### 1、服务器：

*//详细的内容看t\_get函数说明* **public function** t\_set(){  
 **if**(**isset**($\_SERVER[**"HTTP\_ORIGIN"**])){  
 *header*(**'Access-Control-Allow-Origin: '**.$\_SERVER[**"HTTP\_ORIGIN"**]);  
 }**else**{  
 $IP = $this->getIPAddress(); *//使用jsonp的时候没有HTTP\_ORIGIN，这样设置源，不统一设为\*的原因是使用了credentials，不能设为\*  
 header*(**'Access-Control-Allow-Origin: '**.$IP);  
 }  
 *header*(**'Access-Control-Allow-credentials: true'**);  
  
 *session\_start*();  
 $session\_name = *ini\_get*(**"session.name"**);  
 $session\_expiration = *ini\_get*(**"session.cookie\_lifetime"**);  
 **if**(*intval*($session\_expiration) == 0){  
 $session\_expiration = **"Session"**;  
 $gmt = **"Session"**;  
 }**else**{  
 $gmt = *gmdate*(**"D, d-M-Y H:i:s"**, *strtotime*(**"+"**.$session\_expiration.**" second"**, *time*())) . **" GMT"**;  
 }  
 $session\_id = *session\_id*();  
 *header*(**'Set-Cookie: '**.$session\_name.**'='**.$session\_id.**'; expires='**.$gmt.**'; Max-Age='**.$session\_expiration.**'; path=127.0.0.1:8086/; HttpOnly; '**);  
  
 $\_SESSION[**"t\_bt"**] = *date*(**"Y-m-d H:i:s"**);  
 $this->**load**->helper(**"check"**);  
 $jsonpType = $this->**input**->get(**"jsonpcallback"**);  
 **if**(!isNull($jsonpType)){  
 log\_message(**"error"**, $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"session\_id"** => $session\_id, **"mgt"** => $gmt)).**")"**);  
 **echo** $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"session\_id"** => $session\_id, **"mgt"** => $gmt)).**")"**;  
 }**else**{  
 **echo** *json\_encode*(**array**(**"error"** => **"0"**, **"session\_id"** => $session\_id, **"mgt"** => $gmt));  
 }  
 }  
  
 *//假如现在是192.168.11.129:9001请求192.168.5.159:80* **public function** t\_get(){  
 */\* （问题1）直接简单ajax跨站请求被拦截  
 \* Access to XMLHttpRequest at 'http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_set' from origin 'http://192.168.11.129:9001' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource.  
 \* \*/  
// header('Access-Control-Allow-Origin: \*'); //必须要设，不然有（1）  
  
