# **[Node js之使用应用生成器来搭建第一个基于express的应用](https://www.cnblogs.com/FE-yanyi1993/p/6413042.html)**

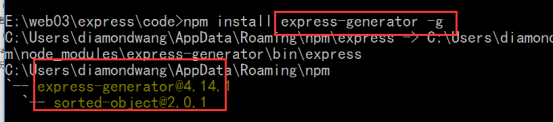
以前的开发中在构建一个express项目时，还是需要我们写很多的代码，还不够强大。我们可以借助应用生成器，帮我们把项目的基本骨架搭建好。一般提到开发，通常有如下几种：

* 原生开发，所有的代码从0开始。**自己做，一砖一瓦。**
* 框架开发，底层的代码，人家已经帮我们写好了，我们要写的是具体的功能。**买的毛坯房，房子成型了，但还不直接入住，需要装修。**
* 二次开发，本身就已经是一个完整的项目，能够运行，但是有些地方需要修改，在原来的基础之上进行开发。**已装修的房子或二手房，拎包入住**。

## **1.**安装生成器 (npm install express-generator -g)****

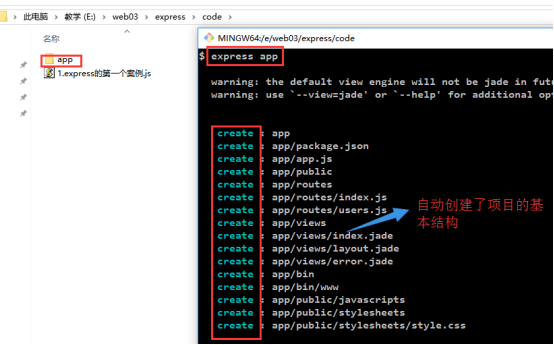
　　所谓的生成器，可以理解为脚手架工具。

　　需要全局安装，确保在任何目录下都可以使用。



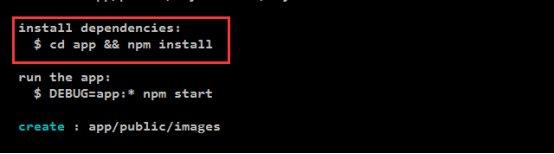
## **2.**创建应用 (express 项目名称)****

　　在指定目录下，使用**express 项目名称** 来创建应用。



## **3.**安装依赖包 (cd目录 && npm install)****

　　按照命令行的提示，操作即可。



　　这其实是两个命令

          1.cd app，切换目录，进入app目录

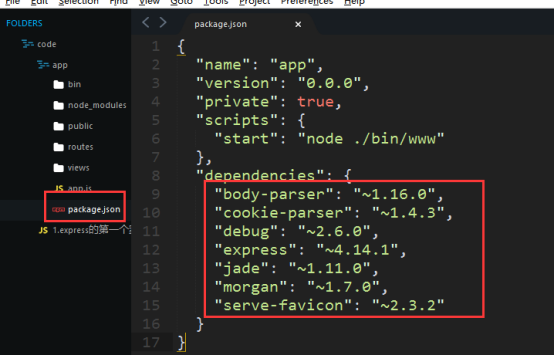
　　　 2.npm install，安装依赖包

　　说明：这个两个命令，可以使用 逻辑与 （&&） 形成一个命令。如果左边的成立的话， 右边会执行。如果左边的不成立的话，右边也不会执行。

　　我们也可以分开写。



　　实际上，在项目的根目录下，有一个配置文件---package.json，如下：

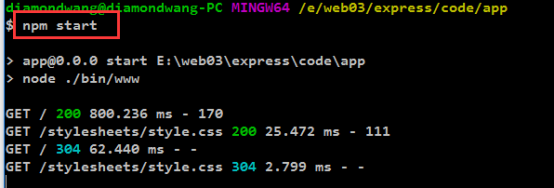


　　其中有一个配置项 --- dependencies，就列出了当前这个项目所依赖的包（模块）。如果我们在安装包的使用，没有指定包名，它就会找这个package.json文件，逐个安装dependencies中所列出的这些包和模块。

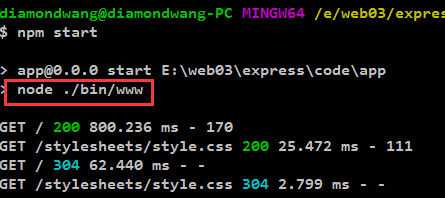
## **4.**启动应用 (npm start 或 node ./bin/www)****

