# 实验7：express框架

实验背景

利用已掌握的知识，成功安装express环境，并完成相关实例。如图所示：

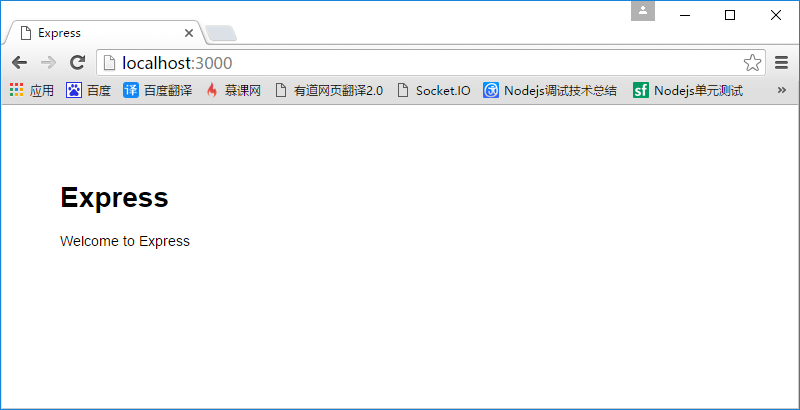


图1-1

实验目的

1．掌握express的安装

2．掌握工程结构

3．掌握路由控制及参数设置

4. 掌握模板的创建

实验分析

本次实验要完成的有三个方面的知识。第一：安装express并进行测试，创建一个简单的工程 。第二：创建工程，第三：创建模板。

实验步骤

1. 安装express

1、Express4

Express是一个简洁、灵活的node.js Web应用开发框架，它提供一系列强大的功能，比如：模板解析、静态文件服务、中间件、路由控制等。还可以使用插件或整合其他模块来创建各种Web和移动设备应用。

Express发展很快，且版本之间有较大差异，我们直接安装最新的Express4。

2、执行安装命令：

npm install express -gd

其中选项-g表示全局安装，目标模块将会被安装到NODE\_PATH的lib里面。-d选项表示一并安装依赖模块。没有-g选项的话会在当前目录（通常是项目目录）建立一个node\_modules目录。

安装成功后，命令行会提示 npm info ok

express 4.0之后将命令工具分出来了，故还需安装express-generator

3、安装express-generator命令

npm install -gd express-generator

安装配置完成

4、测试express的版本

express --version

5、使用Express创建project

express -e HelloExpress

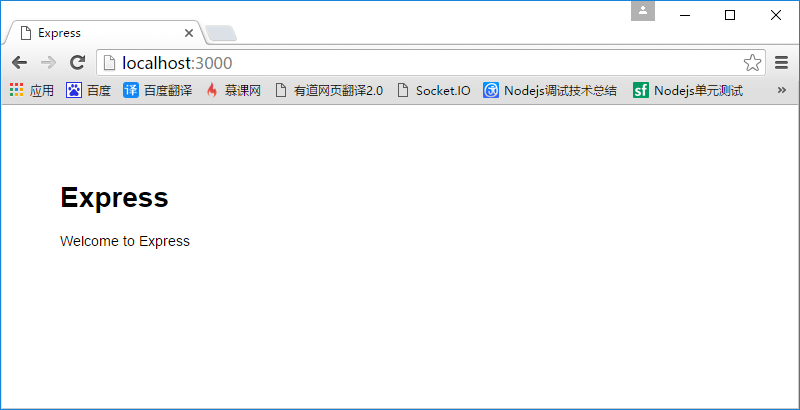
6、在本地添加依赖（依赖项定义在package.json中）：

npm install

7、启动

npm start

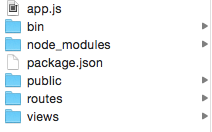
8、然后可以在浏览器通过 http://localhost:3000/ 访问了



图为测试结果

1. 创建工程

1.创建demo工程，工程结构如下：



2、查看app.js

每行代码的含义如下：

(1) var app = express()：生成一个express实例 app。  
(2)app.set('views', path.join(\_\_dirname, 'views’))：设置 views 文件夹为存放视图文件的目录, 即存放模板文件的地方,\_\_dirname 为全局变量,存储当前正在执行的脚本所在的目录。  
(3)app.set('view engine', 'ejs’)：设置视图模板引擎为 ejs。  
(4)app.use(favicon(\_\_dirname + '/public/favicon.ico’))：设置/public/favicon.ico为favicon图标。  
(5)app.use(logger('dev’))：加载日志中间件。  
(6)app.use(bodyParser.json())：加载解析json的中间件。  
(7)app.use(bodyParser.urlencoded({ extended: false }))：加载解析urlencoded请求体的中间件。  
(8)app.use(cookieParser())：加载解析cookie的中间件。  
(9)app.use(express.static(path.join(\_\_dirname, 'public')))：设置public文件夹为存放静态文件的目录。  
(10)app.use('/', routes);和app.use('/users', users)：路由控制器。  
(11)

app.use(function(req, res, next) {

var err = new Error('Not Found');

err.status = 404;

next(err);

});

捕获404错误，并转发到错误处理器。  
(12)

if (app.get('env') === 'development') {

app.use(function(err, req, res, next) {

res.status(err.status || 500);

res.render('error', {

message: err.message,

error: err

});

});

}

开发环境下的错误处理器，将错误信息渲染error模版并显示到浏览器中。  
(13)

app.use(function(err, req, res, next) {

res.status(err.status || 500);

res.render('error', {

message: err.message,

error: {}

});

});

生产环境下的错误处理器，不会将错误信息泄露给用户。  
(14)module.exports = app ：导出app实例供其他模块调用。

1. 创建模板

1.创建post.js文件，设计代码如下：

**index.ejs**

<!DOCTYPE html>

<html>

<head>

<title><%= title %></title>

<link rel='stylesheet' href='/stylesheets/style.css' />

</head>

<body>

<h1><%= title %></h1>

<p>Welcome to <%= title %></p>

</body>

</html>

扩展练习

1.思考并实现，完成案例《7.1 app.js》并实现测试

参考学习资料：详见案例《7.1 app.js》源文件

实验思考

1.思考并实现，案例《7.2 get.js》、案例《7.3 post.js》。