从FTP上下载最新版Node.js安装程序

64位Windows下载：node-v6.10.2-x64.msi

32位Windows下载：node-v6.10.2-x86.msi

无需提前安装，上课一起安装....

1.阿里面试题：用户在浏览器中输入www.taobao.com直到看到页面之间发生了什么？

(1)操作系统访问网络上的DNS服务器，把域名转换为IP地址

(2)浏览器发起HTTP请求消息

(3)Web服务器接收并解析请求消息，查找指定的资源，可能访问数据库，构建并返回HTTP响应消息

(4)浏览器接收并解析响应消息

(5)浏览器缓存接收到响应内容，并解析和渲染响应内容

静态网页 和 动态网页？

静态：网页内容任何人在任何时间访问都是不变的

HTML/CSS/JS/Flash/视频音频....

动态：网页内容不同人在不同时间访问可能是不同的

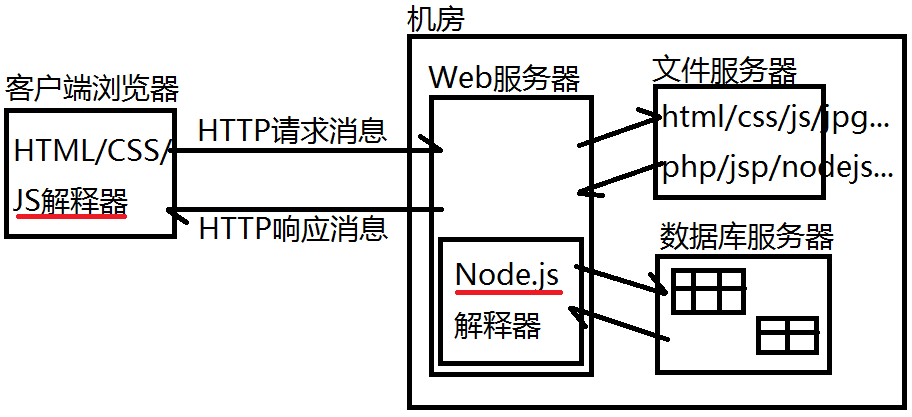
DB/JSP/PHP/ASP.NET/Node.js

JSP=HTML+Java：功能强大可靠，适合大型企业级项目

PHP=HTML+PHP：简单易用，适合互联网项目

ASP.NET=HTML+C#：易用，适合Windows平台

Node.js=HTML+"JS"：性能好，适合于服务器端IO密集型项目，不适合CPU密集型项目



2.Node.js概述

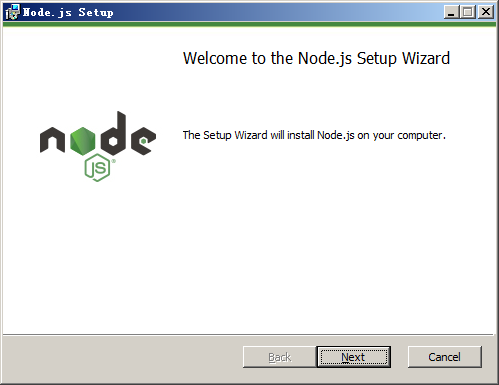
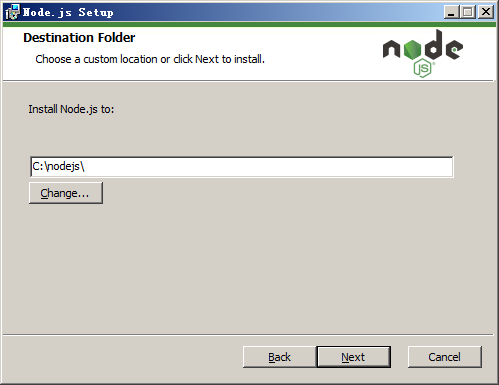
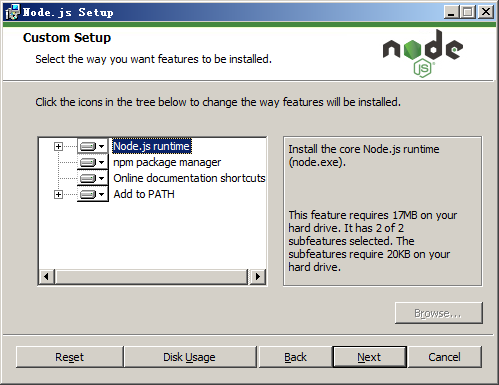
Node.js不是JS，是一种服务器端技术，它的竞争对手是PHP/JSP/ASP.NET！ 历史上第一次有一种语言可以通吃前后台！

