# 今天课程重点知识点

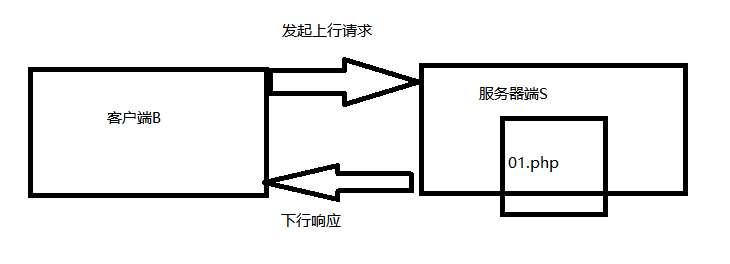
* 掌握AJAX技术
* 掌握JSONP跨域
* 了解bootstrap
* 为了学习Vue进行了解node平台（NPM）

## 1.1AJAX技术简介

问题：当用户在地址栏中输入一个网址（URL），按下回车到底发了什么？

答：当用户在地址栏中输入一个网址按下回车，客户端会向服务器端发起一次上行请求，服务器端接收到请求以后，会响应的做出相应。

经过浏览器渲染用户就可以看见了。



注意：客户端可以向服务器端发起上行请求，常见上行请求有GET、POST上行请求。（起始一共有20多种上行请求）



百度百科解释：

ajax 即“Asynchronous Javascript And XML”（异步 JavaScript 和 XML），是指一种创建交互式[网页](https://baike.baidu.com/item/%E7%BD%91%E9%A1%B5/99347" \t "https://baike.baidu.com/item/ajax/_blank)应用的网页开发技术。

Ajax = 异步 [JavaScript](https://baike.baidu.com/item/JavaScript" \t "https://baike.baidu.com/item/ajax/_blank) 和 XML 或者是 HTML（[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80/6805073" \t "https://baike.baidu.com/item/ajax/_blank)的子集）。

Ajax 是一种用于创建快速动态网页的技术。

Ajax 是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术。

通过在后台与服务器进行少量数据交换，Ajax 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

传统的网页（不使用 Ajax）如果需要更新内容，必须重载整个网页页面。

总结：AJAX【Asynchronous Javascript And XML】是一门前端技术（不是一门语言），客户端可以‘悄悄的’向服务器端发起上行请求，

服务器端可以悄悄的做出相应的下行响应，在页面没有重新加载情况下可以实现页面局部更新。

举例子：163邮箱注册业务，就是用AJAX技术；



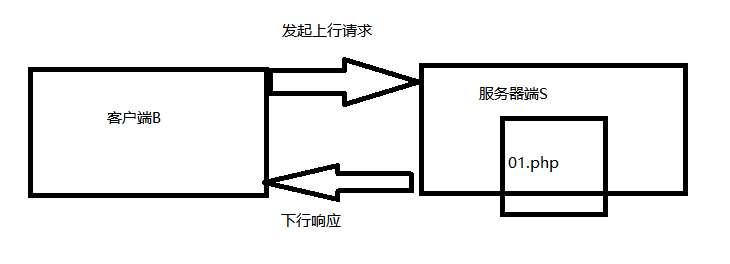
## 1.2AJAX技术基本使用

概述：AJAX用户端可以向服务器端悄悄的发起上行请求，服务器端悄悄的做出相应的下行响应。

在页面没有重新加载情况下实现页面局部更新。

注意:前端单中AJAX技术实现其实是由内置构造函数XMLHttpRequest构造函数实现的。

但是在工作当中一般都是用JQ，因为JQ将原生AJAX技术进行封装。



//JQ当中AJAX技术------GET

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

    <!-- 使用JQ务必引包 -->

    <script src="./js/jQuery.min.js"></script>

</head>

<body>

    <h1>今天课程重点就是AJAX技术</h1>

    <button>单击我向服务器端----悄悄发起上行请求（GET）</button>

</body>

</html>

<script>

    //给按钮绑定单击事件

    $("button").click(function () {

        //向服务器端悄悄发起上行请求----GET,拉去服务器数据

        //第一个参数向向服务器端请求path

        //第二个参数:客户端向服务器端额外传递一些数据(可有可无)

        //第三个参数：当服务器响应数据成功的时候会立即执行一次

        $.get("./data.txt", {

            "name": "小明",

            'ps': 123

        }, function (data) {

            //data:就是服务器端响应数据

            //修改H1标题

            $("h1").html(data);

        });

    });

</script>

//JQ当中AJAX技术------POST

    //第二个按钮发起POST请求

    $("button:eq(1)").click(function () {

        //向服务器端去悄悄的发起上行请求----POST，拉去服务器数据

        $.post("./data.txt", function (data) {

            //在页面没有重新加载情况下实现页面局部跟新

            $("h1").html(data);

