前端跨域通信の <iframe>+

jonycai

大纲

- 为什么存在跨域
- 现有解决方案概览
- <iframe>+ 方案具体实现及一些坑
- DEMO
- 对比总结

同源策略

URL	说明	是否允许访问
http://www.a.com/a.html http://www.a.com/b.html	同域名	✅
http://www.a.com/lab/a.html http://www.a.com/script/b.html	同域名，不同目录	✅
http://www.a.com:8000/a.html http://www.a.com/b.html	同域名，不同端口	🈲
http://www.a.com/a.html https://www.a.com/b.html	同域名，不同协议	🈲
http://www.a.com/a.html http://10.198.14.14/b.html	域名和ip对应	🈲
http://www.a.com/a.html http://script.a.com/b.html	主域相同，不同子域	🈲
http://www.a.com/a.html http://a.com/b.html	同域名，不同二级域名	🈲
http://www.a.com/a.html http://www.b.com/b.html	不同域名	🈲

see: https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

市面解决方案の关键词

JSONP

CORS

后台配合

location.hash

window.name

document.domain

postMessage

<iframe>+

<iframe>+ location.hash

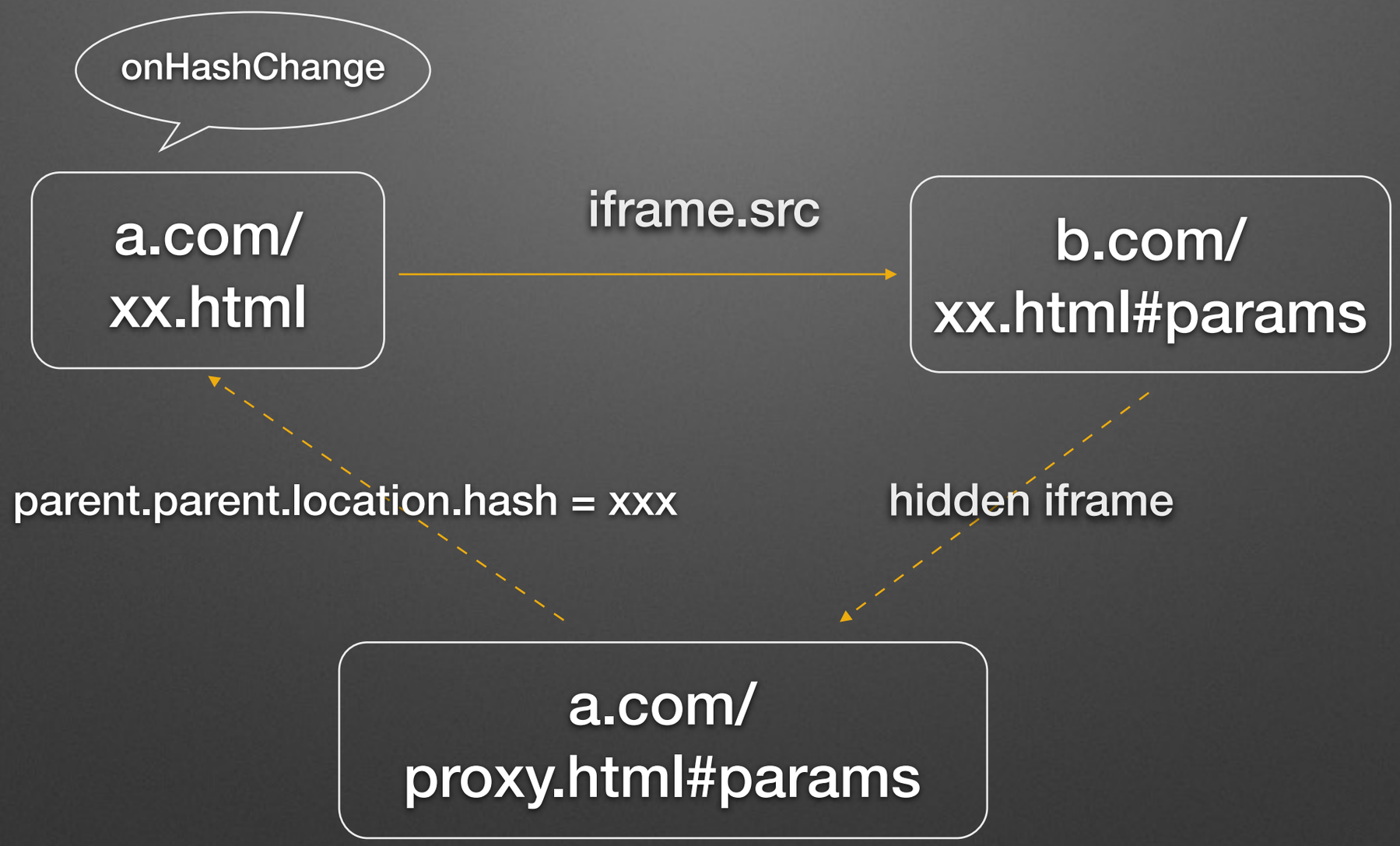
技能解锁

🔓 parent.parent.xxx

说明：

多层嵌套的iframe中，任何两个页面只要满足通信条件就能通过嵌套关系链相互访问

<iframe>+location.hash



location.hash

- 为什么一定要使用hash来传递?
- parent.parent.location.hash 是否多余?