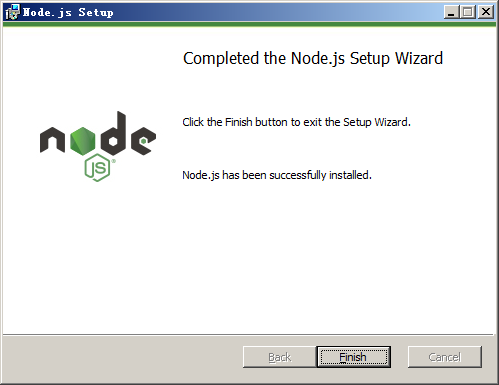
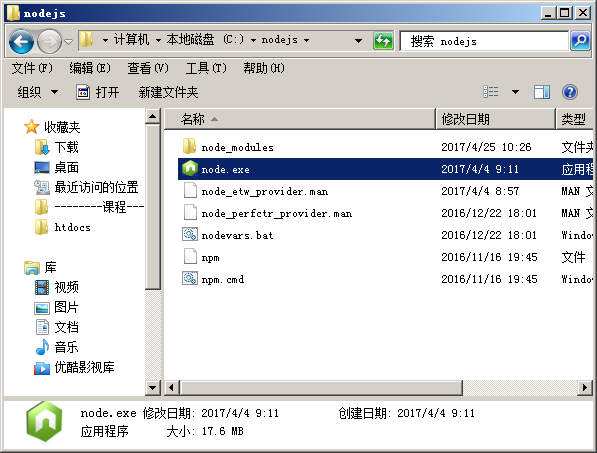
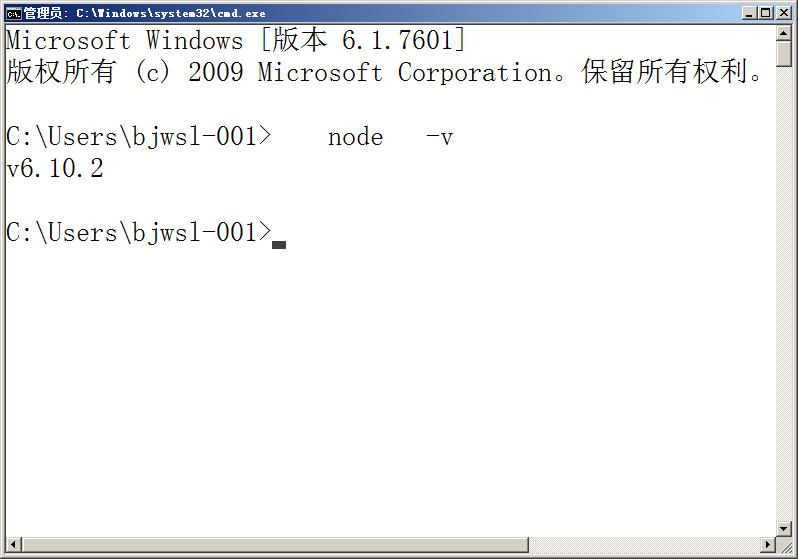
官网：www.nodejs.org

版本：0.12(16年初) 4.x(16年中) 6.x(16年底)

LTS：Long Term Support，长期稳定支持版本

Current：最新版本

3.Node.js的两种运行模式

**(1)交互模式——一般用于临时测试**

REPL：Read Evaluate Print Loop，输入一行代码执行一行

注意：交互模式自带输出功能，不必写console.log

**node 回车**

**(2)脚本模式——正式项目中使用的方式**

把要执行的所有语句编写的一个文本文件中(后缀名任意，没有都行)，一次性提交给node解释器执行

**node 完整路径名/x.js 回车**

提示：只要安装完Node.js，重启一下WebStorm，WS可以自动发现node.exe解释器程序。记得新建的项目一定要修改默认的文件编码方式为UTF-8

练习：创建一个4.js文件，打印出九九乘法表，以脚本模式在WS中运行

4.面试题：如何自学一门新语言 —— Node.js

(1)了解背景

——百度百科Node.js

(2)搭建开发环境，编写HelloWorld

下载并安装node-v6.10.2-x64.msi

交互模式、脚本模式

(3)数据类型——重点面试题

|  |
| --- |
| **前端JS中的数据类型：**  1.基本/原生/值类型  string、number、boolean、null、undefined  2.引用/对象类型  ES对象类型：String、Number、Boolean、Math、Date、RegExp、Object、Function、Error....  BOM对象类型：window、document、screen、history、location、navigator、event...  DOM对象类型：Node、Element、Attr....  用户自定义对象类型： { } |
| **后端Node.js中的数据类型：**  1.基本/原生/值类型  string、number、boolean、null、undefined  2.引用/对象类型  ES对象类型：String、Number、Boolean、Math、Date、RegExp、Object、Function、Error....  Node.js特供对象：目前有几百个....  用户自定义对象类型： { } |
|  |

(4)变量和常量

var age = 20;

const PI = 3.14;

(5)运算符

算术运算符

比较运算符

逻辑运算符

位运算符

三目运算符

赋值运算符

特殊运算符 . instanceof typeof

(6)逻辑结构

循环结构： while do..while for( ; ; ) for(.. in ..) for(.. of ..)

