

# 1.AJAX

---

## 1.什么是AJAX

Asynchronous Javascript And Xml

异步的 JS 和 xml(*EX*tensible Markup Language)

通过 JS 异步的向服务器发送请求并接收响应数据

同步访问:

当客户端向服务器发送请求时, 服务器在处理的过程中, 浏览器只能等待, 效率较低

异步访问:

当客户端向服务器发送请求时, 服务器在处理的过程中, 客户端可以做其他的操作, 不需要一直等待

AJAX优点:

- 1.异步访问
- 2.局部刷新

使用场合:

- 1.搜索建议
- 2.表单验证
- 3.前后端分离

## 2.AJAX核心对象 - 异步对象 (XMLHttpRequest)

### 1.什么是XMLHttpRequest [简称为 xhr]

称为 "异步对象", 代替浏览器向服务器发送异步的请求并接收响应

[xhr 是由JS来提供的]

### 2.创建 异步对象 (xhr)

1.IE7+,Chrome,Firefox,Safari,Opera) -> 调用XMLHttpRequest 生成 xhr对象

2.IE低版本浏览器中(IE6及以下) -> 调用ActiveXObject() 生成xhr

```
<script>
    if(window.XMLHttpRequest){
        //支持 XMLHttpRequest
        var xhr = new XMLHttpRequest();
    }else{
        //不支持XMLHttpRequest,使用
        ActiveXObject 创建异步对象
        var xhr = new
        ActiveXObject("Microsoft.XMLHTTP");
    }
</script>
```

### 3.xhr 的成员

## 1.方法 - open()

作用：创建请求

语法：open(method,url,asyn)

参数：

method:请求方式，取值'get' 或 'post'

url:请求地址，字符串

asyn:是否采用异步的方式 - true:异步 / false:同步

步

ex: xhr.open('get','/server',true);

## 2.方法 - send()

作用：通知xhr向服务器端发送请求

语法：send(body)

参数：

get请求：body的值为null -> send(null)

post请求：body的值为请求数据 -> send("请求数据")

## 3.属性 - readyState

作用：请求状态，通过不同的请求状态来表示xhr与服务器的交互情况

由0-4共5个值来表示5个不同的状态

状态	说明
0	代理被创建，但尚未调用 open() 方法。
1	<code>open()</code> 方法已经被调用。
2	<code>send()</code> 方法已经被调用，响应头也已经被接收
3	下载中； <code>responseText</code> 属性已经包含部分数据。
4	下载操作已完成

#### 4.属性 - responseText

作用：响应数据

#### 5.属性 - status

作用：服务器端的响应状态码

状态码	说明
200	表示服务器正确处理所有的请求以及给出响应
404	请求资源不存在
500	服务器内部错误

#### 6.事件 - onreadystatechange

作用：每当xhr的readyState发生改变的时候都要触发的操作；

也称作回调函数；当readyState的值为4且status值为200的时候，才可以获取响应数据

## 3.AJAX的操作步骤

### 1.GET请求

```
//1.创建xhr请求
var xhr = createXhr();
//2.创建请求 - open()
xhr.open('get',url,asyn[true|false])
//3.设置回调函数 - onreadystatechange
xhr.onreadystatechange = function(){
    if(xhr.readyState == 4 && xhr.status == 200){
        //接收响应
        xhr.responseText;
    }
}
//4.发送请求
xhr.send(null);

//注意：若含有请求参数 - URL后拼接 查询字符串
QueryString
//ex: xhr.open('get','/url?key=value&key=value',asyn)
```

### 2.POST请求

```
//1.创建xhr请求
var xhr = createXhr();
```

```
//2.创建请求 - open()
xhr.open('post',url,asyn[true|false])
//3.设置回调函数 - onreadystatechange
xhr.onreadystatechange = function(){
    if(xhr.readyState == 4 && xhr.status ==
200){
        //接收响应
        xhr.responseText;
    }
}
//4设置Content-Type;
//默认ajax post的Content-Type为
"text/plain;charset=utf-8"
xhr.setRequestHeader('Content-
Type','application/x-www-form-urlencoded');
//5.发送请求
xhr.send('请求数据');
//请求数据同查询字符串 "uname=guoxiaonao&age=18"
```

## 2.JSON

---

### 1.JSON介绍

JSON:JavaScript Object Notation

在ajax中，允许将 复杂格式的响应数据 构建成JSON的格式再进行响应

### 2.JSON表现

#### 1.JSON表示单个对象

- 1.使用 {} 表示单个对象
- 2.在 {} 中使用 key:value 的形式来表示属性(数据)
- 3.Key必须要用 " " 引起来
- 4.value如果是字符串的话, 也需要用" "引起来

```
var obj = {  
    "name": "王老师",  
    "age" : 30,  
    "gender" : "Unknown"  
}
```

## 2.JSON表示一个数组

- 1.使用 [] 表示一个数组
- 2.数组中允许包含若干JSON对象 或 字符串
  - 1.使用JSON数组表示若干字符串

```
var arr = ["王伟超", "王夫人", "王小超"];
```

- 2.使用JSON数组表示若干对象

```
var arr = [  
  {  
    "name": "王老师",  
    "age": 30,  
    "gender": "男"  
  },  
  {  
    "name": "王夫人",  
    "age": 28,  
    "gender": "男"  
  }  
];
```

### 3.使用 jq 的 each() 迭代数组

回顾JS中遍历数组

```
var a = [{"name": "guoxiaonao", "age": 18}, {"name": "guoxiaonao2", "age": 22}];  
  
for (var i = 0 ; i < a.length ; i++) {  
  var obj = a[i];  
  console.log('name is ' + obj.name);  
  console.log('age is ' + obj.age);  
}
```

1.\$arr.each();

\$arr : jQuery中的数组



```
//语法：  
$arr.each(function(index,obj){  
    index:遍历出来的元素的下标  
    obj:遍历出来的元素  
});
```

2.\$each()

```
//语法：  
$.each(arr,function(index,obj){});  
arr : js 中的普通数组
```

## 4.后台处理JSON

在后台查询出数据再转换为JSON格式的字符串，再响应给前端

1.后台先获取数据

类型允许为：元组 | 列表 | 字典

2.在后台将数据转换为符合JSON格式的字符串

3.在后台将JSON格式的字符串进行响应

## 5.Python中的JSON处理

```
import json  
jsonStr = json.dumps(元组 | 列表 | 字典)  
return jsonStr
```

Django中的JSON处理

#方法1 使用Django中提供的序列化类来完成QuerySet到JSON字符串的转换

```
from django.core import serializers
json_str =
serializers.serialize('json',QuerySet)
return HttpResponse(json_str)
```

#方法2

```
d = {'a': 1}
return JsonResponse(d)
```

## 6.前端中的JSON处理

服务器端响应回来的数据是 String，需进行转换

JSON对象=JSON.parse(JSON字符串)