parent.parent.window._Callback 😎

此处应有demo

<iframe> + window.name

技能解锁

🔒 window.name

说明：

window.name 值在不同的页面（甚至不同域名）加载后依旧存在，并且可以支持非常长的 name 值（2MB 😱

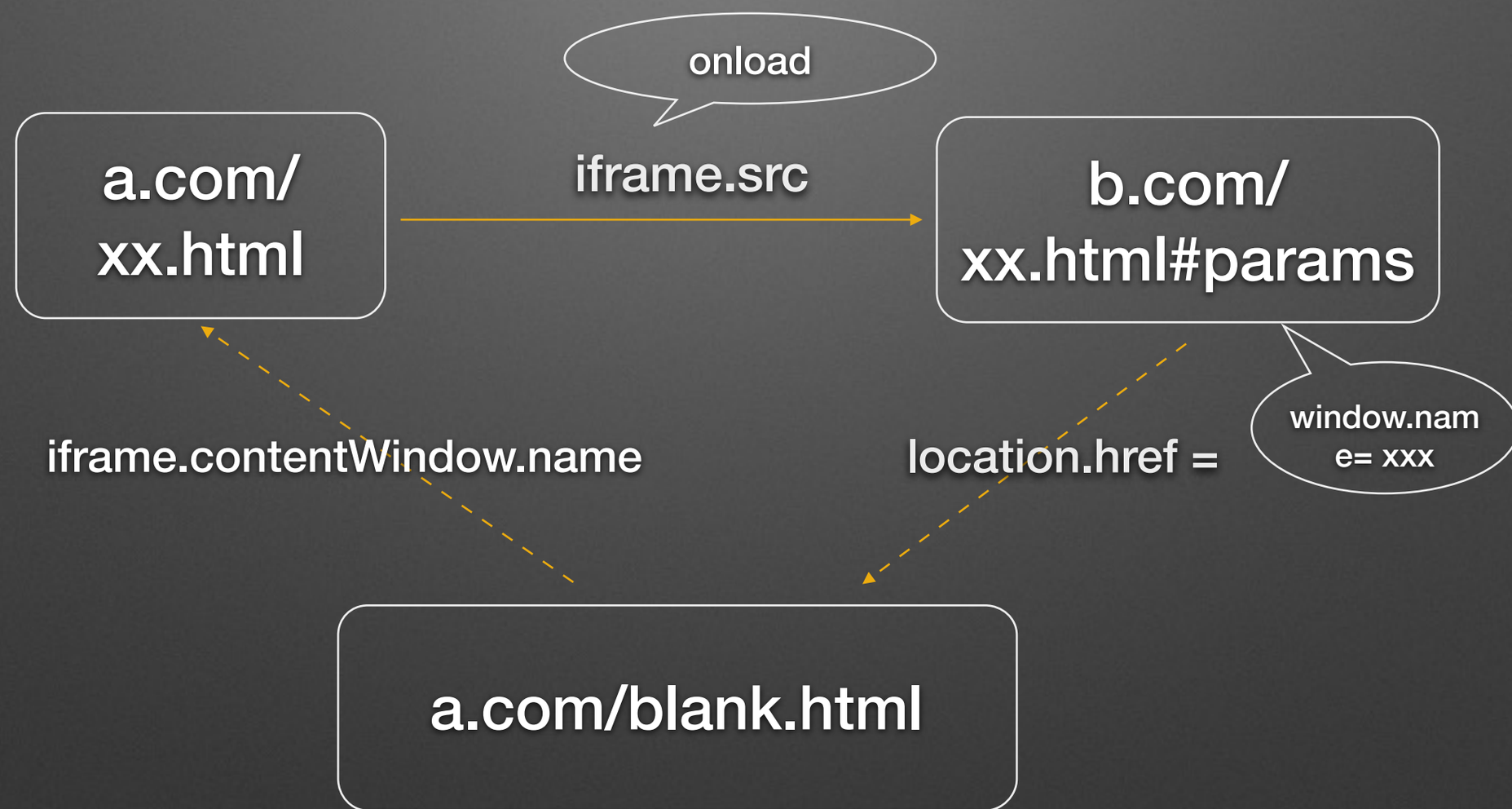
技能解锁

🔒 iframe.onload

说明：

iframe加载之后，如果页面通过location.href跳转，会再次触发onload事件😳

<iframe>+window.name



Show me the code

a.com/xx.html

```
createIframe({
   src : 'http://b.com/xx.html#params',
   onload : function () {
      var step = 0;

      return function () {
         if(step == 1){
            var ifr =
            document.getElementById('xx');
            console.log(ifr.contentWindow.name);
            step = 0;
            closeIframe('xx');
         }else{
            step = 1;
         }
      }
   }()
})
```