        });

    });

//JQ当中AJAX技术------GET||POST

    //第三个按钮单击事件

    $("button:eq(2)").click(function () {

        //向服务器端发起POST请求

        //当前这个方法可以发起GET、POST，请求参数配置务必是一个JSON数据格式

        $.ajax({

            // 请求网址

            "url": "./data.txt",

            //请求方式既可以GET、POST

            "type": "post",

            //给服务器额外传递数据

            "data": {

                "a": 1

            },

            //回调函数：可以接受服务器数据

            "success": function (data) {

                //当服务器响应成功的时候会立即执行一次

                $("h1").html(data);

            }

        });

    });

## 1.3原生AJAX技术

概述：AJAX技术原生写法在工作当中不怎么常用，但是面试的时候（前端人）有的时候会让你手写出来。

问题：你了解前端AjAX技术，实现原理其实是使用内置构造函数XMLHttpRequest构造函数实现的。

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <h1>我是默认文本</h1>

    <button>单击我悄悄发起上行请求----GET</button>

</body>

</html>

<script>

    //获取标签

    var h1 = document.querySelector("h1");

    var btn = document.querySelector("button");

    //单击按钮发起上行请求

    btn.onclick = function () {

        //悄悄发起上行请求

        if (window.XMLHttpRequest) {

            //高级浏览器写法

            var xhr = new XMLHttpRequest();

        } else {

            //IE低版本写法

            var xhr = new ActiveXObject("msxml2.0xmlhttp");

        }

        //设置请求方式

        xhr.open("get", "./data.txt", true);

        //发送请求

        xhr.send();

        //监听就绪状态

        xhr.onreadystatechange = function () {

            if (xhr.status == 200 && xhr.readyState == 4) {

                //在页面没有重新加载情况下实现页面局部跟新

                h1.innerHTML = xhr.responseText;

            }

        }

    }

</script>

大公司面试题非常注意细节：

200 请求成功

404 请求失败

500 服务器端错误

**301是永久重定向，而302是临时重定向**

## 1.4GET和POST区别

相同地方：GET、POST都属于上行请求

不同地方：

|  |  |
| --- | --- |
| GET | POST |
| 相对而言不安全 | 相对而言安全一些 |
| 给服务器额外传递数据是由上限的 | 相对而言没有上限 |
| 便于分享 | 不便于分享 |

# JSON跨域

概述：跨域（前端当中术语）：当用户发起多次请求的时候，如果协议|域名|端口号不同情况下，

称之为跨域。

比如：看下面的小表哥，问问自己是否跨域

|  |  |  |
| --- | --- | --- |
| 第一次请求 | 第二次请求 | 是否跨域 |
| <http://www.baidu.com> | <http://www.sina.com> | 跨域（域名不同） |
| <http://www.baidu.com>:80 | <http://www.baidu.com>:8080 | 跨域（因为端号不同） |
| http://127.0.0.1/index.html | <http://127.0.0.1/01.php> | 没有跨域 |
| http://127.0.0.1 | http:127.0.0.2 | 跨域(域名不同) |

注意：apache服务器可以同时模拟多个服务器

## 2.1AJAX技术不能跨域请求数据

概述：AJAX技术不能实现跨域请求数据，因为AJAX技术遵守‘同源策略’

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

    <!-- 引包 -->

    <script src="./js/jQuery.min.js"></script>

</head>

<body>

    <h1>我是来源于127.0.0.1服务器页面</h1>

    <button>拉去127.0.0.2服务器上面数据</button>

</body>

</html>

<script>

    //AJAX技术不能跨域请求数据

    $("button").click(function () {

        //请求第二个服务器上面数据

        $.get("http://127.0.0.2/案例/data.txt", function () {

        })

    });

</script>

## 2.2JSONP可以跨域请求数据

概述：AJAX技术确实不能实现跨域，但是JSONP数据格式可以实现。

JSONP数据格式跨域so easy：实现原理是如下：

* 利用script标签src属性（本身就可以跨域请求数据）
* 利用的是将一个函数执行部分放置另外一个服务器上面

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <h1>我是来源苏127.0.0.1服务器上面页面----拉去127.0.0.2服务器数据</h1>

</body>

</html>

<script>

    //声明一个同名函数

    function fetchV85(a, b, c, d, e) {

        console.log(a, b, c, d, e);

    }

</script>

<!--将另外一个服务器函数调用部分引入 -->

<script src="http://127.0.0.2/案例/fetchV85.txt"></script>

## 2.3跨域拉去京东数据

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <select></select>

</body>

</html>

<script>

    //获取下拉框

    var select = document.querySelector("select");