 //还是无法处理session，网上说ajax需要设置  
 // crossDomain: true,  
 // xhrFields: {  
 // withCredentials: true  
 // },  
 /\* （问题2）添加cros之后不能设为\*号  
 \* Access to XMLHttpRequest at 'http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_get' from origin 'http://192.168.11.129:9001' has been blocked by CORS policy: The value of the 'Access-Control-Allow-Origin' header in the response must not be the wildcard '\*' when the request's credentials mode is 'include'. The credentials mode of requests initiated by the XMLHttpRequest is controlled by the withCredentials attribute.  
 \* \*/* **if**(**isset**($\_SERVER[**"HTTP\_ORIGIN"**]))  
 {  
 *header*(**'Access-Control-Allow-Origin: '**.$\_SERVER[**"HTTP\_ORIGIN"**]);  
 }  
 **else** {  
 $IP = $this->getIPAddress(); *//使用jsonp的时候没有HTTP\_ORIGIN，这样设置源，不统一设为\*的原因是使用了credentials，不能设为\*  
 header*(**'Access-Control-Allow-Origin: '**.$IP);  
 }  
 *header*(**'Access-Control-Allow-credentials: true'**);  
 *//还是无法处理session，可以在响应头中看到Set-Cookie是感叹号  
 /\* （问题3）chrome无法保存cookie  
 \* This Set-Cookie didn't specify a "SameSite" attribute and was defaulted to "SameSite=Lax," and was blocked because it came from a cross-site response which was not the response to a top-level navigation. The Set-Cookie had to have been set with "SameSite=None" to enable cross-site usage.  
 \* \*/  
 //SameSite 有三个值，Strict、Lax、None  
 //Strict最严格，完全禁止第三方cookie，跨站点时，任何情况下都不会发送 Cookie。所以不考虑  
 //Lax稍宽松，大多数情况也是不发送第三方 Cookie，但是导航到目标网址的 Get 请求除外。我们用post请求所以也不考虑  
 //None，显式关闭SameSite属性，前提是必须同时设置Secure属性（Cookie 只能通过 HTTPS 协议发送），否则无效  
 //这就是导致chrome跨站的必须使用https的原因，不然无法处理session  
 session\_start*();  
 */\* （问题6）  
 \* cookie过期时间  
 \* \*/* $session\_name = *ini\_get*(**"session.name"**); *//不是要config->item的sess\_cookie\_name，因为这里不是使用ci的session library，session名称是apache中的名称，所以过期时间也不是config中的值，是ini中的session.cookie\_lifetime，而这个值默认是0，  
 //已验证，Max-Age设为-1，是没用的，负数和0都是马上删除了，设为10，就10秒后删除，所以没办法做到浏览器关闭旧删除，通过观察其他页面的session，它的Expires/Max-Age 值是“Session”，那么我们这里也改为Session，发现Max-Age值是Session时，会使用Expires的值存储，不设置Expires时存储就是“Session”，设为“Session”就可以做到浏览器关闭删除cookie  
 //由于现在没有准入环境，不知道环境的apache是什么配置，但应该就是0，因为记得关闭浏览器就是重新登录的* $session\_expiration = *ini\_get*(**"session.cookie\_lifetime"**);  
 **if**(*intval*($session\_expiration) == 0){  
 $session\_expiration = **"Session"**;  
 $gmt = **"Session"**;  
 }**else**{  
 $gmt = *gmdate*(**"D, d-M-Y H:i:s"**, *strtotime*(**"+"**.$session\_expiration.**" second"**, *time*())) . **" GMT"**;  
 }  
 $session\_id = *session\_id*();  
 *//不设置path，这样path会是/Guest，就不会和/冲突，因为设为/之后，这个cookie是必须https的，就会导致网关无法使用http了  
 header*(**'Set-Cookie: '**.$session\_name.**'='**.$session\_id.**'; expires='**.$gmt.**'; Max-Age='**.$session\_expiration.**'; Secure; SameSite=None; '**);  
 *//不设置为path的时候，我这边请求https://192.168.5.159:4433/gateway/IPG-16875/index.php/guest/t\_get，cookie的path是"/gateway/IPG-16875/index.php/guest"  
 //假如当前sessionid是u2nu26ibb0tgvr531f46c0283c，这样之后，因为path的原因：  
 //在A站的cookies中不会有该值  
 //在https://192.168.5.159:4433上也不会有该值  
 //在https://192.168.5.159:4433/gateway/IPG-16875/index.php/guest/上有  
 //在https://192.168.5.159:4433/gateway/IPG-16875/index.php/gateway/上没有该值  
 //所以当A站的ajax请求/guest/xxx，就会获取到改sessionid来发送请求  
  
 //以上，就是chrome的实现  
 //由于证书是自签的，在A站ajax请求网关的https地址，会无法访问，提示net::ERR\_CERT\_AUTHORITY\_INVALID，需要先用浏览器访问一次https的网关  
 //或者net::ERR\_CERT\_COMMON\_NAME\_INVALID也是这样  
  
 /\* 扩展  
 \* cookie中max-age属性同expires属性的不同点，如下所示:  
 不同点1:出现时间及兼容性不同：  
 max-age：是新属性，所以低版本的浏览器不支持此属性  
 expires: 是旧属性，可支持目前市面上所有浏览器  
  
 不同点2:设置方法不同：  
 max-age：后面属性值为一个秒数的整型  
 expires: 属性值为一个UTC的具体时间值  
  
 不同点3:优先级不同：  
 max-age优先级高于  
 expires  
 如果同时设置了两个属性，则以max-age为准  
  
 不同点4:运行环境受影响不同：  
 max-age:是以客户端浏览器过多久时间后，cookie过期  
 expires:是客户端浏览器到达指定时间后，cookie过期  
 那么如果客户端浏览器的时间设置错误，则会导致expires过期时间不是我们想要的效果  
  
