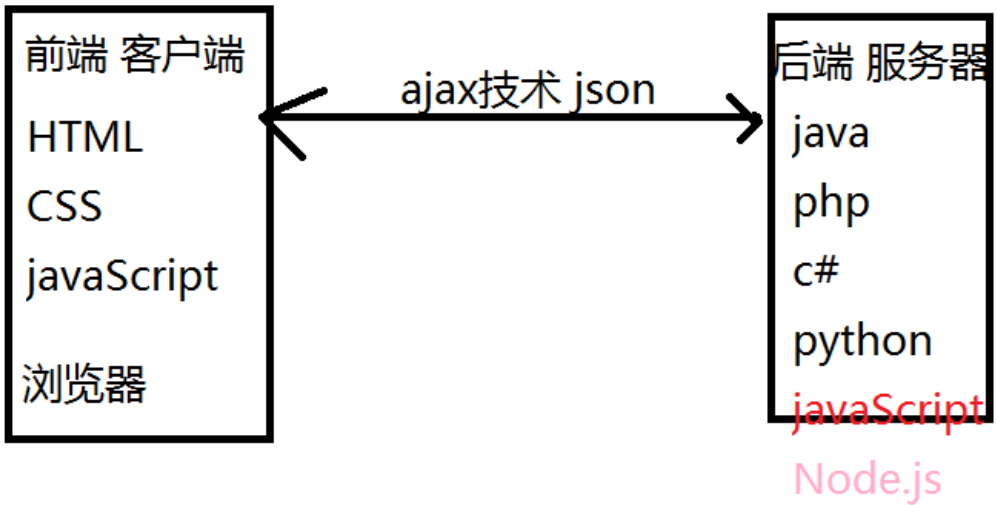
- **认识Node.js**



Web前端开发：语言HTML CSS javaScript

开发环境：浏览器

服务器开发：java、php、python、c#、javaScript

开发环境：例如php语言的运行环境是apache

js语言的运行环境是Node.js

Node.js，是一个Javascript运行环境(runtime)。实际上它是对Google V8引擎进行了封装。V8引 擎执行Javascript的速度非常快，性能非常好。同时，Node.js中还对常用的方法进行了封装。所以，Node.js既是开发环境，也是js框架。

- **搭建Node.js的开发环境**

第一步：搭建Node.js开发环境

从官网上下载符合自己机型的安装包

采用默认安装安装软件

第二步：测试是否安装成功

windows+r 打开窗口输入如下指令

node -v 查看node的版本号 如果安装成功，会输出版本号

npm -v 查看npm的版本号 如果安装成功 会输出npm的版本号

- **使用javaScript开发服务器程序**

01 测试helloWorld

如何在Node.js环境中运行js文件?

第1步：编写js文件 test.js(注意：不可以给js命名为Node.js Nodejs.js)

第2步：在Node.js的环境下运行js文件

方法1：找到js文件所在的目录，然后按下shift键，点鼠标右键，选择"在此处打开命令窗口"，打开黑屏，输出 node test.js执行js文件。

方法2： “windows+r”打开cmd窗口，输出"node "然后将test.js文件拖入黑屏窗口，回车.运行js文件

02 js中的模块系统

为了让Node.js的文件可以相互调用，Node.js提供了一个简单的模块系统。模块是Node.js 应用程序的基本组成部分，文件和模块是一一对应的。换言之，一个 Node.js 文件就是一个模块，这个文件可能是JavaScript 代码、JSON 或者编译过的C/C++ 扩展。

---

m01.js文件

```
1 var test = "01.js";
2 var name = "第一个文件";
3 // 使用exports导入提供给其他模块使用的变量与函数
4 exports.name = name;
5 exports.sayName = function(){
6         console.log(name);
7         console.log(test);
8 }
```

m02文件

```
// 导入其他模块
var m1 = require("./m01");
// 使用m1模块中的方法
m1.sayName();
console.log(m1.name);
```

03 Node.js中文件操作模块

```
var fs = require("fs");
// 异步读取文件
// fs.readFile("文件路径",读取结束之后的回调函数)
fs.readFile("03users.json",function(err,data){
        if (err) {
                console.err(err);
        };

        console.log("异步读取文件-----"+data);
});
console.log("文件读取结束");


// 同步读取文件
// fs.readFileSync(文件路径)
var data = fs.readFileSync("test.txt");
console.log("同步读取文件-------"+data);
console.log("读取文件结束");
```