b.com/xx.html

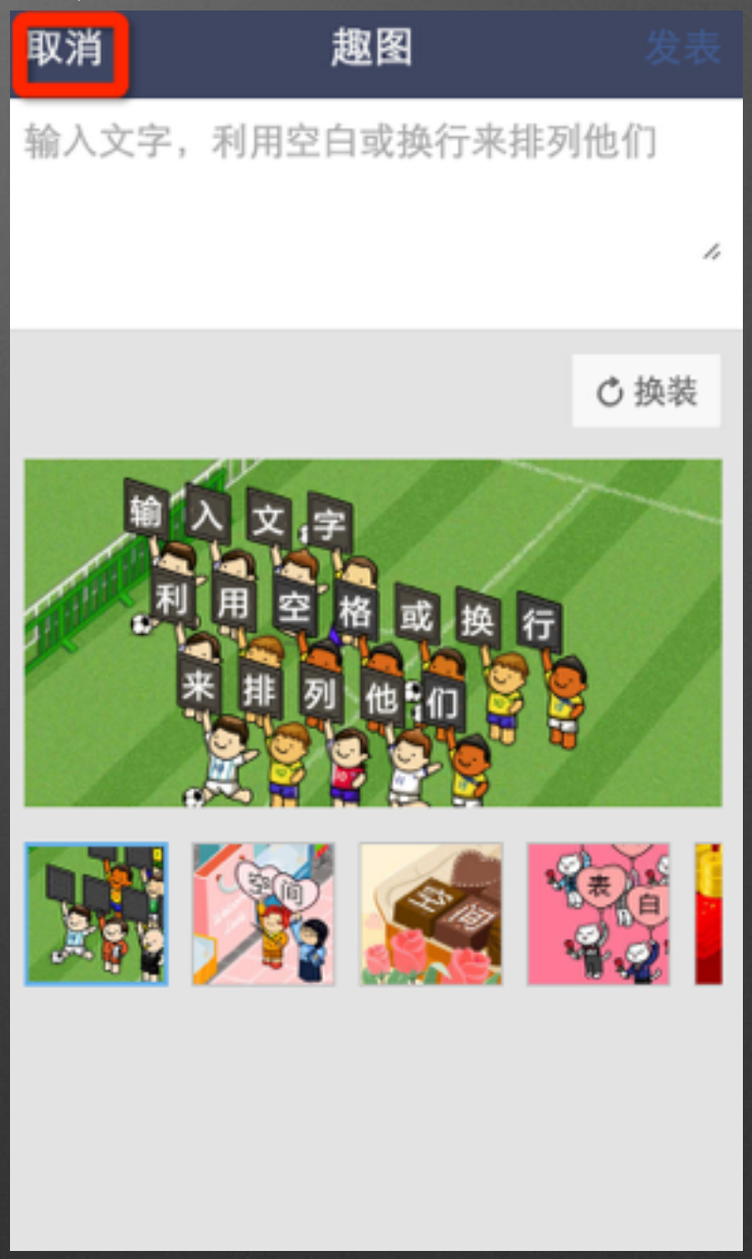
```
window.name = 'xxx';
location.href = 'http://a.com/blank.html'
```

有木有问题 ？

谜の跳转

三次跳转

返回哪个页面？



谜の跳转

pingd?dm=m.qzon...	GET	200	text...	Other	62 B	439 ms	
ho_cross_domain?t...	GET	302	text...	index.js:17	500 B	141 ms	
ho_cross_domain2...	GET	302	text...	http://pt...	771 B	897 ms	
mqzoneFunny.html...	GET	200	doc...	http://pt...	1.6 KB	245 ms	
favicon.ico	GET	200	x-ic...	mqzoneF...	1.1 KB	61 ms	
infocenter	GET	200	doc...	mqzoneF...	12.6 KB	834 ms	

```
        var urlObj = parse(location.href).params;
        var toUrl = "http://m.qzone.com/infocenter?#form/";
        var cateName = "funny";
        if(urlObj && urlObj['cate']) {
                cateName = urlObj['cate']
        }
        toUrl += cateName;
        location.href = toUrl
</script>
```

```
setTimeout(function () {
  location.href = toUrl;
},0)
```

浏览器在什么条件下会更新history纪录？

谜の跳转

METHOD	是否能返回最初页面
http code 302	?
meta refresh	?
location.href	?
setTimeout location.href	?
window.onload location.href	?

对不起，我们不考虑IE

谜の跳转

5.5.2 Page load processing model for HTML files

When an HTML document is to be loaded in a browsing context, the user agent must queue a task to create a Document object, mark it as being an HTML document, create an HTML parser, and associate it with the document. Each task that the networking task source places on the task queue while the fetching algorithm runs must then fill the parser's input stream with the fetched bytes and cause the HTML parser to perform the appropriate processing of the input stream.

Note: The input stream converts bytes into characters for use in the tokenizer. This process relies, in part, on character encoding information found in the real Content-Type metadata of the resource; the "sniffed type" is not used for this purpose.

When no more bytes are available, the user agent must queue a task for the parser to process the implied EOF character, which eventually causes a load event to be fired.

After creating the Document object, but before any script execution, certainly before the parser stops, the user agent must update the session history with the new page.

Note: Application cache selection happens in the HTML parser.