 \* \*/  
  
 //在上方代码的基础上，测试一下火狐和IE  
  
 //火狐81.0.2，请求http的http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_get，会报  
 /\*  
 \* 由于 Cookie “request-test”的“SameSite”属性设置为“None”，但缺少“Secure”属性，此 Cookie 未来将被拒绝。若要了解“SameSite“的更多信息，请参阅：https://developer.mozilla.org/docs/Web/HTTP/Headers/Set-Cookie/SameSite request.html:70:2  
由于非 HTTPS Cookie 无法设置“secure”属性，已拒绝 Cookie “PHPSESSID”。  
 \* \*/  
 //所以在这个设置下，火狐也必须请求https的网关  
 //同样，不事前访问https的网关，火狐会报  
 /\*  
 \* 已拦截跨源请求：同源策略禁止读取位于 https://192.168.5.159:4433/gateway/IPG-16875/index.php/guest/t\_get 的远程资源。（原因：CORS 请求未能成功）。  
 \* \*/  
 //所以会没有响应头，没有响应内容，被拦截了。然后访问一下https的网关  
 //ajax就可以请求成功了，并且正常使用session  
  
 //ie8，请求http的http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_get，通过输出失败原因，  
 /\*  
 \* {"readyState":0,"status":0,"statusText":"No Transport"}  
 \* \*/  
 //网上查找资料，是ie8不支持CORS协议，所以改为用jsonp的方式  
 /\* （问题4）  
 \* ie8不支持CORS协议  
 \* \*/  
 /\* 解决方法：  
 dataType:'jsonp',  
 jsonp:'xxx',  
 \* \*/  
 //并且jsonp只能使用get方式* $this->**load**->helper(**"check"**);  
 $jsonpType = $this->**input**->get(**"jsonpcallback"**);  
 **if**(!isNull($jsonpType)){  
 log\_message(**"error"**, $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"data"** => $\_SESSION[**"t\_bt"**], **"session\_id"** => $session\_id)).**")"**);  
 **echo** $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"data"** => $\_SESSION[**"t\_bt"**], **"session\_id"** => $session\_id, **"param"**=>$this->**input**->get())).**")"**;  
 }**else**{  
 **echo** *json\_encode*(**array**(**"error"** => **"0"**, **"data"** => $\_SESSION[**"t\_bt"**], **"session\_id"** => $session\_id));  
 }  
 *//改为jsonp方式后正常了，sessionid还是会变化，无法使用session  
  
 //改为https请求，请求失败，甚至乎没有打印上面的日志，输出ajax的错误：  
 /\*  
 \* {"readyState":4,"status":200,"statusText":"success"}  
 \* \*/  
 /\* （问题5）  
 \* 浏览器上方显示“为了保护您的安全，Internet Explorer已阻止此网站显示有安全证书错误的内容。单击此处查看选项”  
 \* \*/  
 //点击它，再点击“有何风险？”，根据提示，就是要点击“继续浏览此网站(不推荐)”，但我使用https访问网关了，还是不行  
 //最后是通过导入apache根证书到受信任的根证书颁发机构，就可以请求了，并且解决了在ie8中请求https很慢的情况，未导入的时候是很慢的  
 //session正常使用  
  
 //总结： chrome在跨站必须使用https。  
 // chrome和firefox都需要先用https访问一次，然后信任证书，才能发起请求  
 // ie8即使点击了继续浏览还是不行，需要证书受信任，以前测试为什么成功也不知道了  
 // 因为设置Secure的原因，所有都需要https请求* }

### 2、js

<**html**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>test</**title**>  
</**head**>  
<**body**>  
hello<**br**>  
<**input type="text" id="prefix" value="http://192.168.5.159/gateway/IPG-16875/" style="width**: 100%;**"**><**br**>  
<**input type="text" id="function" placeholder="url" style="width**: 100%;**"**/><**br**>  
<**input type="text" id="type" placeholder="type" value="POST" style="width**: 100%;**"**/><**br**>  
<**textarea type="text" id="params" placeholder="param" style="width**: 100%;**"** >{}</**textarea**><**br**>  
<**input type="text" id="datetype" placeholder="dataType" style="width**: 100%;**" value="json"**><**br**>  
<**input type="text" id="header" placeholder="header" value="application/json" style="width**: 100%;**"**><**br**>  
<**input type="text" id="xdebug" placeholder="xdebug" style="width**: 100%;**"**><**br**>  
<**button id="request1"**>cors request</**button**>  
<**button id="request2"**>jsonp request</**button**><**br**>  
<**span id="res"**></**span**>  
  