04 创建http服务器

```
var http = require("http");//http模块用来创建web服务器
// 创建web服务器
// http.createServer(访问回调函数);
var server = http.createServer(function(req,res){
        // 设置响应头
        res.writeHead("200",{'content-type':'text/plain;charset=utf-8'});
        // 设置响应体
        res.write("欢迎光临本网站!");
        // 响应结束
        res.end();

});
// 给服务器设置箭头端口
// 服务器.listen(端口号,服务器开启的响应函数)
server.listen(8888,function(){
        console.log("服务器开启，端口号是8888");
})
```


```
var http = require("http");//http模块用来创建web服务器
// 创建web服务器
// http.createServer(访问回调函数);
var server = http.createServer(function(req,res){
 // 设置响应头
 res.writeHead("200",{'content-type':'text/plain;charset=utf-8'});
 // 设置响应体
 res.write("欢迎光临本网站!");
 // 响应结束
 res.end();

});
// 给服务器设置箭头端口
// 服务器.listen(端口号,服务器开启的响应函数)
server.listen(8888,function(){
 console.log("服务器开启，端口号是8888");
})
```


- **安装Express框架**

  01 什么是Express框架

  Express 是一个简洁而灵活的 node.js Web应用框架, 提供了一系列强大特性帮助你创建各种 Web 应用，和丰富的 HTTP 工具。

  使用 Express 可以快速地搭建一个完整功能的网站。

  （Express之于Node.js,相当于jQuery之于javaScript）

  02 安装Express框架

首先假定你已经安装了 Node.js，接下来为你的应用创建一个目录，然后进入此目录并将其作为当前工作目录。

```
$ mkdir myapp
$ cd myapp
```

通过 npm init 命令为你的应用创建一个 package.json 文件。欲了解 package.json 是如何起作用的，请参考 Specifics of npm's package.json handling。

```
$ npm init
```

此命令将要求你输入几个参数，例如此应用的名称和版本。你可以直接按"回车"键接受默认设置即可，下面这个除外：

```
entry point: (index.js)
```

键入 app.js 或者你所希望的名称，这是当前应用的入口文件。如果你希望采用默认的 index.js 文件名，只需按"回车"键即可。

接下来安装 Express 并将其保存到依赖列表中：

```
$ npm install express --save
```

如果只是临时安装 Express，不想将它添加到依赖列表中，只需略去 --save 参数即可：

```
$ npm install express
```

安装 Node 模块时，如果指定了 --save 参数，那么此模块将被添加到 package.json 文件中 dependencies 依赖列表中。然后通过 npm install 命令即可自动安装依赖列表中所列出的所有模块。

03 Hello，world!第一个Express应用

```
1  var express = require("express");
2  var app = express();//创建了一个web服务器应用
3  //设置端口号
4  var server = app.listen(8888,function(){
5          var host = server.address().address;//服务器的地址
6          var port = server.address().port;//服务器的端口号
7          console.log("服务器地址%s,端口号是%s",host,port);
8  })
9  // 设置get方式请求http://localhost:8888
10 app.get("/",function(req,res){
11         res.send('欢迎你好');
12 })
13 // http://localhost:8888/help
14 app.get("/help",function(req,res){
15         res.send("这是帮助页面！ ");
```

```
var express = require("express");
var app = express();//创建了一个web服务器应用
//设置端口号
var server = app.listen(8888,function(){
 var host = server.address().address;//服务器的地址
 var port = server.address().port;//服务器的端口号
 console.log("服务器地址%s,端口号是%s",host,port);
})
// 设置get方式请求http://localhost:8888
app.get("/",function(req,res){
 res.send('欢迎你好');
})
// http://localhost:8888/help
app.get("/help",function(req,res){
 res.send("这是帮助页面！ ");
})
```

04 托管静态文件

```
9  // 托管静态文件
10 //http:/localhost:8888/index.html访问web文件夹下的文件了
11 app.use(express.static("web"));
12
13 //http:/localhost:8888/static/index.html访问web文件夹下的文件了
14 app.use("/static",express.static("web"));
15
```

