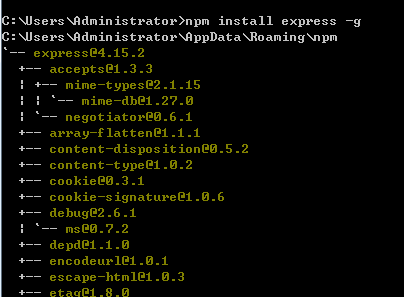
# express手册

## 安装

安装成功后位置在：C:\Users\(本机名称)\AppData\Roaming\npm\node\_modules\express

npm install express –g



查看版本(当前文档记录时间为2017.04.21)：



## 快速创建express框架

1. 安装express generator

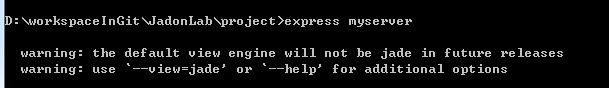
全局安装

>npm install express-generator -g

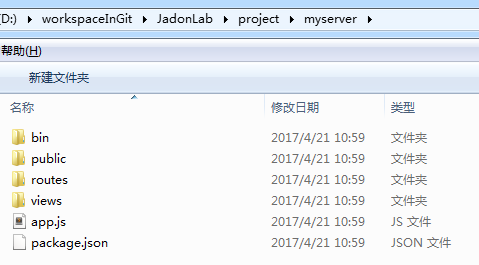
1. 创建项目

>express (项目名称)

创建了一个名为 myserver的项目



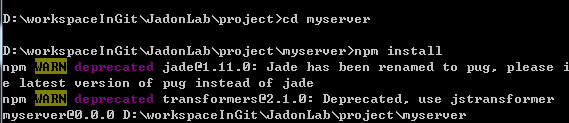
整个项目结构：



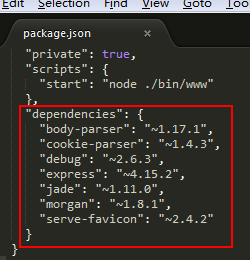
1. 安装与该项目相关的依赖包

>cd myserver

>npm install



ps: 安装依赖包是根据package.json 中 “dependencies” 属性所决定

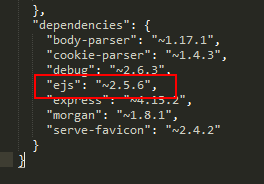


## 基于ejs模板创建express项目

声明ejs模板在项目中

>express –e (项目名称)

此时package.json

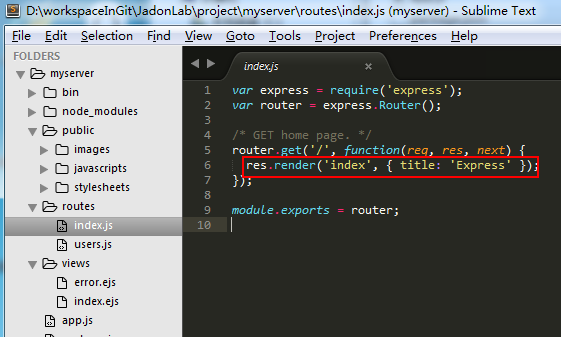


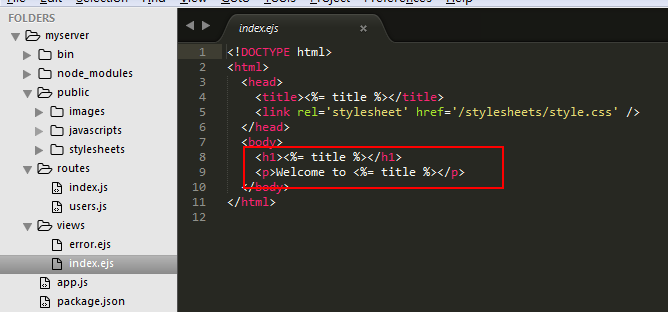
>npm install

安装package.json 的依赖

ejs模板 更好的 MVC思维。有点类似jsp的用法，数据对象可以通过 <% %> 被渲染出来

例子：





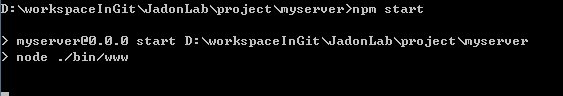
在 index.js中，response对象调用render方法，将{title:’Express’} 对象传给 index.html