<**script src="js/jquery.min.js"**></**script**>  
*<!--ie8使用JSON-->*<**script src="https://cdn.bootcdn.net/ajax/libs/json2/20160511/json2.js"**></**script**>  
<**script**>  
  
 **var *conf*** = *getCookie*(**"request-test"**);  
 **if**(***conf***){  
 ***conf*** = ***JSON***.parse(***conf***);  
 **$**(**"#prefix"**).val(***conf***.**prefix**);  
 **$**(**"#function"**).val(***conf***.**func**);  
 **$**(**"#xdebug"**).val(***conf***.**xdebug**);  
 **$**(**"#type"**).val(***conf***.**type**);  
 **$**(**"#params"**).val(***conf***.**params**);  
 **$**(**"#datetype"**).val(***conf***.**datatype**);  
 **$**(**"#header"**).val(***conf***.**header**);  
 }  
  
 *//跨域的时候，如果有application/json，被认为是复杂请求，https://blog.csdn.net/yasha009/article/details/78784620* **$**(**"#request1"**).click(**function**(){  
 **$**(**"#res"**).**html**(**""**);  
  
 *setCookie*(**"request-test"**, ***JSON***.stringify({  
 **prefix**: **$**(**"#prefix"**).val(),  
 **func**: **$**(**"#function"**).val(),  
 **xdebug**: **$**(**"#xdebug"**).val(),  
 **type**: **$**(**"#type"**).val(),  
 **params**: **$**(**"#params"**).val(),  
 **datatype**: **$**(**"#datetype"**).val(),  
 **header**: **$**(**"#header"**).val()  
 }), 7);  
  
 **$**.**ajax**({  
 *//参数* **url**: **$**(**"#prefix"**).val()+**$**(**"#function"**).val()+**$**(**"#xdebug"**).val(),  
 **type**: **$**(**"#type"**).val(),  
 **data**: ***JSON***.parse(**$**(**"#params"**).val()),  
 **dataType**: **$**(**"#datetype"**).val(),  
 **crossDomain**: **true**,  
 **xhrFields**: {  
 **withCredentials**: **true** },  
 success: **function**(res){  
 *// console.log(res);* ***console***.log(***JSON***.stringify(res));  
 **$**(**"#res"**).**html**(***JSON***.stringify(res));  
 },  
 error: **function**(err){  
 *// console.log(err);* ***console***.log(***JSON***.stringify(err));  
 **$**(**"#res"**).**html**(***JSON***.stringify(err));  
 }  
 });  
 });  
  
 **$**(**"#request2"**).click(**function**(){  
 **$**(**"#res"**).**html**(**""**);  
  
 *setCookie*(**"request-test"**, ***JSON***.stringify({  
 **prefix**: **$**(**"#prefix"**).val(),  
 **func**: **$**(**"#function"**).val(),  
 **xdebug**: **$**(**"#xdebug"**).val(),  
 **type**: **$**(**"#type"**).val(),  
 **params**: **$**(**"#params"**).val(),  
 **datatype**: **$**(**"#datetype"**).val(),  
 **header**: **$**(**"#header"**).val()  
 }), 7);  
  
 **$**.**ajax**({  
 *//参数* **url**: **$**(**"#prefix"**).val()+**$**(**"#function"**).val()+**$**(**"#xdebug"**).val()+**"?a=b2"**,  
 **type**: **"GET"**,  
 **dataType**:**'jsonp'**,  
 **jsonp**:**'jsonpcallback'**,  
 success: **function**(res){  
 *// console.log(res);* ***console***.log(***JSON***.stringify(res));  
 **$**(**"#res"**).**html**(***JSON***.stringify(res));  
 },  
 error: **function**(err){  
 *// console.log(err);* ***console***.log(***JSON***.stringify(err));  
 **$**(**"#res"**).**html**(***JSON***.stringify(err));  
 }  
 });  
 });  
  