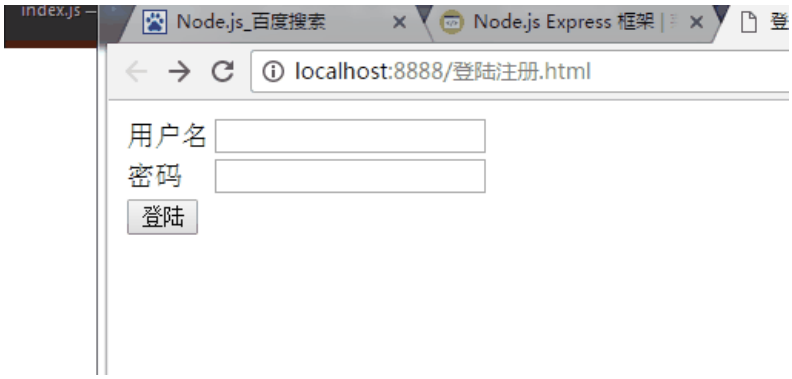
05 第二个程序 ---响应数据返回页面

```
var express = require("express");
var app = express();//创建了一个web服务器应用
//设置端口号
var server = app.listen(8888,function(){
var host = server.address().address;//服务器的地址
var port = server.address().port;//服务器的端口号
console.log("服务器地址%s,端口号是%s",host,port);
})
// 托管静态文件
//http:/localhost:8888/index.html访问web文件夹下的文件了
app.use(express.static("web"));

//http:/localhost:8888/static/index.html访问web文件夹下的文件了
app.use("/static",express.static("web"));
// 设置get方式请求http://localhost:8888
app.get("/",function(req,res){
// res.send('欢迎你好');
res.sendFile(__dirName+"/index.html")
})
// http://localhost:8888/help
app.get("/help",function(req,res){
res.send("这是帮助页面！");
})
```

06 获取get方式的请求参数

【案例】登陆页面



登陆注册.html

```
<form action="http://localhost:8888/register" method="get">
        <table>
                <tr>
                        <td><label>用户名</label></td>
                        <td><input type="text" name="username"></td>
                </tr>
                <tr>
                        <td><label>密码</label></td>
                        <td><input type="password" name="pwd"></td>
                </tr>
                <tr>
                        <td colspan="2"><input type="submit" value="登陆"></td>
                </tr>
        </table>
```

server.js

```
var express = require("express");
var app = express();
app.listen(8888,function(){
        console.log("服务器开启,端口号是8888");
})
// 托管静态文件
app.use(express.static("web"));
app.get("/register",function(req,res){
        console.log(req.query);
        if (req.query.username=="admin"&&req.query.pwd=="123") {
                res.send("登陆成功");
        }else{
                res.send("登陆失败");
        }

```

相关代码

登陆注册.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>登陆注册</title>
</head>
<body>
<form action="http://localhost:8888/register" method="get">
<table>
<tr>
<td><label>用户名</label></td>
<td><input type="text" name="username"></td>
</tr>
<tr>
<td><label>密码</label></td>
<td><input type="password" name="pwd"></td>
</tr>
<tr>
<td colspan="2"><input type="submit" value="登陆"></td>
</tr>
</table>
</form>
</body>
</html>
```

server.js

```
var express = require("express");
var app = express();
app.listen(8888,function(){
 console.log("服务器开启,端口号是8888");
})
// 托管静态文件
app.use(express.static("web"));
app.get("/register",function(req,res){
 console.log(req.query);
 if (req.query.username=="admin"&&req.query.pwd=="123") {
  res.send("登陆成功");
 }else{
  res.send("登陆失败");
 }

});
```

07 获取post方式请求参数

效果图

登陆注册.html

```html
<form action="http://localhost:8888/register2" method="post">
        <table>
                <tr>
                        <td><label>用户名</label></td>
                        <td><input type="text" name="username"></td>
                </tr>
                <tr>
                        <td><label>密码</label></td>
                        <td><input type="password" name="pwd"></td>
                </tr>
                <tr>
                        <td colspan="2"><input type="submit" value="登陆"></td>
                </tr>
        </table>
</form>
```

server.js

```
1  var express = require("express");
2  var app = express();
3  app.listen(8888,function(){
4          console.log("服务器开启,端口号是8888");
5  })
6  app.use(express.static("web"));// 托管静态文件
7  var bodyParser = require("body-parser");
8  // 创建application/x-www-form-urlencode编码解析
9  var urlcodeparser = bodyParser.urlencoded({extended:false});
10 app.post("/register2",urlcodeparser,function(req,res){
11         console.log(req.body);
12         var data = {
13                 username:req.body.username,
14                 pwd:req.body.pwd
15         };
16         res.send(JSON.stringify(data) );
17 })
```

源码

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>登陆注册</title>
</head>
<body>
<form action="http://localhost:8888/register2" method="post">
<table>
<tr>
<td><label>用户名</label></td>
<td><input type="text" name="username"></td>
</tr>
<tr>
<td><label>密码</label></td>
<td><input type="password" name="pwd"></td>
</tr>
<tr>
<td colspan="2"><input type="submit" value="登陆"></td>
</tr>
</table>
</form>
</body>
</html>
```

