一、AJAX基本内容

Asynchronous JavaScript and Xml，翻译为Js和Xml的异步

定义：客户端与服务器端进行交互，而无需刷新当前页面的技术

1、同步交互与异步交互

(1)同步交互

客户端向服务器端发送请求，直到服务器端进行响应，这个过程中，用户不能做任何其他事情

(2)异步交互

同样的过程中，用户可以做任何其他事情

其他能够实现异步交互的技术：

<iframe src="">元素

Ajax实现的是B/S架构下的异步交互

(3)同步与异步的区别

同步交互

执行速度相对比较慢

响应的是完整的HTML页面

异步交互

执行速度相对比较快

响应的是部分数据

二、Ajax -- 核心对象

1、XMLHttpRequest对象

如何创建：(固定套路)

function getXhr(){

var xhr=null; //声明XMLHttpRequest对象

if(window.XMLHttpRequest){ //根据浏览器的不同情况进行创建

xhr=new XMLHttpRequest(); //表示除IE外的其他浏览器

}else{

xhr=new ActiveXObject('Microsoft.XMLHttp'); //表示IE浏览器

}

return xhr;

}

2、属性

readyState

status

3、方法

open(method,url,async)，与服务器端建立连接

method，设置当前的请求类型(GET或POST)

url，设置当前的请求地址

async，设置是否异步

Boolean类型，默认为true，表示异步

send(请求参数)，向服务器端发送请求

请求参数的格式：key=value;

\*\*如果请求类型为GET方式

send方法是不能向服务器端发送请求数据的，但是不能省略不写

要写为send(null);

请求参数要添加到open方法的url后，如url?key=value;

如果请求类型为POST方式

必须要在send()方法调用之前，使用setRequestHeader()方法

该方法用于设置请求头

setRequestHeader("Content-Type","application/x-www-form-urlencoded");

4、事件

onreadystatechange

作用：监听服务器端的通信状态改变

readyState属性，得到服务器端当前通信状态

备选项： 0，尚未初始化(服务器端)

1，正在发送请求

2，请求完成

3，请求成功，正在接收数据

4，接收数据成功

status属性，状态码

404，路径错误，找不到地址

200，成功

responseText属性，接收服务器端的数据

该属性只能接收文本(HTML)格式

5、实现Ajax的异步交互的步骤(3种方法)

(1) 创建XMLHttpRequest核心对象

与服务器建立连接

向服务器端发送请求

接收服务器端的响应数据

\*具体用法见实例练习1

(2) 创建XMLHttpRequest核心对象

注册监听

建立连接

发送请求

\*具体用法见实例练习4

(3) 创建XMLHttpRequest核心对象

建立连接

发送请求

注册监听(onreadystatechange事件)

获取服务器端的通信状态(readyState)

获取服务器端的状态码

三、XML格式

文件格式扩展名为".xml"

1、XML的版本

1.0版本，目前主流唯一版本

1.1版本，几乎没人使用

\*版本不会再有更新

2、HTML、XHTML、DHTML和XML的区别

HTML就是网页，元素定义大小写

XHTML就是严格意义的HTML，元素定义小写

DHTML -- BOM/DOM

XML -- 配置文件/数据格式

3、XML的作用

作为数据格式 - 存储数据

4、语法

(1)声明

<?xml version="1.0" encoding="UTF-8"?>

version，设置当前XML文件的版本

必须只能写在【第一行】，且前面不能有空格之类的字符

(2)定义元素

根元素

必须是起始标签

只能定义【一个】

允许自定义元素

分类

起始标签或单标签

(3)定义属性

(4)定义注释

格式：<!-- -->

\*如何定义都可以，只要符合XML语法即可，但是决定着后期使用JavaScript解析的难易程度

5、DOM解析XML

(1)IE浏览器

new ActiveXObject("Microsoft.XMLDOM");

(2)其他浏览器

创建一个DOM的解析器

var parser=new DOMParser();

使用DOM解析器解析XML文件

var xmlDoc=parser.parseFromString(xmlFile,"application/xml");

(3)创建XML解析器(固定套路)

function parseXML(xml){

var xmlDoc=null; //声明解析XML后的DOM对象

if(window.DOMParser){ //根据不同的浏览器

var parser=new DOMParser(); //其他浏览器

xmlDoc=parser.parseFromString(xml,"application/xml");

}else{

xmlDoc=new ActiveXObject("Microsoft.XMLDOM"); //IE浏览器

xmlDoc.async=false;

xmlDoc.loadXML(xml);

}

return xmlDoc;

}

(4)DOM解析XML与解析HTML没有任何区别

可以使用ById或ByName方法，但是一般用ByTagName

原因：属性值在XML格式中都是有效数据

\*浏览器不允许读取外部的XML文件

所以只能解析符合XML格式的字符串，比如

var xmlDoc=parseXML("<user><name>zhangwuji</name></user>");

6、Ajax中的XML格式

(1)请求的数据格式 - XML

①客户端如何构建符合XML格式的数据

\*请求数据的格式key=value是不能改变的，要将value值构建成符合XML格式的数据

数据类型 -- 字符串(string)，字符串的格式要符合XML的语法要求

例如 var user="<user><name>zhangwuji</name><pwd>123456</pwd></user>";

xhr.send("user="+user);

②服务器端如何接收符合XML格式的数据

数据类型 -- 字符串(string)

PHP集成了DOM的相关内容

DOMDocument类

DOMElement类

DOMNode类

(2)响应的数据格式 - XML

①服务器端如何构建符合XML格式的数据

修改服务器端的响应头的Content-Type值为"text/xml"

header('Content-Type:text/xml');