 **function** *setCookie*(cname, cvalue, exdays) {  
 **var** d = **new *Date***();  
 d.setTime(d.getTime() + (exdays \* 24 \* 60 \* 60 \* 1000));  
 **var** expires = **"expires="**+d.toUTCString();  
 ***document***.**cookie** = cname + **"="** + cvalue + **";"** + expires + **";path=/"**;  
 }  
  
 **function** *getCookie*(cname) {  
 **var** name = cname + **"="**;  
 **var** ca = ***document***.**cookie**.split(**';'**);  
 **for**(**var** i = 0; i < ca.**length**; i++) {  
 **var** c = ca[i];  
 **while** (c.charAt(0) == **' '**) {  
 c = c.substring(1);  
 }  
 **if** (c.indexOf(name) == 0) {  
 **return** c.substring(name.**length**, c.**length**);  
 }  
 }  
 **return ""**;  
 }  
  
</**script**>  
</**body**>  
</**html**>

### 3、快查

3.1 Access to XMLHttpRequest at 'http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_set' from origin 'http://192.168.11.129:9001' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource.

解决：

header('Access-Control-Allow-Origin: \*');

3.2 ajax不携带其他站保存的cookie

解决：

crossDomain: true,  
 xhrFields: {  
 withCredentials: true  
 },

3.3 Access to XMLHttpRequest at 'http://192.168.5.159/gateway/IPG-16875/index.php/guest/t\_get' from origin 'http://192.168.11.129:9001' has been blocked by CORS policy: The value of the 'Access-Control-Allow-Origin' header in the response must not be the wildcard '\*' when the request's credentials mode is 'include'. The credentials mode of requests initiated by the XMLHttpRequest is controlled by the withCredentials attribute.

解决：

if(isset($\_SERVER["HTTP\_ORIGIN"]))

{

header('Access-Control-Allow-Origin: '.$\_SERVER["HTTP\_ORIGIN"]);

}

else

{

$IP = $this->getIPAddress(); //使用jsonp的时候没有HTTP\_ORIGIN，这样设置源，不统一设为\*的原因是使用了credentials，不能设为\*

header('Access-Control-Allow-Origin: '.$IP);

}

header('Access-Control-Allow-credentials: true');

3.4 This Set-Cookie didn't specify a "SameSite" attribute and was defaulted to "SameSite=Lax," and was blocked because it came from a cross-site response which was not the response to a top-level navigation. The Set-Cookie had to have been set with "SameSite=None" to enable cross-site usage.

解决：

$session\_name = ini\_get("session.name");

$session\_expiration = ini\_get("session.cookie\_lifetime");

if(intval($session\_expiration) == 0){

$session\_expiration = "Session";

$gmt = "Session";

}else{

$gmt = gmdate("D, d-M-Y H:i:s", strtotime("+".$session\_expiration." second", time())) . " GMT";

}

$session\_id = session\_id();

header('Set-Cookie: '.$session\_name.'='.$session\_id.'; expires='.$gmt.'; Max-Age='.$session\_expiration.'; Secure; SameSite=None; ');

3.5 ie8 {"readyState":0,"status":0,"statusText":"No Transport"}

解决：无法使用CORS协议

dataType:'jsonp',

jsonp:'jsonpcallback',

$jsonpType = $this->input->get("jsonpcallback");

if(!isNull($jsonpType)){

echo $jsonpType."(".json\_encode(array("error" => "0")).")";

}else{

echo json\_encode(array("error" => "0"));

}

3.6 ie8浏览器上方显示“为了保护您的安全，Internet Explorer已阻止此网站显示有安全证书错误的内容。单击此处查看选项”

解决：导入根证书到受信任

### 4、取巧方法

上方的是因为要设置SameSite为None，所以必须https请求，如果是http的话，这里通过服务器返回sessionid，由js自己存储cookie到自己站内，存储之后，请求其他站是不会携带这个cookie的，所以需要读站内cookie，把值添加到请求头中，服务器根据请求头的值来开启session