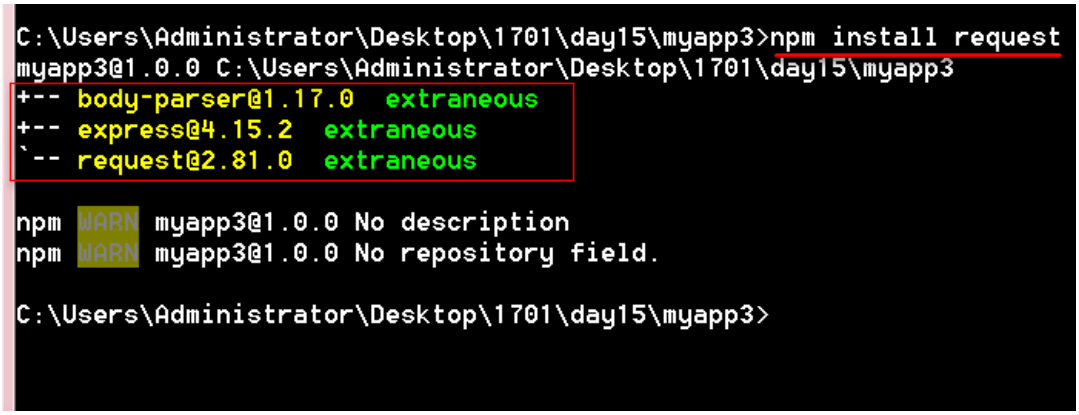
首先，先安装body-parser模块。npm install body-parser

```
var express = require("express");
var bodyParser = require("body-parser");//导入该模块
var app = express();
app.listen(8888,function(){
console.log("服务器开启,端口号是8888");
})
app.use(express.static("web"));// 托管静态文件

// 创建application/x-www-form-urlencode编码解析
var urlcodeparser = bodyParser.urlencoded({extended:false});
app.post("/register2",urlcodeparser,function(req,res){
console.log(req.body);
var data = {
username:req.body.username,
pwd:req.body.pwd
};
res.send(JSON.stringify(data) );
})
```

08 跨源请求----代理服务器

【注意】在客户端想请求其他服务器上的数据时，我们可以在自己的服务器上请求其他服务器上的数据，然后再从自己的服务器上取数据，这种跨源解决策略是代理服务器。Node.js中使用代理服务器通过request这个模块。npm install request安装request模块



（1）以get方式请求其他的服务器上的数据

```
var huodongData = "";
```

```
// 请求快牙服务器 活动模块数据
request("http://duif.applinzi.com/leyuan/activitys.php",function(err,res,body){
 if (!err&&res.statusCode==200) {
  huodongData+=body;
  console.log("活动数据----------------------------------------");
  console.log(huodongData);
 };
})
```

（2）以get方式传参请求其他服务器上的数据

```
var zixun_page1 = "";
request("http://duif.applinzi.com/leyuan/news.php?pageNum=1",function(err,res,body){
 if (!err&&res.statusCode==200) {
  zixun_page1 += body;
  console.log("1-------------------------------------");
  console.log(zixun_page1);
 };
});
```

（3）以post方式传参请求其他服务器

4. 登录页面
   请求类型：post
   请求地址：'http://duif.applinzi.com/login.php'
   请求数据：

参数名	参数说明
name	用户名
pwd	密码

```
var params = {name:"18201025923"}
request({
 url: "http://duif.applinzi.com/login.php",
 method: "post",
 body: JSON.stringify(params)  //post的数据
}, function(err, response, body){
 console.log(response.statusCode)
 console.log( "123"+ body);  //在回调里处理结果
});
```

3. 注册页面
   请求类型：post
   请求地址：'http://duif.applinzi.com/register.php'
   请求数据：

参数名	参数说明
name	用户名
pwd	密码
email	邮箱

```
var params2 = {name:"18201025923",pwd:"123456",email:"123456789@qq.com"}
request({
 url: "http://duif.applinzi.com/register.php",
 method: "post",
 body: JSON.stringify(params2)  //post的数据
```

```
}, function(err, response, body){
 console.log(response.statusCode)
 console.log( "12333333333"+ body);  //在回调里处理结果
});
```