是我们需要设置的域名，Access-Control-Allow-Methods是允许的请求方式，Access-Control-Allow-Headers跨域允许包含的头。

——————–java—————————-   
public void doFilter(ServletRequest req, ServletResponse res,   
FilterChain chain) throws IOException, ServletException {   
HttpServletResponse httpResponse = (HttpServletResponse) res;   
String [] allowDomain= {“[http://132.12.11.11:8888](http://132.12.11.11:8888/)“,”[http://123.112.112.12:80](http://123.112.112.12/)“,”[http://123.16.12.23](http://123.16.12.23/)“,”[http://121.12.18.13:10195](http://121.12.18.13:10195/)“};   
Set allowedOrigins= new HashSet(Arrays.asList(allowDomain));   
String originHeader=((HttpServletRequest) req).getHeader(“Origin”);   
if (allowedOrigins.contains(originHeader)){   
((HttpServletResponse) res).setHeader(“Access-Control-Allow-Origin”, originHeader);   
((HttpServletResponse) res).setContentType(“application/json;charset=UTF-8”);   
((HttpServletResponse) res).setHeader(“Access-Control-Allow-Methods”, “POST, GET, OPTIONS, DELETE”);   
((HttpServletResponse) res).setHeader(“Access-Control-Max-Age”, “3600”);   
((HttpServletResponse) res).setHeader(“Access-Control-Allow-Headers”, “Origin, No-Cache, X-Requested-With, If-Modified-Since, Pragma, Last-Modified, Cache-Control, Expires, Content-Type, X-E4M-With,userId,token”);//表明服务器支持的所有头信息字段   
((HttpServletResponse) res).setHeader(“Access-Control-Allow-Credentials”, “true”); //如果要把Cookie发到服务器，需要指定Access-Control-Allow-Credentials字段为true;   
((HttpServletResponse) res).setHeader(“XDomainRequestAllowed”,”1”);   
}   
chain.doFilter(req, res);   
return;   
}