**public function** t\_set2(){  
 *header*(**"Access-Control-Allow-Headers: TECSESSID"**);  
 *header*(**"Access-Control-Expose-Headers: TECSESSID"**);  
 **if**(**isset**($\_SERVER[**"HTTP\_ORIGIN"**])){  
 *header*(**'Access-Control-Allow-Origin: '**.$\_SERVER[**"HTTP\_ORIGIN"**]);  
 }**else**{  
 $IP = $this->getIPAddress(); *//使用jsonp的时候没有HTTP\_ORIGIN，这样设置源，不统一设为\*的原因是使用了credentials，不能设为\*  
 header*(**'Access-Control-Allow-Origin: '**.$IP);  
 }  
 *header*(**'Access-Control-Allow-credentials: true'**);  
  
 **if**(**isset**($\_SERVER[**"HTTP\_TECSESSID"**])){  
 *session\_id*($\_SERVER[**"HTTP\_TECSESSID"**]);  
 }  
  
 *session\_start*();  
 $session\_id = *session\_id*();  
  
 $\_SESSION[**"t\_bt"**] = *date*(**"Y-m-d H:i:s"**);  
 $this->**load**->helper(**"check"**);  
 $jsonpType = $this->**input**->get(**"jsonpcallback"**);  
 **if**(!isNull($jsonpType)){  
 **echo** $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"session\_id"** => $session\_id)).**")"**;  
 }**else**{  
 **echo** *json\_encode*(**array**(  
 **"error"** => **"0"**,  
 **"session\_id"** => $session\_id  
 ));  
 }  
}  
  
*//在getLoginFormConf中返回分配的sessionid，控件存起来，存储时间是浏览器关闭就删除  
//  
//如果有提交TECSESSID，就使用它来作为session\_id，如果没有，就由程序判断有没有Cookie***public function** t\_get2(){  
 *header*(**"Access-Control-Allow-Headers: TECSESSID"**);  
 *header*(**"Access-Control-Expose-Headers: TECSESSID"**); *//没必要返回header，js获取不到，输出null* **if**(**isset**($\_SERVER[**"HTTP\_ORIGIN"**])){  
 *header*(**'Access-Control-Allow-Origin: '**.$\_SERVER[**"HTTP\_ORIGIN"**]);  
 }**else**{  
 $IP = $this->getIPAddress(); *//使用jsonp的时候没有HTTP\_ORIGIN，这样设置源，不统一设为\*的原因是使用了credentials，不能设为\*  
 header*(**'Access-Control-Allow-Origin: '**.$IP);  
 }  
 *header*(**'Access-Control-Allow-credentials: true'**);  
  
 log\_message(**"error"**, $\_SERVER[**"HTTP\_TECSESSID"**]);  
  
 **if**(**isset**($\_SERVER[**"HTTP\_TECSESSID"**])){  
 *session\_id*($\_SERVER[**"HTTP\_TECSESSID"**]);  
 }  
  
 *session\_start*();  
 $session\_id = *session\_id*();  
  
 $this->**load**->helper(**"check"**);  
 $jsonpType = $this->**input**->get(**"jsonpcallback"**);  
 **if**(!isNull($jsonpType)){  
 **echo** $jsonpType.**"("**.*json\_encode*(**array**(**"error"** => **"0"**, **"data"** => $\_SESSION[**"t\_bt"**], **"session\_id"** => $session\_id)).**")"**;  
 }**else**{  
 **echo** *json\_encode*(**array**(  
 **"error"** => **"0"**,  
 **"data"** => $\_SESSION[**"t\_bt"**],  
 **"session\_id"** => $session\_id  
 ));  
 }  
}

控件在加载前都是会先请求一次getLoginFormConf获取一些配置的，在这里额外返回一个sessionid，控件收到之后，保存到cookie中，有效期是关闭浏览器删除。  
控件在加载之后，发出的任何请求都添加自定义头，TECSESSID:xxx  
当服务器收到$\_SEVER["HTTP\_TECSESSID"]，就使用该值作为sessionid，session\_id("xxx");，然后再开启session，如果没有该值，就直接开启session，由程序自己判断有没有提交Cookie。  
通过js来保存cookie，就不会出现跨域无法保存cookie到当前域的情况。

