# Ajax

1. **Ajax（asynchronous javascript and xml）：允许浏览器器和服务器通信，无序刷新页面。（不用刷整个页面）**
2. **Ajax工作原理：**



1. **XMLHttpRequest对象：**

用于实现Ajax功能，有3个关键函数

* **onreadystatechange**：事件处理函数，由服务器触发，Ajax在执行过程，服务器会通知客户当前的通信状态：**readyState=1、2、3、4（4：表示响应完成）。**

readytState每次值得变化，都会触发readystatechange事件。

* **state取值说明：**
* **404：没找到页面（notfound）**
* **403：禁止访问（forbiden）**
* **500：内部服务器出错（internal service error）**
* **200：正常（ok）**
* **304：没被修改（not modified）**
* **open（method，url，asynch）**：允许用Ajax调用服务器发送请求
* **method：**GET（从服务器获取数据时使用）、POST（想服务器发送数据时使用）
* **url：**路径字符串，指向所亲求得服务器上得文件，绝对路径或相对路径都可以
* **asynch：**是否异步传输，默认为true
* **send（data）**：发送数据，将data字符串发送给服务器，若使用的是GET，则用send(null)
* **responseText属性**：包含从服务器发送的数据

1. **index.jsp使用 XMLHttpRequest示例：使用DOM方式**

<%@ **page** contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
 <head>  
 <title>TEST XMLHttpRequest</title>  
 <script type="text/javascript">  
 window.onload = function(){  
  **//1、获取a节点**  
 var aNodes = document.getElementsByTagName("a");  
 for(var i = 0; i < aNodes.length; i++){  
 aNodes[i].onclick = function () {  
  **//2、创建XMLHttpRequest对象**  
 var request = new XMLHttpRequest();  
  **//3、准备发送请求的数据**  
 var url = this.href;  
 var method = "GET";  
  **//4、调用XMLHttpRequset的open方法**  
 request.open(method,url);  
 **//5、send**  
 request.send(null);  
  **//6、XMLHttpRequest添加onreadystatechange响应函数**  
 request.onreadystatechange = function () {  
  **//7、判断响应函数是否完成：当readyState=4时，完成**  
 if(request.readyState == 4){  
  **//8、判断响应是否可用：status=200，304时可用**  
 if(request.status == 200 || request == 304){  
 alert(request.responseText);  
 }  
 }  
 }  
  **//9、取消节点a的默认行为：防止发生事件冒泡**  
 return false;  
 }  
 }  
 </script>  
 </head>  
 <body>  
 Test Tomcat  
 <br><br>  
  **<%-- 引用 helloAjax.txt的内容：helloAjax.txt和index.jsp在同级目录下 --%>**  
 <a href="helloAjax.txt">Test AJax: Click </a>  
 </body>  
</html>

1. **Ajax的数据格式**

* **HTML格式：当数据不需要和其他程序交互时，用HTML最简单**

使用Ajax来获取Html格式的数据 (用JQuery方式获取)

**示例：HTMl格式的数据**

<h2><a href="mailto:andy@clearleft.com">Andy Budd</a></h2>  
<a href="http://andybudd.com/">http://andybudd.com/</a>

**解析：**

<style type="text/css">  
  **@import** url("clearleft.css");  **//引用css文件**  
</style>

<script type="text/javascript" src="../scripts/jquery-1.7.2.js"></script>

<script type="text/javascript">  
 $(function() {  
 $("a").**click**(function() {  
 var url = this.href + " h2 a";

**//增加时间戳，防止因浏览器缓存，不能实时更新**  
 var args = {"time":new Date()};

**//任何一个 html 节点都可以使用 load 方法来加载 Ajax, 结果将直接插入 html 节点中** $("**#details**").**load**(url, args);  
 return false;  
 })  
 })  
 </script>  
</head>  
<body>  
<h1>People</h1>  
 <ul>  
 <li>  
 <a href="file/yj.html">yj</a>  
 </li>  
 <li>  
 <a href="file/chris.html">chris</a>  
 </li>  
 </ul>  
 <div id="**details**"></div>  
</body>

* **XML格式：当远程应用程序未知时，XML格式是首选，在web中通用**

XMl是一种通用的数据格式，可使用DOM控制文档,但解析困难

**示例：XML格式的数据**

<?xml version="1.0" encoding="utf-8"?>  
<details>  
 <name>Andy Budd</name>  
 <website>http://andybudd.com/</website>  
 <email>andy@clearleft.com</email>  
</details>