　　按照刚才第二步的提示。

　　直接使用npm start 命令启动项目。

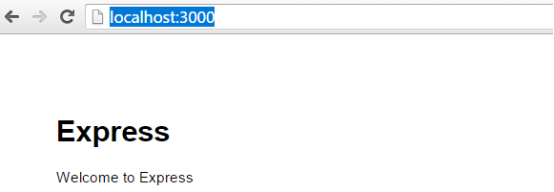


　　其中，启动应用时，还可以使用**node ./bin/www**

****

## **5.**浏览 (localhost:3000)****

　　在浏览器中使用 localhost:3000访问之，如下：

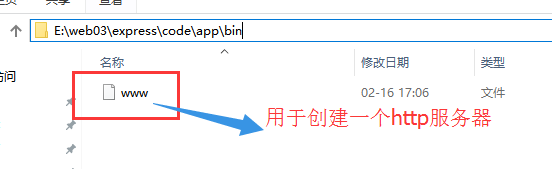


## **6.**项目结构分析****

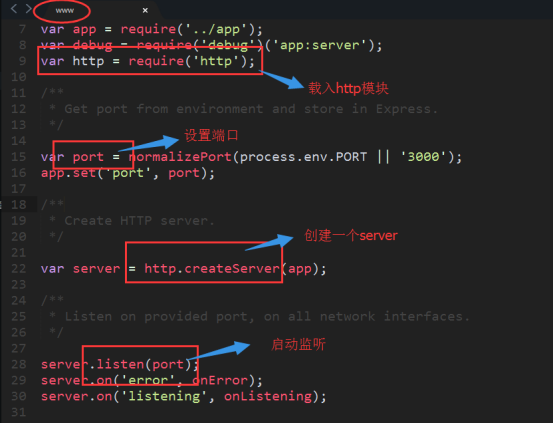
　　我们需要搞清楚，项目结构，然后才可以自己写代码。

　　项目目录结构划分如下：

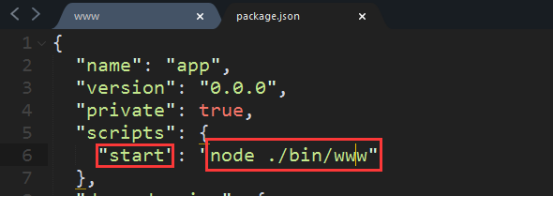
**bin目录**，如下



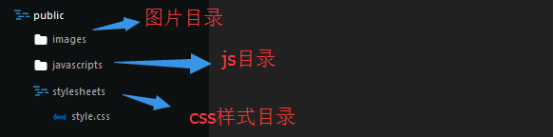
　　其核心的代码如下：



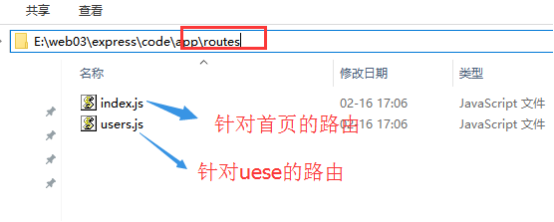
　　刚才执行 npm start命令，其实就是执行这个www文件。



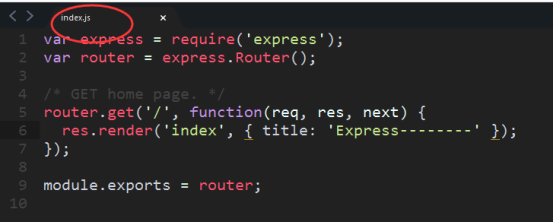
**public目录：**存放的就是静态资源目录，所谓的静态资源就是指图片、css和js文件。