    //声明一个同名函数

    function jQuery4494404(arr) {

        //遍历数组

        for (var i = 0; i < arr.length; i++) {

            //创建节点

            var op = document.createElement("option");

            op.innerHTML = arr[i].name;

            select.appendChild(op);

        }

    }

</script>

<!-- 将京东服务器调用部分引入 -->

<script src="https://fts.jd.com/area/get?fid=72&callback=jQuery4494404&\_=1578732186910"></script>

# bootstrap

<https://www.runoob.com/bootstrap/bootstrap-tutorial.html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

    <link rel="stylesheet" href="./css/bootstrap.min.css">

</head>

<body>

    <div class="container">

        <div class="row">

            <div class="col-xs-6">么么哒</div>

            <div class="col-xs-6">哈哈哈</div>

        </div>

        <div class="row">

            <div class="col-xs-2">小明</div>

            <div class="col-xs-10">小123131</div>

        </div>

        <span class="glyphicon glyphicon-fullscreen"></span>

        <ul class="pagination">

            <li><a href="#">&laquo;</a></li>

            <li><a href="#">1</a></li>

            <li><a href="#">2</a></li>

            <li><a href="#">3</a></li>

            <li><a href="#">4</a></li>

            <li><a href="#">5</a></li>

            <li><a href="#">&raquo;</a></li>

        </ul>

    </div>

</body>

</html>

# node平台\*\*\*\*\*

<http://nodejs.cn/> 官方地址

概述:node诞生于2009年，它是一款软件（并不是一门语言）,这个软件有一个很牛功能，可以让JS在这个软件上运行。

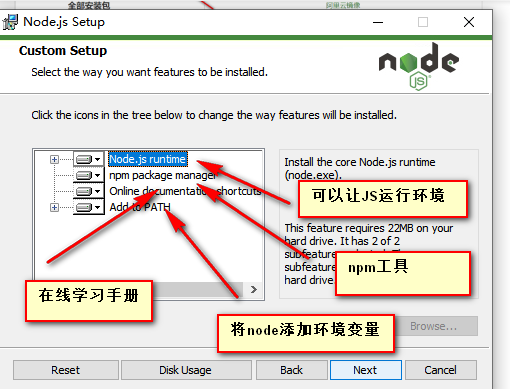
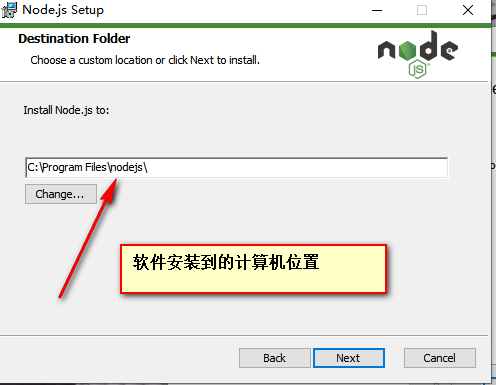
node在设计的时候目的向替代老牌Java、PHP后台语言，你会发现‘痴心妄想’；

注意：node软件最开始的时候，想搞服务器端开发（最近已经停止更新了），一定要注意，前端工程师使用node软件目的不是为了开发开发服务器。它（node）相当于前端开发当中‘脚手架’;



## 4.1安装

概述：安装非常简单，无脑下一步。



注意：安装node.js四样东西；

* 可以让JS运行环境
* NPM工具
* 在线学习手册
* 将node软件添加环境变量

验证是否安装成功：

打开CMD命令行，输入node -v 如果现实出版本号代表安装成功;

## 4.2node基本使用

概述：node是一个软件，这个软件可以让JavaScript脚本语言在这个软件上运行。

注意：node软件可以运行JS语法，node软件只能运行尾缀为.js文件（向静态页面、CSS、PHP等等运行不了）

//node平台支持JS全部核心语法

//变量

var a = 100;

var b = 200;

console.log(a + b);

//运算符[数学运算符、比较运算符、逻辑运算符]都支持

console.log(3 > 8 ? 6 : 9);

//流程控制语句

for (var i = 1; i <= 100; i++) {

  if (i % 2 == 0) console.log(i);

}

//函数

function sum(a, b) {

  return a + b;

}

console.log(sum(11, 11));

//数组

var arr = ["吃饭", "睡觉", "打豆豆"];

console.log(arr.reverse());

//构造函数

function People(name, age, sex) {

  this.name = name;

  this.age = age;

  this.sex = sex;

}

//原型对象方法

People.prototype.eat = function() {

  console.log("我可以吃八斤米饭");

