Express MongoDB

[一、 Express框架安装使用 1](#_Toc477788298)

[二、 Express框架中ejs安装使用 2](#_Toc477788299)

[三、 Express框架中的路由 3](#_Toc477788300)

[四、 Express中间件 5](#_Toc477788301)

[五、 GET请求和POST请求的参数 8](#_Toc477788302)

[六、 Node.js操作MongoDB 9](#_Toc477788303)

[七、 formidable图片上传 12](#_Toc477788304)

[八、 express xheditor文本编辑器使用 13](#_Toc477788305)

[九、 DB库封装 14](#_Toc477788306)

# Express框架安装使用

**Express简单介绍：**

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。

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，和jQuery一样，就是企业的事实上的标准。

● 原生Node开发，会发现有很多问题。比如：

■ 呈递静态页面很不方便，需要处理每个HTTP请求，还要考虑304问题

■ 路由处理代码不直观清晰，需要写很多正则表达式和字符串函数

■ 不能集中精力写业务，要考虑很多其他的东西

● 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的哲学是在你的想法和服务器之间充当薄薄的一层。这并不意味着他不够健壮，或者没有足够的有用特性，而是尽量少干预你，让你充分表达自己的思想，同时提供一些有用的东西。

英语官网：http://expressjs.com/

中文官网：<http://www.expressjs.com.cn/>

**Express安装使用：**

安装Express框架，就是使用npm的命令。

|  |
| --- |
| 1. npm install **--save** express |

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

路由能力：

|  |
| --- |
| 1. var express = require("express"); 2. var app = express(); /\*实例化使用\*/ 3. app.get("/",function(req,res){ 4. res.send("你好"); 5. }); 6. app.get("/haha",function(req,res){ 7. res.send("这是haha页面，哈哈哈哈哈哈"); 8. }); 9. app.get(/^\/student\/([\d]{10})$/,function(req,res){ 10. res.send("学生信息，学号" + req.params[0]); 11. }); 12. app.get("/teacher/:gonghao",function(req,res){ 13. res.send("老师信息，工号" + req.params.gonghao); 14. }); 15. app.listen(3000); |
|  |

静态文件服务能力：

|  |
| --- |
| 1. app.use(express.static("./public")); |

# Express框架中ejs安装使用

**Express中ejs的安装使用：**

|  |
| --- |
| **安装：npm install ejs –save-dev**  var express = require("express");  var app = express();  app.set("view engine","ejs");  app.get("/",function(req,res){  res.render("haha",{  "news" : ["我是小新闻啊","我也是啊","哈哈哈哈"]  });  });  app.listen(3000); |

**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。 |

**指定模板位置**

|  |
| --- |
| app.set('views', \_\_dirname + '/views'); |

**ejs引入模板**

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

**ejs绑定html数据**

|  |
| --- |
| <%-h%> |

# Express框架中的路由

当用get请求访问一个网址的时候，做什么事情：

|  |
| --- |
| 1. app.get("网址",function(req,res){ 3. }); |

当用post访问一个网址的时候，做什么事情：

|  |
| --- |
| 1. app.post("网址",function(req,res){ 3. }); |

如果想处理这个网址的任何method的请求，那么写all

|  |
| --- |
| 1. app.all("/",function(){ 3. }); |

这里的网址，不分大小写，也就是说，你路由是

|  |
| --- |
| 1. app.get("/AAb",function(req,res){ 2. res.send("你好"); 3. }); |

实际上小写的访问也行。

所有的GET参数，? 后面的都已经被忽略。 锚点#也被忽略

你路由到/a ， 实际/a?id=2&sex=nan 也能被处理。

正则表达式可以被使用。正则表达式中，未知部分用圆括号分组，然后可以用req.params[0]、[1]得到。

req.params类数组对象。

|  |
| --- |
| 1. app.get(**/^\/student\/([\d]{10})$/**,function(req,res){ 2. res.send("学生信息，学号" + **req.params[0]**); 3. }); |

冒号是更推荐的写法。

|  |
| --- |
| 1. app.get("/student/:id",function(req,res){ 2. var id = req.params["id"]; 3. var reg= /^[\d]{6}$/; //正则验证 4. if(reg.test(id)){ 5. res.send(id); 6. }else{ 7. res.send("请检查格式"); 8. } 9. }); |

