

Express 安装和使用

主讲教师：(大地)

合作网站：www.itying.com

目录

一、	Express 简单介绍.....	1
二、	Express 安装使用.....	2
三、	Express 框架中的路由	3
四、	Express 框架中 ejs 的安装使用	5
五、	利用 Express.static 托管静态文件.....	7
六、	Express 中间件.....	8
七、	获取 Get Post 请求的参数.....	11

一、Express 简单介绍

Express 是一个基于 Node.js 平台，快速、开放、极简的 web 开发框架

Express 框架是后台的 Node 框架，所以和 jQuery、zepto、yui、bootstrap 都不一个东西。Express 在后台的受欢迎的程度类似前端的 jQuery，就是企业的事实上的标准。

Express 特点：

1. Express 是一个基于 Node.js 平台的极简、灵活的 web 应用开发框架，它提供一系列强大的特性，帮助你创建各种 Web 和移动设备应用。
2. 丰富的 HTTP 快捷方法和任意排列组合的 Connect 中间件，让你创建健壮、友好

的 API 变得既快速又简单

3. Express 不对 Node.js 已有的特性进行二次抽象，我们只是在它之上扩展了 Web 应用所需的基本功能。

Express 官网：

英语官网：<http://expressjs.com/>

中文官网：<http://www.expressjs.com.cn/>
<http://www.expressjiaocheng.com>

二、Express 安装使用

安装：

安装 Express 框架，就是使用 npm 的命令。

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

--save 参数，表示自动修改 package.json 文件，自动添加依赖项。

简单使用：

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

//1.引入

```
var express = require('express');  
var app = express();
```

//2.配置路由

```
app.get('/', function (req, res) {  
  res.send('Hello World!');  
});
```

//3.监听端口

```
app.listen(3000,'127.0.0.1');
```

完整 Demo

```
var express=require('express'); /*引入 express*/

var app=new express();/*实例化 express 赋值给 app*/

//配置路由 匹配 URI 地址实现不同的功能

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

    res.send('首页');

})

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

    res.send('搜索');
    //?keyword=华为手机&enc=utf-8&suggest=1.his.0.0&wq
})

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

    res.send('登录');

})

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

    res.send('注册');

})

app.listen(3000,"127.0.0.1");
```

三、Express 框架中的路由

路由（Routing）是由一个 URI（或者叫路径）和一个特定的 HTTP 方法（GET、POST 等）组成的，涉及到应用如何响应客户端对某个网站节点的访问

简单的路由配置

当用 `get` 请求访问一个网址的时候，做什么事情：

```
app.get("网址",function(req,res){  
  
});
```

当用 `post` 访问一个网址的时候，做什么事情：

```
app.post("网址",function(req,res){  
  
});
```

```
// user 节点接受 PUT 请求  
app.put('/user', function (req, res) {  
    res.send('Got a PUT request at /user');  
});
```

```
// user 节点接受 DELETE 请求  
app.delete('/user', function (req, res) {  
    res.send('Got a DELETE request at /user');  
});
```

动态路由配置：

```
app.get("/user/:id",function(req,res){  
    var id = req.params["id"];  
    res.send(id);  
});
```

路由的正则匹配：（了解）

```
app.get('/ab*cd', function(req, res) {  
    res.send('ab*cd');  
});
```

路由里面获取 `Get` 传值

/news?id=2&sex=nan

```
app.get('/news', function(req, res) {  
  console.log(req.query);  
});
```

四、Express 框架中 ejs 的安装使用

Express 中 ejs 的安装:

```
npm install ejs --save
```

或者:

```
npm install ejs --save-dev
```

Express 中 ejs 的使用:

```
var express = require("express");  
  
var app = express();  
  
app.set("view engine", "ejs");  
  
app.get("/", function(req, res){  
  res.render("news", {  
    "news": ["我是小新闻啊", "我也是啊", "哈哈哈哈哈"]  
  });  
});  
  
app.listen(3000);
```

指定模板位置，默认模板位置在 views

```
app.set('views', __dirname + '/views');
```

Ejs 引入模板

```
<%- include header.ejs %>
```

Ejs 绑定数据

```
<%=h%>
```

Ejs 绑定 html 数据

```
<%-h%>
```

Ejs 模板判断语句

```
<% if(true){ %>

    <div>true</div>

<%} else{ %>

    <div>false</div>

<%} %>
```

Ejs 模板中循环数据

```
<%for(var i=0;i<list.length;i++) { %>
    <li><%=list[i] %></li>
<%}%>
```