在index.html 可获得title属性值 使用 <%=title%> 可渲染出来

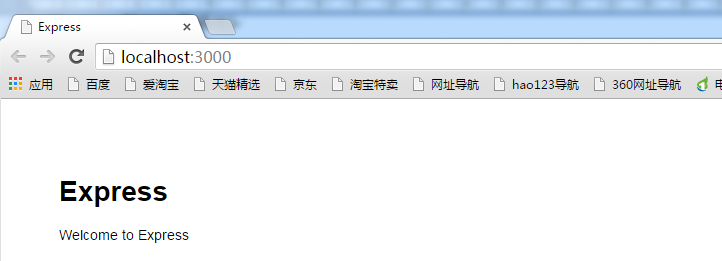
## 启动express

>cd (express项目所在路径)

>npm start

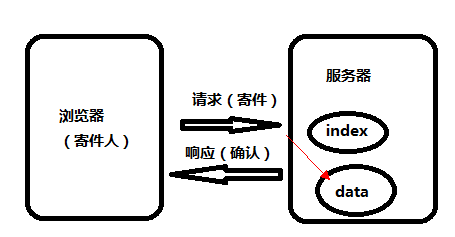


此时打开浏览器：



express 成功启动了

## express的routing(路由)

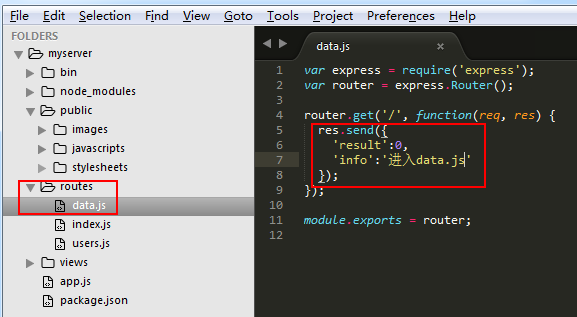


express的routing两个步骤

1. 新建处理.js

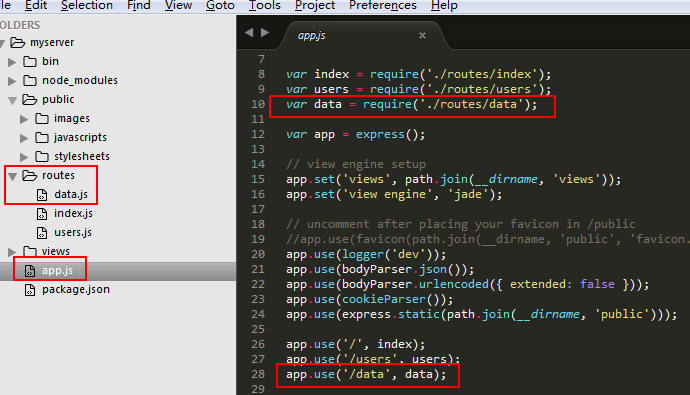
当请求 http://localhost:3000/data时，对请求数据处理

res为响应对象，send方法将json返回给客户端

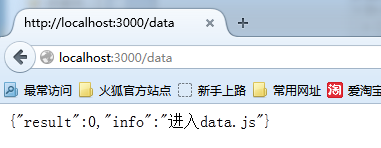


1. app.js声明

app.js作为该服务器的大门口，声明 /data



测试：http://localhost:3000/data



## Request 和 Response

请求与响应

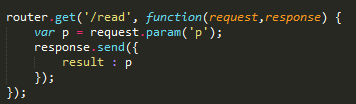
通过http请求get方式

request为请求对象

respose 为响应对象

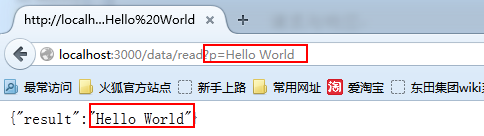
request.param() 获取请求参数

response.send() 发送返回参数



发送请求 参数p=Hello World

返回响应 输出Hello World



## express-session

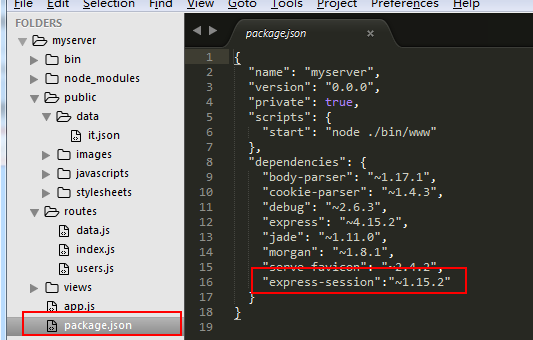
安装

全局安装

>npm install express-session –g

package.json加上（当前express-session的版本号为1.15.2）

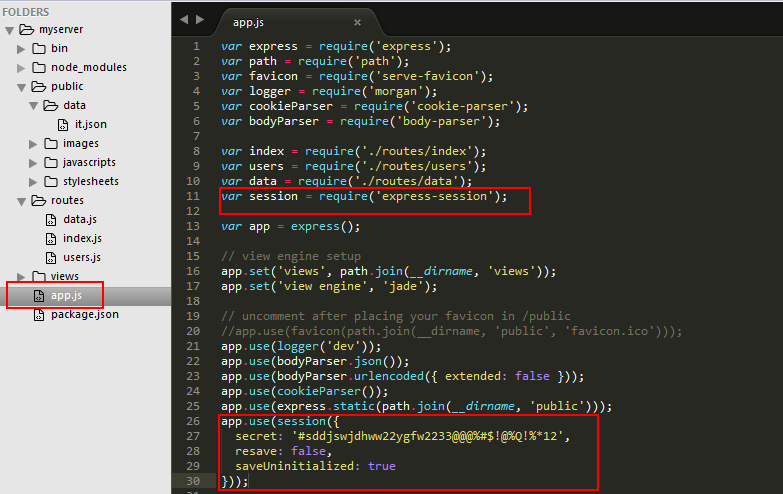
"express-session":"~1.15.2"



>cd （项目路径）

>npm install

app.js声明



**saveUninitialized**:是否设置session在存储容器中可以给修改

　　session过期30分钟，没有人操作时候session 30分钟后过期，如果有人操作，每次以当前时间为起点，使用原设定的maxAge重设session过期时间到30分钟只有这种业务场景必须同行设置resave rolling为true.同时saveUninitialized要设置为false允许修改。

**secret**:用来注册session id 到cookie中，相当与一个密钥。

**resave**:是否允许session重新设置，要保证session有操作的时候必须设置这个属性为true

## 举个栗子（request和response）

以下一个简单的登录功能栗子

前端代码：

<!doctype html>  
<**html** lang=**"en"**>  
<**head**>  
 <**meta** charset=**"UTF-8"**>  
 <**title**>**登录模块**</**title**>  
