跨域的解决方案：

1. JSONP(前后端一起配合)

2. CORS(cross origin resource share)：在http的response加一个header： Access-Control-Allow-Origin，就能实现ajax跨域

xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

属于前端配置的与CORS无关

3. 反向代理

CORS与JSONP的使用目的相同，但是比JSONP更强大。

JSONP只支持GET请求，CORS支持所有类型的HTTP请求。JSONP的优势在于支持老式浏览器，

以及可以向不支持CORS的网站请求数据。

同源政策的目的，是为了保证用户信息的安全，防止恶意的网站窃取数据。目前，如果非同源，

共有三种行为受到限制。

（1） Cookie、LocalStorage 和 IndexDB 无法读取。

（2） DOM 无法获得。

（3） AJAX 请求不能发送。

JSONP原理：根据浏览器同源策略，所谓同源就是协议、主机、端口号都相同时成为同源。a 域的js不能直接访问 b域名的信息，script 标签的src属性可以跨域引用文件，jsonp是请求之后后台包装好一段json，并且把数据放在一个callback函数，返回一个js文件，动态引入这个文件，下载完成js之后，会去调用这个callback,通过这样访问数据。

JSONP是服务器与客户端跨源通信的常用方法。最大特点就是简单适用，老式浏览器全部支持，服务器改造非常小。

JSONP利用script标签不受同源策略限制，来发送跨域请求

流程：

1. 创建一个script标签

2. 设定该标签的src属性(按照服务器的要求来设定)

3. 将该标签上DOM树

===> 该请求就会发送给服务器,服务器会响应会一段JS代码

===> 代码内容为一个函数调用(该函数名，是我们传递给服务器的)，函数的实参，就是我们要取得的数据

CORS是跨源资源分享（Cross-Origin Resource Sharing）的缩写。它是W3C标准，是跨源AJAX请求的根本解决方法。相比JSONP只能发GET请求，CORS允许任何类型的请求。

它允许浏览器向跨源服务器，发出XMLHttpRequest请求，从而克服了AJAX只能同源使用的限制。

CORS需要浏览器和服务器同时支持。目前，所有浏览器都支持该功能，IE浏览器不能低于IE10。