表单可以自己提交到自己上。

|  |
| --- |
| 1. app.get("/",function(req,res){ 2. res.render("form"); 3. }); 4. app.post("/",function(req,res){ 5. //将数据添加进入数据库 6. res.send("成功"); 7. }); |

适合进行 RESTful路由设计。简单说，就是一个路径，但是http method不同，对这个页面的使用也不同。

/student/345345

get 读取学生信息

add 添加学生信息

delete 删除学生新

# 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()**

|  |
| --- |
| app.get("/",function(req,res,**next**){  console.log("1");  **next();**  });  app.get("/",function(req,res){  console.log("2");  }); |
|  |

# GET请求和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，必须使用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))  }) |

# 六、Node.js操作MongoDB

**需要引包：**

|  |
| --- |
| npm install mongodb --save-dev / cnpm install mongodb --save-dev |

**网址：**

<https://docs.mongodb.com/getting-started/node/introduction/>

<https://docs.mongodb.com/getting-started/node/query/>

**var** rel=db.collection(**'user'**).find({**"name"**: { **$regex**:**new** RegExp(name)} });

**连接数据库：**

|  |
| --- |
| **var *express*** = *require*(**"express"**); *//数据库引用* **var *MongoClient*** = *require*(**'mongodb'**).**MongoClient**;  **var *app*** = ***express***();  *//数据库连接的地址，最后的斜杠表示数据库名字* **var *shujukuURL*** = **'mongodb://localhost:27017/news'**;  ***app***.get(**"/"**,**function**(req,res){  *//连接数据库，这是一个异步的操作* ***MongoClient***.connect(***shujukuURL***, **function**(err, db) {  res.**writeHead**(200,{**"Content-Type"**:**"text/html;charset=UTF8"**});  **if**(err){  res.send(**"数据库连接失败"**);  **return**;  }  res.write(**"恭喜，数据库已经成功连接 \n"**);  db.collection(**"user"**).insertOne({**"name"**:**"哈哈"**},**function**(err,result){  **if**(err){  res.send(**"数据库写入失败"**);  **return**;  }  res.write(**"恭喜，数据已经成功插入"**);  res.end();  *//关闭数据库* db.close();  });  }); });  ***app***.listen(8020); |

**查找据库：**

|  |
| --- |
| ***MongoClient***.connect(***dbUrl***,**function**(err,db){   **if**(err){ */\*数据库连接失败\*/* ***console***.log(**'数据库连接失败'**);  **return**;  }   **var** result=[];   **var** userRel=db.collection(**'user'**).find();   *//res.send(userRel);* userRel.each(**function**(err, doc) {  **if**(err){  res.write(**"游标遍历错误"**);  **return**;  }  **if** (doc != **null**) {  result.push(doc);  } **else** {  ***console***.log(result);  *//遍历完毕* db.close();  res.render(**"index"**,{  **"result"** : result  });  }  });  }) |

**增加数据:**

|  |
| --- |
| ***MongoClient***.connect(***dbUrl***,**function**(err,db){  **if**(err){  **return** }  db.collection(**'user'**).insertOne({  **"name"** : name,  **"age"** : age,  **"score"** : {  **"shuxue"** : shuxuechengji,  **"yuwen"** : yuwenchengji  }  },**function**(err,result){  **if**(err){  ***console***.log(**'写入数据失败'**);  }  *//关闭数据库* db.close();  *//res.redirect('/add');* res.redirect(**'/'**); */\*路由跳转\*/* res.end();  *////res.location('/add')* })  }) |

**修改数据：**

|  |
| --- |
| ***MongoClient***.connect(***dbUrl***,**function**(err,db){  **if**(err){  ***console***.log(**'数据库连接错误'**);  **return**;  }  db.collection(**'user'**).updateOne({**"\_id"**:***ObjectID***(id)}, {   **"name"**: name,  **"age"**: age,  **"score"**: {  **"shuxue"**: shuxue,  **"yuwen"**: yuwen  }  },**function** (err, results) {  ***console***.log(results);  db.close();   res.redirect(**'/'**); */\*路由跳转\*/* res.end(**'end'**);  }) }) |