Ejs 后缀修改为 Html

这是一个小技巧，看着.ejs 的后缀总觉得不爽，使用如下方法，可以将模板文件的后缀换成我们习惯的.html。

1.在 app.js 的头上定义 ejs;代码如下：

```
var ejs = require('ejs');
```

2.注册 html 模板引擎代码如下:

```
app.engine('html',ejs.__express);
```

3.将模板引擎换成 html 代码如下:

```
app.set('view engine', 'html');
```

4.修改模板文件的后缀为.html。

五、利用 Express.static 托管静态文件

1、如果你的静态资源存放在多个目录下面，你可以多次调用 `express.static` 中间件：

```
app.use(express.static('public'));
```

现在，`public` 目录下面的文件就可以访问了。

```
http://localhost:3000/images/kitten.jpg
```

```
http://localhost:3000/css/style.css
```

```
http://localhost:3000/js/app.js
```

```
http://localhost:3000/images/bg.png
```

```
http://localhost:3000/hello.html
```

2、如果你希望所有通过 `express.static` 访问的文件都存放在一个“虚拟（virtual）”目录（即目录根本不存在）下面，可以通过为静态资源目录指定一个挂载路径的方式来实现，如下所示：

```
app.use('/static', express.static('public'));
```

现在，你就爱可以通过带有 “/static” 前缀的地址来访问 public 目录下面的文件了。

```
http://localhost:3000/static/images/kitten.jpg  
http://localhost:3000/static/css/style.css  
http://localhost:3000/static/js/app.js  
http://localhost:3000/static/images/bg.png  
http://localhost:3000/static/hello.html
```

六、Express 中间件

Express 是一个自身功能极简，完全是由路由和中间件构成一个的 web 开发框架：从本质上来说，一个 Express 应用就是在调用各种中间件。

中间件 (Middleware) 是一个函数，它可以访问请求对象 (request object (req))，响应对象 (response object (res))，和 web 应用中处理请求-响应循环流程中的中间件，一般被命名为 next 的变量。

中间件的功能包括：

- 执行任何代码。
- 修改请求和响应对象。
- 终结请求-响应循环。
- 调用堆栈中的下一个中间件。

如果我的 get、post 回调函数中，没有 next 参数，那么就匹配上第一个路由，就不会往下匹配了。如果想往下匹配的话，那么需要写 `next()`

Express 应用可使用如下几种中间件：

应用级中间件

路由级中间件
错误处理中间件
内置中间件
第三方中间件

1. 应用级中间件

```
app.use(function(req,res,next){ /*匹配任何路由*/
    //res.send('中间件');

    console.log(new Date());

    next(); /*表示匹配完成这个中间件以后程序继续向下执行*/
})

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

    res.send('根');

})

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

    res.send('首页');

})
```

2. 路由中间件

```
app.get("/",function(req,res,next){
    console.log("1");
    next();
});

app.get("/",function(req,res){
    console.log("2");
});
```

3. 错误处理中间件

```
app.get('/index',function(req,res){
    res.send('首页');
})

/*中间件相应 404*/
app.use(function(req,res){
    //res.render('404',{});
    res.status(404).render('404',{});
})
```

4. 内置中间件

```
//静态服务 index.html
app.use('/static',express.static("./static"));    /*匹配所有的路径*/

app.use('/news',express.static("./static"));    /*匹配所有的路径*/
```

5. 第三方中间件

body-parser 中间件 第三方的 获取 post 提交的数据

1.cnpm install body-parser --save

2.var bodyParser = require('body-parser')

3. 设置中间件

//处理 form 表单的中间件

```
// parse application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: false })); form 表单提交的数据

// parse application/json
app.use(bodyParser.json()); 提交的 json 数据的数据

4.req.body 获取数据
```

七、获取 Get Post 请求的参数

● GET 请求的参数在 URL 中，在原生 Node 中，需要使用 url 模块来识别参数字符串。在 Express 中，不需要使用 url 模块了。可以直接使用 req.query 对象。

● POST 请求在 express 中不能直接获得，可以使用 body-parser 模块。使用后，将可以用 req.body 得到参数。但是如果表单中含有文件上传，那么还是需要使用 formidable 模块。

1.安装

```
npm install body-parser
```

2.使用 req.body 获取 post 过来的参数

```
var express = require('express')
var bodyParser = require('body-parser')

var app = express()

// parse application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: false }))

// parse application/json
app.use(bodyParser.json())

app.use(function (req, res) {
  res.setHeader('Content-Type', 'text/plain')
  res.write('you posted:\n')
```

```
res.end(JSON.stringify(req.body, null, 2))  
})
```