练习：创建6.js，声明一个保存5个学生成绩的数组，使用三种for循环依次打印出每个成绩值

选择结构：if..else.. switch..case..

练习：创建一个变量 var path='/index'; 使用二种选择结构，判断path的值为哪种（/index、/search、/login），调用不同的执行函数

(7)通用小程序

九九乘法表、100以内的质数、数组排序、...

练习：打印出100以内所有的质数

(8)函数和对象

(9)常用的组件、第三方工具、框架

(10)实际小项目

|  |
| --- |
| Modal：模态框  Model：模型  Module：模块 |

6.Node.js中的特有概念——模块

一个Web项目功能可以分为很多不同的“模块”，如商品管理模块、用户管理模块、支付模块、促销模块、商家管理模块....

Node.js按照功能的不同，可以把函数、对象分处到不同的文件、目录下，这些文件/目录在Node.js中就称为“Module”

Node.js中每个模块都是一个独立构造函数，解释器会为每个.js文件添加如下代码：

(function(exports, require, module, \_\_filename, \_\_dirname){

//exports：{ } 用于声明向外部导出的自己的成员

//require：fn 用于导入其它的模块，创建指定模块对象

//module：

//\_\_filename：

//\_\_dirname：

//自己编写的文件内容

})

|  |
| --- |
| Node.js的模块中exports和module.exports对象的区别是什么？  二者都可以用于向外界导出自己内部的成员。但：  module变量指代当前模块对象，真正导出的是modue.exports；  Node.js底层有代码： exports = module.exports。  所以：若只是给exports对象添加新的成员，则等价于给module.exports添加新成员； 但是若修改了exports的指向，则不会产生实质的作用。 |

每个模块都可以使用自己的require()函数引入另一个模块——底层本质就是创建了指定模块的一个对象实例。

require('./模块文件名')

每个模块可以使用exports对象向外导出/公开一些自己内部的成员供其它模块使用。

exports.成员名 = 成员值;

练习：创建一个文件模块：11\_Circle.js，其中定义个常量PI，声明一个方法getSize(r)，根据传入的圆形半径，返回其面积；以及一个方法getPerimiter(r)，根据传入的圆形半径，返回其周长。 再创建一个应用程序的主模块12\_app.js，引入11\_Circle模块，调用其公开的两个成员方法。

练习：编写一个模块13\_ArrayUtil，包含一个方法sum(arr)，返回指定数组中所有数值的和；一个方法avg(arr)返回指定数组中所有数值的平均值，使用module.exports导出上述两个成员；

再编写14\_app模块，引入13\_ArrayUtil模块，调用其公开的方法。

7.Node.js中模块的分类

(1)Node.js官方提供的模块——安装在解释器内部

require('模块名')

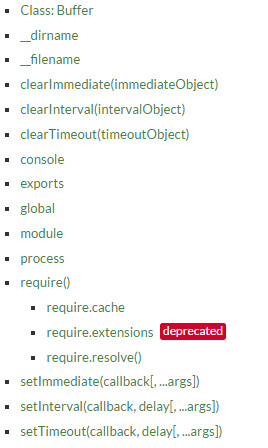
(2)第三方编写的模块

(3)用户自定义的模块

exports.x=y + require('./模块文件名')

8.Node.js预定义模块 —— Global

该模块提供的可以直接使用，而无需require('global')



exports：用于向外部导出当前模块内部的成员

module：用于指代当前模块

require：用于引入其他模块

\_\_filename：返回当前模块的文件全名

\_\_dirname：返回当前模块文件所在的目录全名

console：指代控制台对象，注意该对象与Chrome中console不同！

console.log/dir/time/timeEnd/assert....

setInterval(time, fn)

setTimeout(time, fn)

setImmediate(fn) 等价于setTimeout(0, fn)

今日练习：

仿写Node.js提供的一个模块：URL

创建模块文件MyURL，向外导出一个方法：resolve(url)，该方法接收的参数形如：

http://www.jd.com:80/ad/index?uname=tom&pno=3

返回对象，形如：

{

protocol: 'http',

server: 'www.jd.com',

port: 80,

path: '/ad/index',

query: {uname: 'tom', pno:'3'}

}

再编写一个主模块，引入上述模块并调用