 <**script** type=**"text/javascript"** src=**"/javascripts/jquery-3.2.1.min.js"**></**script**>  
</**head**>  
<**body**>  
 <**div**>  
 <**input** id=**"usernameTxt"** type=**"text"**/>  
 <**input** id=**"passwordTxt"** type=**"password"**/>  
 <**input** id=**"btn"** type=**"button"** value=**"提交"**/>  
 </**div**>  
 <**script** type=**"text/javascript"**>  
 $('#btn').on('click',function() {  
 var name = $('#usernameTxt').val();  
 var pwd = $('#passwordTxt').val();  
  
 var obj = {  
 username: name,  
 password: pwd  
 };  
  
 $.ajax({  
 type: 'POST',  
 url: '/login',  
 data: obj,  
 success: function(*data*){  
 if(*data*.status){  
 window.location.href = '/';  
 }else{  
 alert(status + '登录失败');  
 }  
 },  
 error: function(*XMLHttpRequest*, *textStatus*, *errorThrown*) {  
 console.log(*textStatus*);//若服务端返回数据有误时，错误状态信息  
 console.log(*errorThrown*);  
 },  
 dataType: 'json' //返回信息必须是json格式  
 });  
 });  
 </**script**>  
</**body**>  
</**html**>

后端代码：

var *express* = require('express');  
var *router* = *express*.Router();  
var *session* = require('express-session');  
  
*router*.post('/',function(*req*,*res*,*next*) {  
  
 var name = *req*.param('username');  
 var pwd = *req*.param('password');  
 console.log(name,pwd);  
  
 if(name === 'admin' && pwd === '123456'){  
 *session*.user = {  
 username : name,  
 password : pwd  
 };  
 return *res*.send({ //返回 json结果  
 status : 1,  
 msg : '登录成功'  
 });  
 }else{  
 return *res*.send({  
 status : 0,  
 msg : '登录失败'  
 });  
 }  
});  
  
*module*.exports = *router*;