1.XMLHttpRequest

XMLHttpRequest 允许一步发送和接收数据。

```
var url="/data.php";
var xhr=new XMLHttpRequest();
xhr.open("get",url+?+"key=value&key1=value1",true);
xhr.onreadystatechange=function(){
    if(xhr.readyStatr==4){
        //var responseHeader=xhr.getAllResponseHeaders();
        //var data=xhr.responseText;
    }
}
xhr.send(null);
```

使用 xhr 时，get 与 post 对比：对那些不会改变服务器状态，只会获取数据（ "冥等行为" ）的请求应该使用 get，经 get 请求的数据会被缓存起来，如果需要多次请求统一数据的话，有助于提升性能。当请求数 url 加上参数的长度接近或超过 2048 个字符时，才应该用 post 获取数据。

2.Dynamic script tag insertion

```
var  scriptElement=document.createElement("script");
scriptElement.src="http://www.test.com/lib.js";
document.getElementByTagName("head")[0].appendChild(scriptElement);
```

3.iframes

4.comet

5.Multipart XHR

允许客户端只用一个 http 请求就可以从服务端向客户端传送多个资源，它通过在服务器端将资源（css 文件、HTML 片段、JavaScript 代码或 base64 编码的图片）打包为一个由双方约定的字符串分割的长字符串并发送到客户端，然后用 JavaScript 代码处理这个长字符串，根据他的 mime-type 类型和传入的其他头信息来解析出每个资源。

```
var xhr=new XMLHttpRequest();
xhr.open("get",url,true);
xhr.onreadystatechange=function(){
        if(xhr.readyState==4){
                splitImage(xhr.responseText);
        }
}
xhr.send(null);

//假设服务器端返回一个 base64 编码过后的 imageString，每个图片用\u0001 分隔
function splitImage(imageString){
        var imageData=imageString.split("\u0001");
        var imageElement;
        for(var i=0,len=imageData.length;i<len;i++){
                imageElement=document.createElement("img");
                imageElement.src=""+imageData[i];
                container.appendChild(imageElement);
        }
}
function handleCss(data){
    var style=document.createElement("style");
```

```
    style.type=”text/css”;
    var node=document.createTextNode(data);
    style.appenChild(node);
    document.getElementsByTagName(“head”)[0].appendChild(style);
}
```