**删除数据：**

|  |
| --- |
| ***MongoClient***.connect(***dbUrl***,**function**(err,db){  **if**(err){  **throw new** Error(**"数据库连接失败"**);  **return**;  }  db.collection(**'user'**).deleteOne({**"\_id"**:***ObjectID***(id)},**function**(error,result){  **if**(error)  {  **throw new** Error(**'删除数据失败'**);  **return**;  }  db.close();  res.redirect(**'/'**); */\*路由跳转\*/* }) }) |

# formidable图片上传

**注意：**文件图片上传在form表单中必须加入enctype="multipart/form-data"

s

**Enctype:**

<http://blog.csdn.net/xiaojianpitt/article/details/6856536>

**formidable图片上传插件：**<https://www.npmjs.com/package/formidable>

1.安装

|  |
| --- |
| npm i -S formidable |

2.使用

|  |
| --- |
| var formidable = require('formidable'), util = require('util');  var sd = require("silly-datetime");  var path = require("path");  var form = new formidable.IncomingForm();  form.uploadDir = "./uploads";  form.parse(req, function(err, fields, files) {  var date = sd.format(new Date(), 'YYYYMMDDHHmmss');  var extname = path.extname(files.face.name);  //执行改名  var oldpath = \_\_dirname + "/" + files.face.path;  //新的路径由三个部分组成：时间戳、随机数、拓展名  var newpath = \_\_dirname + "/uploads/" + date + extname;  //改名  fs.rename(oldpath,newpath,function(err){  if(err){  throw Error("改名失败");  }  });  }); |

# express xheditor文本编辑器使用

1. **下载xheditor 放入程序静态目录中：**

**注意: 必须设置静态目录 app.use(express.static(path.join(\_\_dirname, 'static')));**

<http://xheditor.com/download.html>

**2.在模板中引入 xheditor**

|  |
| --- |
| <script src="./xheditor/jquery/jquery-1.11.2.min.js" charset="utf-8"></script>  <script src="./xheditor/xheditor-1.2.2.min.js" charset="utf-8"></script>  <script src="./xheditor/xheditor\_lang/zh-cn.js" charset="utf-8"></script> |

**3.在模板中定义 textarea 显示编辑器**

|  |
| --- |
| <textarea name="yuwen" id="content" cols="120" rows="30"></textarea> |

**4.配置参数**

|  |
| --- |
| **$**(**'#content'**).xheditor({  **html5Upload**: **false**,  **upMultiple**: **'3'**,  **tools**:**'full'**,  **skin**:**'default'**,  **width**:500, **height**:400,  **upImgUrl**:**"/upload"**,  **upImgExt**:**"jpg,jpeg,gif,png"** }); |

**5.在服务中书写上传方法**

res.send(**'{"err":"","msg":"'**+uploadurl+**'"}'**)

*//给编辑器提供的上传图片的接口****app***.post(**'/upload'**,**function**(req,res){  
  
  
 **var** form = **new *multiparty***.**Form**();  
 form.**uploadDir**=**'./static/upload'** */\*设置图片上传的路径\*/* form.parse(req, **function**(err, fields, files) {  
  
 *//fields post过来的表单的文本内容  
 //  
 //files 包含吗了post过来的图片信息* ***console***.log(fields);  
  
 ***console***.log(files);  
  
 **var** path=files.filedata[0].path;  
  
  
 res.json({**"err"**:**""**,**"msg"**:path}) */\*给编辑器返回地址信息\*/* });  
})

# DB库封装

|  |
| --- |
| */\*\*  \* Created by Administrator on 2017/5/12 0012.  \*/ //封装数据库操作* **var *Mongo***=*require*(**'mongodb'**);  **var *Config***=*require*(**"./config.js"**)  **var *MongoClient***=***Mongo***.**MongoClient**;  **var *ObjectID***=***Mongo***.**ObjectID**;  *//数据库连接的方法* **function** *\_connectDB*(callback){   ***MongoClient***.connect(***Config***.**dbUrl**,**function**(err,db){   **if**(err){  ***console***.log(err);  **return**;  }  callback(db);   })  }  *//封装插入数据的方法 //json 数据* ***exports***.insertOne=**function**(collectionName,json,callback){   *\_connectDB*(**function**(db){  db.collection(collectionName).insertOne(json,**function**(err,data){   callback(err,data);  })   })  }  *//封装插入数据的方法 //json 数据* |