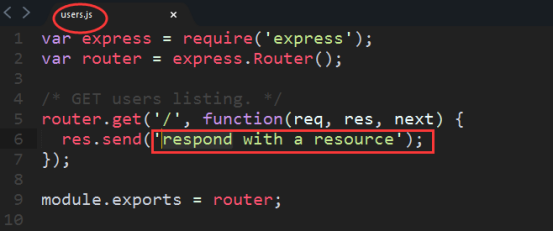
　　routes目录：路由目录，保存的是路由文件。



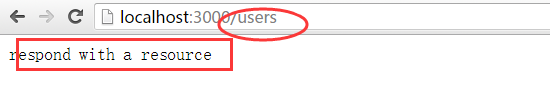
　　其中，index.js是实现首页路由的，如下：



　　而user.js则是实现/user路由的。

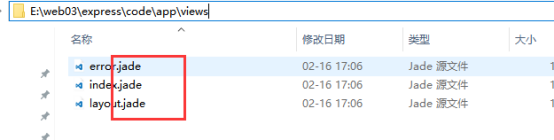


　　访问 /users，效果如下：

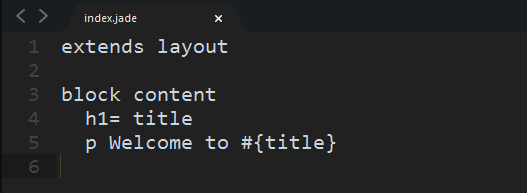


　　如果我们需要增加新的路由，就可以在这里写上路由的处理。

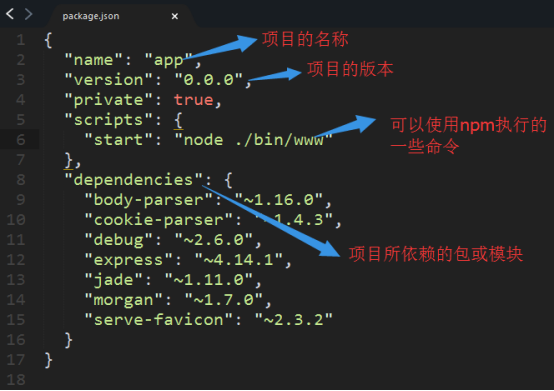
**Views目录：**视图目录，也就说存在模板文件的，这些模板文件，其实都是基于html的，Express默认使用了jade模板。



　　我们还可以使用ejs模板。其中index.jade，就是访问首页时，载入的模板，如下：



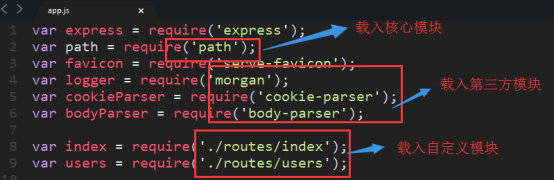
**package.json：**项目的配置文件。任何基于npm的的项目，都应该具备这个package.json文件。



**app.js：**整个项目的入口文件。

　　功能分解如下：

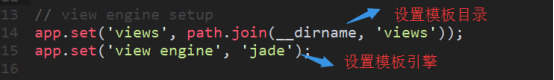
**第一步，载入相应模块**

****

**第二步，实例化一个app对象**



**第三步，设置模板相关参数**

****

**第四步，使用中间件**

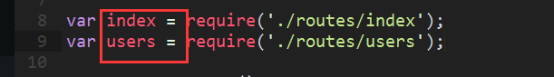
****

　　不同的中间件实现的是不同的功能。其中，如下两个是实现路由功能

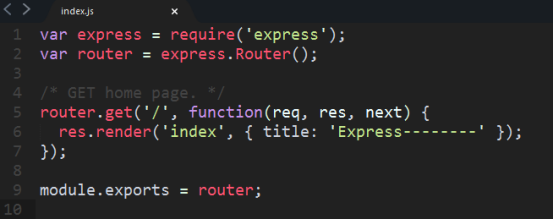
IMG_279

　　如果输入的是localhost:3000/，就交给index处理，如果输入的是localhost:3000/users，就交给users处理

　　Index和users就是上面引入的自定义模块



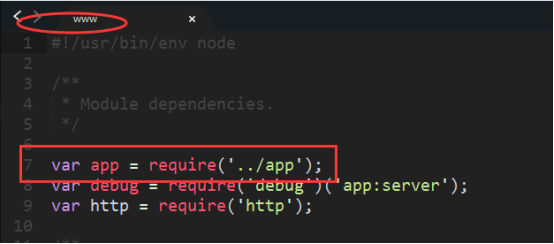
　　查看routes/index.js，就知道是怎么回事了



**第五步，导出模块**

**IMG_282**

　　导出给WWW文件来使用。



　　当服务器开启的时候，app对象就已经可以使用了。