**问：**用express初始化一个项目，为什么npm start可以启动项目node app.js启动不了？

**答：**express迁到4.x版本后，使用express-generator的话项目中会有一个bin/www文件，而且，在package.json中的入口项”scripts/start”的配置是bin/www，也就是npm start 调用的入口在 ./bin/www, 然后在其中调用到app.js。（node ./bin/www == npm start）

node app.js 只是直接运行app.js文件，4.x版本不能启动项目了。

**express -Git YourNewApp** //创建express项目时带着.gitignore等

**var app = express();**

* app是执行express后的结果，不区分app和express。
* 可以认为，在express内部，有一个函数的数组，暂时叫这个数组tasks，每来一个请求express内部会依次执行这个数组中的函数（这里说依次并不严谨，每个函数必须满足一定条件才行，这个后面说），应该可以想到，在这个函数数组里，每个函数的签名应该像下面那样

function(req, res, next){// …}

**对于典型的使用express的app.js，结构如下**

1. 导入相关模块

2. 执行过 var app = express() 后，

* 使用app.set 设置express内部的一些参数（options）
* 使用app.use 来注册函数，可以简单的认为是向那个（被我叫做）tasks的数组进行push操作

3. 通过http.createServer 用app来处理请求：express 4.x中将这部分放在./bin/www中

**对Express中Next的理解**

* 我们在定义express中间件函数的时候都会将第三个参数定义为next，这个next就是我们今天的主角，next函数主要负责将控制权交给下一个中间件，如果当前中间件没有终结请求，并且next没有被调用，那么请求将被挂起，后边定义的中间件将得不到被执行的机会。
* next函数主要是用来确保所有注册的中间件被一个接一个的执行，那么我们就应该在所有的中间件中调用next函数，但有一个特例，如果我们定义的中间件终结了本次请求，那就不应该再调用next函数，否则就可能会出问题
* 有一点需要特别指出，其实我们在定义路由中间件的时候函数的第三个参数next和我们定义非路由中间件的函数的第三个参数next不是同一个next。
* 路由中间件：负责同一个路由的多个中间件的控制权的传递，并且它会接收一个参数"route"，如果调用next(“route”)，则会跳过当前路由的其它中间件，直接将控制权交给下一个路由。

**路由route** 跟Router不一样

<http://host[:port][path>]

http 协议； host 主机； path 指定请求资源的URL。路由将决定怎么**处理**路径path.

app.get('/', function (req, res) {

res.send('Hello World!');

});

添加了一条路由，指定“/”这个路径由get的第二个参数所代表的函数来处理。

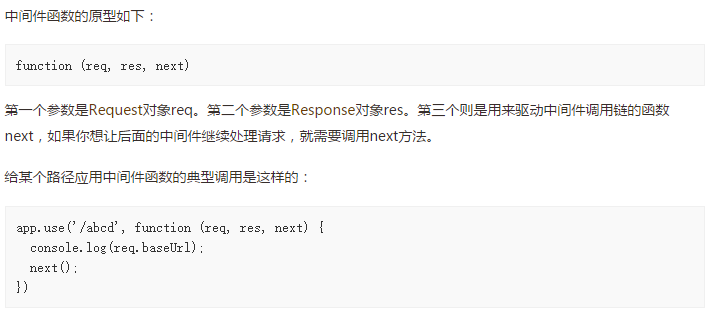
app.METHOD(path, callback, [,callback …..]) //METHOD可以是GET,POST等HTTP方法

**中间件**

在收到请求后和发送响应之前这个阶段执行的一些函数

app.use([path,] function [,function…]);

当app.use没有提供path参数时，路径默认为“/”。当你为某个路径安装了中间件，则当以该路径为基础的路径被访问时，都会应用该中间件。比如你为“/abcd”设置了中间件，那么“/abcd/xxx”被访问时也会应用该中间件。



Express.static: 处理网站里的静态文件的GET请求



**Router**

Express还提供了一个叫做Router的对象，行为很像中间件，你可以把Router直接传递给app.use，像使用中间件那样使用Router。另外你还可以使用router来处理针对GET、POST等的路由，也可以用它来添加中间件，总之你可以将Router看作一个微缩版的app

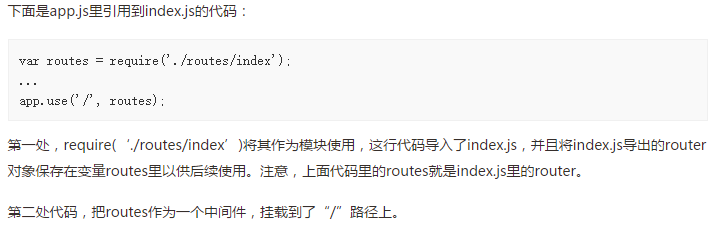
var express = require('express');

var router = express.Router();

router.METHOD(path, callback, [,callback …..]) //METHOD可以是GET,POST等HTTP方法

**路由模块**

Routes目录下index.js创建了一个Router实例，然后调用router.get为“/”路径应用了路由函数。最后使用module.exports将Router对象导出。

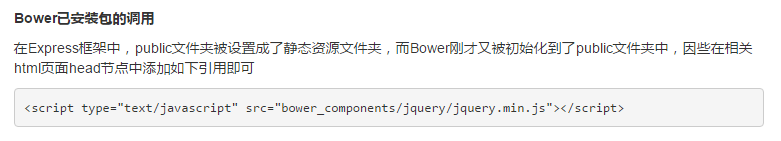


**使用bower配置前端依赖**

.bowerrc： directory下载的包存放位置

Bower init：自动生成bower.json

Bower install xxx # –save：安装指定版本的包并保存到bower.json文件中



**前端框架部分**