前端：

<html>

<head>

<meta charset="UTF-8">

<title>test</title>

</head>

<body>

hello<br>

<input type="text" id="prefix" value="http://192.168.5.159/gateway/IPG-16875/" style="width: 100%;"><br>

<input type="text" id="function" placeholder="url" style="width: 100%;"/><br>

<input type="text" id="type" placeholder="type" value="POST" style="width: 100%;"/><br>

<textarea type="text" id="params" placeholder="param" style="width: 100%;" >{}</textarea><br>

<input type="text" id="datetype" placeholder="dataType" style="width: 100%;" value="json"><br>

<input type="text" id="header" placeholder="header" value="application/json" style="width: 100%;"><br>

<input type="text" id="xdebug" placeholder="xdebug" style="width: 100%;"><br>

<button id="request1">cors request</button>

<button id="request2">jsonp request</button><br>

<span id="res"></span>

<script src="js/jquery.min.js"></script>

<script src="https://cdn.bootcdn.net/ajax/libs/json2/20160511/json2.js"></script>

<script>

var TECSESSID = getCookie("TECSESSID");

// TECSESSID = "pnegvqn0mqgdgj2qfe5d9ohemo";

var conf = getCookie("request-test");

if(conf){

conf = JSON.parse(conf);

$("#prefix").val(conf.prefix);

$("#function").val(conf.func);

$("#xdebug").val(conf.xdebug);

$("#type").val(conf.type);

$("#params").val(conf.params);

$("#datetype").val(conf.datatype);

$("#header").val(conf.header);

}

//跨域的时候，如果有application/json，被认为是复杂请求，https://blog.csdn.net/yasha009/article/details/78784620

$("#request1").click(function(){

$("#res").html("");

setCookie("request-test", JSON.stringify({

prefix: $("#prefix").val(),

func: $("#function").val(),

xdebug: $("#xdebug").val(),

type: $("#type").val(),

params: $("#params").val(),

datatype: $("#datetype").val(),

header: $("#header").val()

}), 7);

$.ajax({

//参数

url: $("#prefix").val()+$("#function").val()+$("#xdebug").val(),

type: $("#type").val(),

data: JSON.parse($("#params").val()),

dataType: $("#datetype").val(),

// crossDomain: true,

// xhrFields: {

// withCredentials: true

// },

headers: { //会有预请求，可以不用管

"TECSESSID": TECSESSID

},

success: function(res, textStatus, request){

// console.log(res);

console.log(JSON.stringify(res));

$("#res").html(JSON.stringify(res));

TECSESSID = res.session\_id;

setCookie("TECSESSID", res.session\_id, "Session");

},

error: function(err){

// console.log(err);

console.log(JSON.stringify(err));

$("#res").html(JSON.stringify(err));

}

});

});

$("#request2").click(function(){

$("#res").html("");

setCookie("request-test", JSON.stringify({

prefix: $("#prefix").val(),

func: $("#function").val(),

xdebug: $("#xdebug").val(),

type: $("#type").val(),

params: $("#params").val(),

datatype: $("#datetype").val(),

header: $("#header").val()

}), 7);

$.ajax({

//参数

url: $("#prefix").val()+$("#function").val()+$("#xdebug").val()+"?a=b2",

type: "GET",

dataType:'jsonp',

jsonp:'jsonpcallback',

success: function(res){

// console.log(res);

console.log(JSON.stringify(res));

$("#res").html(JSON.stringify(res));

},

error: function(err){

// console.log(err);

console.log(JSON.stringify(err));

$("#res").html(JSON.stringify(err));

}

});

});

function setCookie(cname, cvalue, exdays) {

if(exdays == "Session"){

var expires = "expires=Session"

}else{

var d = new Date();

d.setTime(d.getTime() + (exdays \* 24 \* 60 \* 60 \* 1000));

var expires = "expires="+d.toUTCString();

}

document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";

}

function getCookie(cname) {

var name = cname + "=";

var ca = document.cookie.split(';');

for(var i = 0; i < ca.length; i++) {

var c = ca[i];

while (c.charAt(0) == ' ') {

c = c.substring(1);

}

if (c.indexOf(name) == 0) {

return c.substring(name.length, c.length);

}

}

return "";

}

</script>

</body>

</html>