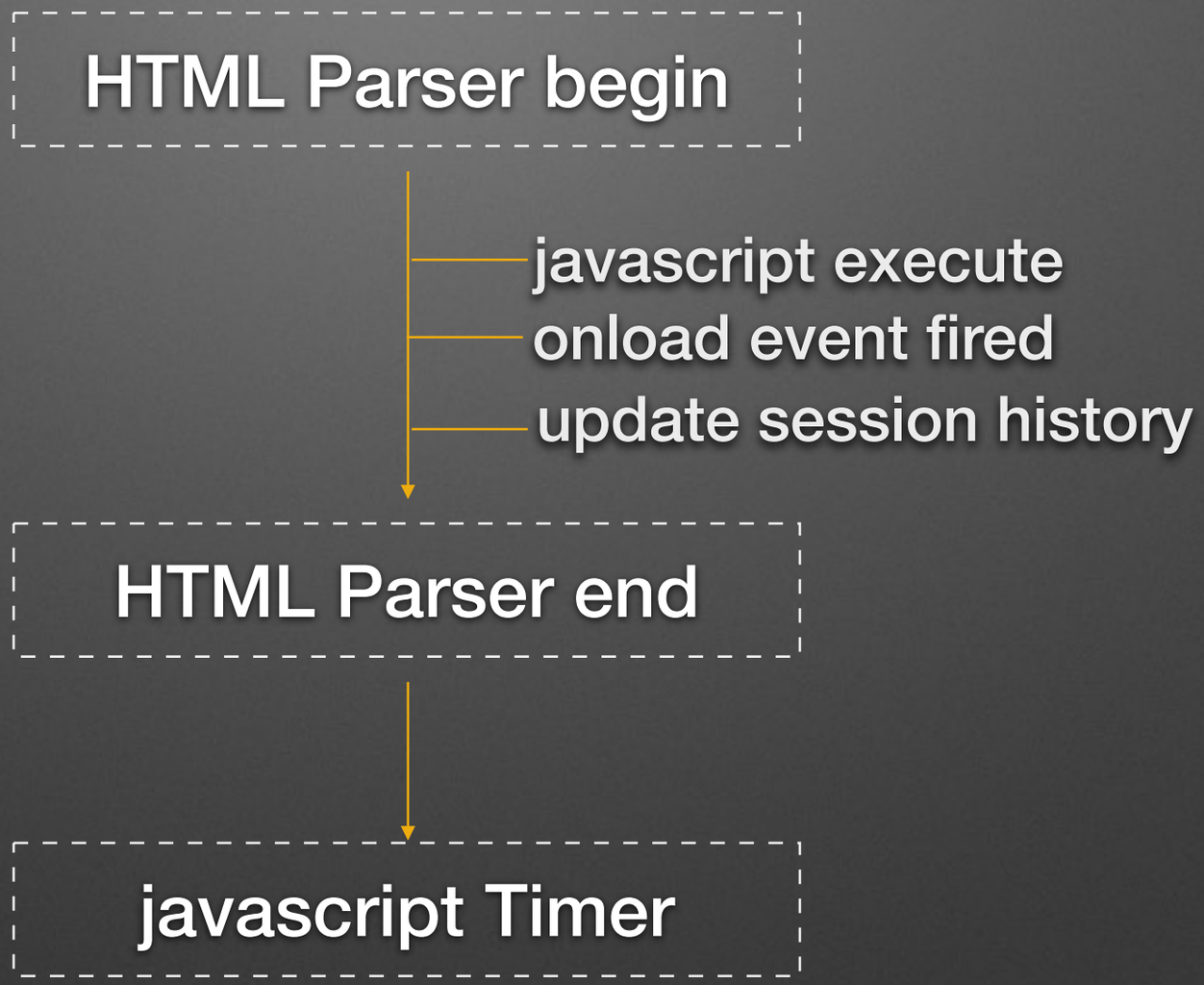
The task source for the two tasks mentioned in this section must be the networking task source.