};

//创建小明

var xiaoming = new People("小明", 18, "男");

xiaoming.eat();

* node软件支持JS全部核心语法（ES6、7、8、9、10）
* node平台没有DOM、BOM概念。

运行方式：node软件只能运行JS文件，运行JS文件是在CMD窗口里面运行的（没有浏览器概念）

进入到你JS文件的文件夹里面（shift+鼠标右键）使用CMD，运行方式执行那个JS文件，就node + JS文件名字即可。

## 4.3node平台中内置模块使用

概述：传统JS使用，是在静态页面中通过scrit标签引入。Node软件给JS赋予一些‘超能力’，

传统JS是没有办法做到的：比如通过JS向某一个文件里面写入一些数据，而这些所谓超能力，即为node软件给我们提供内置模块。

### 4.3.1fs模块

概述：fs是node软件给我们提供内置模块功能，可以向某一个文件里面写入数据。

//node软件给我们提供模块（给JS赋予超能力），node软件本省就有的

//fs：可以通过JS向某一个文件写入数据

var fs = require("fs");

//写入数据

fs.writeFile("./jch.txt", "老师是祖国的老花骨朵把", function () {

    console.log("数据写入......")

});

//读取数据

fs.readFile("./jch.txt", function (err, data) {

    console.log(data.toString());

})

### 4.3.2queryString模块

//node软件给我们提供模块（给JS赋予超能力），node软件本省就有的

//fs：可以通过JS向某一个文件写入数据

var fs = require("fs");

//queryString模块：可以将JSON数据格式转换为queryString字符串

var querystring = require("querystring");

//写入数据

fs.writeFile("./jch.txt", "老师是祖国的老花骨朵把", function () {

    console.log("数据写入......")

});

//读取数据

fs.readFile("./jch.txt", function (err, data) {

    console.log(data.toString());

});

//将JSON转换为queryString字符串

console.log(querystring.stringify({

    "a": 1,

    "b": 2

}));

//不是重点

var http = require("http");

//创建服务器对象

var app = http.createServer(function (req, res) {

    res.setHeader('Content-Type', 'text/html;charset=utf-8');

    //服务器响应数据

    res.end("老师自己开发的服务器");

});

//端口号设置

app.listen(3000);

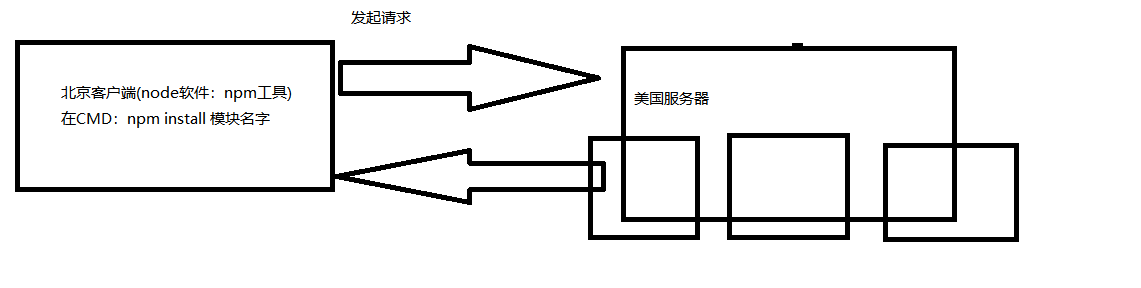
## 4.4NPM工具使用

<https://www.npmjs.com/>社区网站（这个网站服务器位于美国）

可以利用npm工具去社区网站下载别人已经写好插件使用（Vue、React）

概述：在安装node的时候，我们安装了一个工具npm工具，这个工具可以去社区下载其他人已经写好自定义模块，直接在你的程序中使用。

### 4.4.1colors社区自定义模块



自定义：程序员自己写好了的一些模块，不是node平台提供（需要我们去社区网站通过npm工具进行下载）

问题：CMD控制台打印的文字变颜色？

第一步：去社区网站npm搜索

第二步：打开CMD命令行（路径：务必是当前案例文件夹路径）

第三步：安装通过npm install 自定义模块名字

//引入社区自定义模块colors

//可以改变文字颜色

var colors = require("colors");

console.log("我非常喜欢贾成豪,因为贾成豪只是深渊".green);

console.log("我很喜欢小明".rainbow);

### 4.4.2solarlunar社区自定义模块

概述：进行公历、农历转换。

//引入社区自定义模块solarLunar

var solarLunar = require("solarLunar");

var solar2lunarData = solarLunar.solar2lunar(1996, 3, 8);

console.log(solar2lunarData);

console.log(solar2lunarData.animal);