==============

Node.js

Node.js不是一门语言，也不是库、框架，而是一个javaScript运行时环境

就是Node.js可以解析和执行javaScript代码，且可以脱离浏览器运行

浏览器中的javaScript

\* EcmaScript

- 基本语法

- if

- function

- Array

- Object

......

\* BOM

\* DOM

Node.js中的javaScript

\* 没有DOM、BOM

\* EcmaScript

\* 在Node.js中为javascript提供了一些服务器级别的操作API

例如： - 文件读写

- 网络服务器的构建

- 网络通信

- http服务器

Node.js特性

- event-driven事件驱动

- non-blocking I/O model 非阻塞IO模型（异步）

- lightweigt and efficient 轻量和高效

读取文件

1、使用require方法加载fs核心模块

var fs = require(‘fs’);

1. 读取文件

fs.readFile(‘./data/test.txt’, function(error, data) {

if (error) {

console.log(‘读取文件失败’);

} else {

// 通过toString()把其转换为我们认识的字符

console.log(data.toString());

}

})

// 成功

data 数据

error null

失败

data null

error 错误对象

写入文件

var fs = require(‘fs’);

// 第一个参数：文件路径

// 第二个参数：文件内容

fs.writeFile(‘./data/你好.md’, ‘北冥有鱼，其名为鲲，鲲之大，不知其几千里也’, function(error) {

console.log(‘ko’);

})

成功：

文件写入成功

error null

失败：

文件写入失败

error 错误对象

node中提供了一个核心模块：**http**

http就是创建编写服务器的

**1、加载http模块**

var http = require(‘http’);

**2、使用http.createServer()方法创建一个web服务器**

var server = http.createServer(); //返回一个server实例

**3、服务器干嘛？**

\*提供服务： 对数据的服务

\*发请求

\*接受请求

\*处理请求

\*给个反馈（发送响应）

\*注册request请求事件

\*当客户端请求过来，就会自动触发服务器的request请求事件，然后执行第二个参数：回调处理

server.on(‘request’, function(request, response) {

//客户端输入 http://127.0.0.1:3000/

http://127.0.0.1:3000/a/b

//request.url的值为 /

/a/b

console.log(‘收到客户端的请求了，请求路径是：’ + request.url);

response.write(‘你好’);

//告诉客户端，我的话说完了，你可以呈递给用户了

response.end();

//根据不同的请求路径发送不同的响应结果

var url = request.url;

if (url === ‘/’) {

response.end(‘index page’); //会在客户端页面输出index page

}

else if (url === ‘/login’) {

response.end(‘login page’);

}

else {

response.end(‘404 Not Found’)

//响应内容只能是二进制数据或字符串

数字、对象、数组、布尔值都不行

response.end(123) //数字等会报错

所有有些数据需要转换

response.end(JSON.stringify(products))

}

if (url === ‘/products’) {

var products = [

{

name: ‘苹果’,

price: 8888

},

{

name: ‘小米’,

price: 5000

},

{

name: ‘魅族’,

price: 1999

}

]

response.end(JSON.stringify(products));

}

// 在服务器默认发送的数据是utf-8编码的内容，但浏览器会按照当前操作系统的默认编码去解析服务器响应的内容，所以需要告诉浏览器是什么编码的

// 在http协议中，Content-Type就是用来告知对方我给你发送的数据内容是什么类型

if (url === ‘/plain’) {

// text/plain 就是普通文本

res.setHeader(‘Content-Type’, ‘text/plain; charset=utf-8’);

res.end(‘hello 世界’);

} else if (url === ‘/html’) {

//如果你发送的是html格式的字符串，则也需要告诉浏览器发送的是text/html 格式的内容

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

res.end(‘<p>hello html <a href=””> 点我 </a></p>’);

}

})

//request请求事件处理函数，需要接收两个参数：

request 请求对象

请求对象可以用来获取客户端的一些请求信息，例如：请求路径

response 响应对象

响应对象可以用来给客户端发送响应信息

response对象中有一个方法：write可以用来给客户端发送响应数据

write可以多次使用，但最后一次要使用end来结束响应，否则客户端会一直等待

**4、绑定端口号，启动服务器**

server.listen(3000);

// server.listen(3000, function() {

console.log(‘服务器启动成功了，可以通过 http://127.0.0.1:3000/ 来进行访问’);

});

**核心模块**

node为javaScript提供了很多服务器级别的api，这些api绝大多数都被包装到了一个具名的核心模块中了。例如文件操作的fs核心模块、http服务构建的http模块、path路径操作模块、os操作系统信息模块......

var os = require(‘os’);

console.log(os.cpus()); // 获取当前机器的cpu信息

var path = require(‘path’);

//获取一个路径中的扩展名部分

console.log(path.extname(‘c:/a/b/c/hello.txt));

require 是一个方法，用来加载模块（具名模块(fs、http)、自己编写的文件模块）

require(‘./b.js’);

在弄node中，没有全局作用域，只有模块作用域

外部访问不到内部，内部也访问不到内部

原本镜像

npm config set registry https://registry.npmjs.org

淘宝镜像

npm install -g cnpm --registry=https://registry.npm.taobao.org

卸载： npm uninstall cnpm -g