When no more bytes are available, the user agent must queue a task for the parser to process the implied EOF character, which eventually causes a load event to be fired.

After creating the Document object, but before any script execution, certainly before the parser stops, the user agent must update the session history with the new page.

see: http://www.w3.org/TR/2011/WD-html5-20110113/history.html#update-the-session-history-with-the-new-page

谜の跳转


HTML Parser begin
javascript execute
onload event fired
update session history
HTML Parser end
javascript Timer

谜の跳转

METHOD	是否能返回最初页面
http code 302	✅
meta refresh	✅
location.href	✅
setTimeout location.href	🈲
window.onload location.href	✅

DEMO

Show me the code,again

a.com/xx.html

```
createIframe({
   src : 'http://b.com/xx.html#params',
   onload : function () {
      var step = 0;

      return function () {
         if(step == 1){
            var ifr =
            document.getElementById('xx');
            console.log(ifr.contentWindow.name);
            step = 0;
            closeIframe('xx');
         }else{
            step = 1;
         }
      }
   }()
})
```

onload只触发一次

b.com/xx.html

```
window.name = 'xxx';
location.href = 'http://a.com/blank.html'
```

↓

```
window.name = 'xxx';

//提供一种可以促发第二次onload的方法
setTimeout(function () {
   location.href = 'http://a.com/blank.html'
},0);
```

经验值 ＋1 😄

局限性

- <iframe>+location.hash or window.name两者都要求对a.com域存在代理页or空白页

<iframe>+ window.postMessage

postMessage

iframe,
window.open

http://
qzs.qq.com or
“*”

targetWindow.postMessage(message,target)

```
window.addEventListener(‘message’,receiveMessage,false)

function receiveMessage (event) {
   if(event.origin !== ‘http://qzs.qq.com’) return;

   //event.data

   //event.source
   event.source.postMessage(‘hello’,event.origin);

}
```

同域信息干扰

支付接口

window.addEventListener(message,payHandler,false)

分享接口

window.addEventListener(message,shareHandler,false)

event.origin === ‘http://qzs.qq.com’

经验：

1.message 使用object，但是里面加特殊标记位，

如：message.symbol = “__fusionShare__”

2.message 全部使用string类型传递（更好的兼容性），加特殊前缀

（’__fusionShare__’+JSON.stringify(message))

receiveMessage的时候，根据event.data的类型和特殊标记位

判断要不要处理

Caniuse postMessage

Cross-document messaging - LS

Method of sending information from a page on one domain to a page on a different one (using postMessage)

Global	85.64% + 11.79% =	97.43%
China	74.85% + 15.27% =	90.12%

Current aligned | Usage relative | Show all

IE / Edge	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
[1] 8							4.1	
[1] 9		31					4.3	
[2] 10		42					4.4	
[2] 11	38	43	7.1		7.1		4.4.4	
Edge	39	44	8	30	8.4	8	40	42
	40	45	9	31	9			
	41	46		32				
	42	47						

Notes | Sub-features (1) | Known issues (2) | Resources (7) | Feedback

[1] Partial support refers to only working in frames/iframes (not other tabs/windows). Also, objects cannot be sent using postMessage.

[2] Partial support refers to limitations in certain conditions.

see: http://caniuse.com/#search=postMessage

方案对比

方案	优点	缺点
<iframe>+location.hash	兼容性好	需要代理页
<iframe>+window.name	兼容性好， 存储数据量大	需要空白页
<iframe>+postMessage	不需要代理页， 大势所趋	兼容性

哪个跑得更快？

总结

纯前端跨域通信看使用场景，如果对主域名有控制权，建议使用location.hash + 代理页的方式，如果是给第三方提供jssdk，建议使用postMessage，兼容性可以从产品策略角度屏蔽。

- 🔒 parent.parent.xxx
- 🔒 window.name
- 🔒 iframe.onload
- 🔒 谜の跳转

Thank You !