**解析：使用JQuery解析数据**

$(function() {  
 $("a").click(function() {   
 var url = this.href;  
 var args = {"time":new Date()};  
 **//url:  
 //args: JSON 格式  
 //function: 回调函数：当响应结束时，回调函数被触发。响应结果在 data 中**  
  **$.post**(url, args, function(data) {  
 var name = $(data).find("name").text();  
 var email = $(data).find("email").text();  
 var website = $(data).find("website").text();  
 **//保存数据到id=details的元素子节点下：解析成HTML数据格式**  
 $("#details").empty()  
 .append("<h2><a href='mailto:" + email + "'>" + name + "</a></h2>")  
 .append("<a href='" + website +"'>" + website + "</a>");  
 });  
   
 return false;  
 });  
})

* **JSON格式：涉及到数据重用，可使用JSON格式**

JavaScript object notation，比XML更轻巧，是javascript原生格式。在javascript中处理时，不需要使用任何的jar包。

**示例：**

{"person": {  
 "name":"Richard Rutter",  
 "website":"http://clagnut.com/",  
 "email":"richard@clearleft.com"  
 }  
}

**解析：**

**（1）使用XMLHttpRequest的eval()**

var request = new XMLHttpRequest();

var **jsonRespinse** = request.responseText;

var dataObject = (“(” + **jsonRespinse** + “)”);

**（2）使用JQuery解析**

$(function() {  
 $("a").click(function() {  
 var url = this.href;  
 var args = {"time":new Date()};  
 **//url:  
 //args: JSON 格式  
 //function: 回调函数：当响应结束时，回调函数被触发。响应结果在 data 中**  
  **$.getJSON**(url, args, function(data) {  
 var name = data.person.name;  
 var email = data.person.email;  
 var website = data.person.website;

**//保存数据到id=details的元素子节点下：解析成HTML数据格式**  
 $("#details").empty()  
 .append("<h2><a href='mailto:" + email + "'>" + name + "</a></h2>")  
 .append("<a href='" + website +"'>" + website + "</a>");  
 });  
 return false;  
 });  
})

1. **JQuery解析AJax**

* **一共三层：**
* **第一层：**
* $.ajax：
* **第二层**：
* **.load(url[,data][,callback])**：远程载入HTML代码到DOM中,callback为请求成功后，所执行的函数

**eg：点击button按钮后，将获得的数据放到元素id=div1后面**

$("button").click(function(){

$("#div1").load("demo.txt",function(responseTxt,statusTxt,xhr){

if(statusTxt=="success")

alert("外部内容加载成功!");

if(statusTxt=="error")

alert("Error: "+xhr.status+": "+xhr.statusText);

});

});

* **$.get(url[,data][,callback][,type])**：从指定的资源请求数据。用GET方式进行异步请求，type为服务器返回内容的格式：xml、html、json…

**eg：**

$("button").click(function(){

$.get("demo\_test.php",function(data,status){

alert("数据: " + data + "\n状态: " + status);

});

});

* **$.post(url[,data][,callback])**：向指定的资源提交数据（也可用于获取数据）。

**eg：**

$("button").click(function(){

$.post("/try/ajax/demo\_test\_post.php", **//这是url**

{ name:"菜鸟教程", **//这些是参数agrs**

[url:"http://www.runoob.com](url:%22http://www.runoob.com)"

},

function(data,status){  **//这个function就是callback**

alert("数据: \n" + data + "\n状态: " + status);

});

});

* **第三层：**
* $.getScript()：
* $.getJSON()：



1. **Jackson**

可用于获取解析后的JSON格式数据，不需要手写JSON格式，只需获取对应的数据，即可使用Jackson转换成JSON格式的数据。

1. **级联操作：**
2. **Ajax回调函数示例：封装Ajax代码**

<!DOCTYPE html>

<html>

<head>

<script>

**var xmlhttp;**

**function loadXMLDoc(url,cfunc)**

{

if (window.XMLHttpRequest)

{// IE7+, Firefox, Chrome, Opera, Safari 代码

xmlhttp=new XMLHttpRequest();

}

else

{// IE6, IE5 代码

xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");

}

xmlhttp.onreadystatechange=cfunc;

xmlhttp.open("GET",url,true);

xmlhttp.send();

}

**function myFunction()**

{

**loadXMLDoc**("/try/ajax/ajax\_info.txt",function()

{

if (xmlhttp.readyState==4 && xmlhttp.status==200)

{

document.getElementById("myDiv").innerHTML=xmlhttp.responseText;

}

});

}

</script>

</head>

<body

<div id="myDiv"><h2>使用 AJAX 修改文本内容</h2></div>

<button type="button" **onclick**="**myFunction()**">修改内容</button>

</body>

</html>