构建符合XML格式数据内容

I 手动方式构建字符串(string)内容

II DOMDocument对象的方法

loadXML(符合XML格式的字符串);

saveXML()方法进行响应

②客户端如何接收符合XML格式的数据

使用XMLHttpRequest对象的responseXML属性接收

接收回来的就是XML DOM对象(不需要使用XML解析器进行解析了)

直接使用DOM解析XML DOM对象即可

四、JSON格式

JavaScript Object Notation

JSON数据格式源于JavaScript，Js原生支持

JSON目前是网络上使用最广泛的数据格式之一

1、JSON的结构

Array，数据

Object，对象

2、支持的数据类型

String、Number、Boolean、Object、Array、null

3、Ajax中的JSON格式

(1)请求格式为JSON

客户端向服务器端发送请求为JSON格式的数据

构建符合JSON格式的字符串

保证定义字符串时，使用单引号(里面使用双引号)

服务器端接收JSON格式的数据

使用json\_decode()函数进行解析

json\_decode($json,true)

(2)响应格式为JSON

服务器端向客户端发送响应为JSON格式的数据

使用json\_encode()函数将PHP变量(array)转换成符合JSON格式的字符串

客户端接收JSON格式的数据

使用XMLHttpRequest对象的responseText属性接收

\*没有responseJSON属性

使用eval()函数进行转换

4、HTML、XML、JSON格式的优缺点

HTML

优点：简单

缺点：解析复杂

XML

优点：易于构建复杂数据

缺点：构建、解析复杂

JSON

优点：轻量级

缺点：可能转换失败

五、JQuery中的Ajax

1、方法

(1)封装第一层 -- 类似于原生Ajax的用法

①$.ajax() - 最复杂，格式 {key:value}

选项

url，请求地址

type，请求类型，默认为GET

async，是否异步，默认为true

contentType，POST方式发送数据的前提

默认值为application/x-www-form-urlencoded

data，请求数据，格式必须为key:value

success，请求成功后的回调函数

function(data,textStatus){}

data，服务器端响应的数据内容

textStatus，表示Ajax请求的状态

success

error，请求失败后的回调函数

function(XMLHttpRequest,textStatus,errorThrown){}

XMLHttpRequest，Ajax的核心对象

textStatus，表示Ajax请求的状态

error、timeout(超时)、notmodified(未找到)

errorThrown，错误异常信息

dataType，设置响应数据格式

(2)封装第二层 -- 基于第一层再次封装

②$().load(url,data,callback) - 最简单、局限性最大

url，设置当前Ajax请求的地址

data，可选项，设置当前Ajax请求的数据

格式 key:value，就是object对象

callback，可选项，当前Ajax请求成功后的回调函数

该回调函数的形参就是服务器端响应的数据内容

\*load()方法的请求类型由是否发送请求数据来决定

没有请求数据时，请求类型是GET

发送请求数据时，请求类型是POST

\*load()方法以字符串类型(HTML格式)来接收服务器端的响应数据，无法使用XML格式或JSON格式

③$.get(url,data,callback,type)，请求类型是GET

url、data、callback与上面load方法中一致

type，可选项，设置服务器端响应的数据格式

HTML格式，默认值(不写即为默认值)

xml，XML格式

json，JSON格式

\*无论是否发送请求数据，请求类型都是GET

\*get方法可以使用HTML格式、XML格式、JSON格式来接收服务器端的响应数据

④$.post()，请求类型是POST

与get方法一致

(3)封装第三层 -- 基于第二层再次封装，特殊用法

⑤$.getScript(url,callback)，动态读取脚本(Js代码)

url，读取脚本的地址(本地或服务器)

callback，读取成功后的回调函数

⑥$.getJSON(url,data,callback)，【跨域请求】，接收JSON格式数据

2、表单的Ajax异步请求

表单的序列化

serialize()，返回JSON字符串

格式 {key:value}

必须使用表单元素的name属性

serializeArray()，返回JSON对象

格式 [obj1,obj2, ...]

JSON对象是由一个对象数组组成的

必须使用表单元素的name属性

3、JQuery.form插件

作用：实现表单的异步提交

方法

ajaxForm()

ajaxSubmit()，使用Ajax异步提交表单.

底层机制：表单提交机制

4、表单异步提交的两种方式

(1)不再使用submit按钮，而是使用button按钮

通过为button按钮绑定click事件，实现Ajax异步提交

表单的序列化

表单的异步提交

(2)依旧使用submit按钮

为<form>元素绑定onsubmit事件

在onsubmit事件的处理函数中

表单的序列化

表单的异步提交

阻止表单默认行为(return false)

5、跨域请求

$.getJSON(url,data,callback)方法，接收JSON格式数据

url，请求地址

data，请求数据

callback，请求成功后的回调函数

跨域分类

完全跨域 -- IP不同

例如http://www.baidu.com

跨子域 -- IP相同，但端口号不同

例如http://127.0.0.1:80

\*万维网协议 - 默认是不允许跨域请求的

如何实现跨域请求：

使用JSONP形式的回调函数来加载其他网域的JSON数据

JSONP -- JSON Padding，JSON的一种使用模式

核心内容 - HTML中的具有开放策略的<script>元素

实现方式：

$.getJSON()方法的url后增加请求数据

url?callback=?

如果callback是一个函数的话，假设$callback就是函数的名称

函数的调用体 - $callback(实参)

向该函数传递的实参的格式为JSON