**Express框架**

Express 是一个简洁而灵活的 [node.js](https://baike.baidu.com/item/node.js/7567977" \t "https://baike.baidu.com/item/eXpress/_blank) Web应用框架, 提供一系列强大特性帮助你创建各种Web应用。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应用所需的功能。丰富的HTTP工具以及来自Connect框架的中间件随取随用，创建强健、友好的API变得快速又简单。

Express 是一个简洁而灵活的 node.js Web应用框架, 提供了一系列强大特性帮助你创建各种 Web 应用，和丰富的 HTTP 工具。

使用 Express 可以快速地搭建一个完整功能的网站。

Express 框架核心特性：

（1）可以设置中间件来响应 HTTP 请求。

（2）定义了路由表用于执行不同的 HTTP 请求动作。

1. 可以通过向模板传递参数来动态渲染 HTML 页面。

**二、express的使用**  
**1、安装**  
前提：在项目目录下要安装express框架（或者全局安装 -g）；   
win+R ——CMD（打开命令行窗口）——输入 npm install express -g；——安装完成后通过   
express -V 查看express安装的版本号，注意V大写   
**2、使用：**  
在js对应的文件模块中  
  
//引入所需的模块  
var http=require('http')  
var express=require('express')  
//创建express对象app  
var app=express();  
//在app上获取请求，响应内容  
app.get('/',function(req,res){   //通过get方式请求页面，'/'代表请求的是根目录（一般为首页）  
    res.send('欢迎访问首页');   //send（）方法发送数据到客户端  
    res.end();           //响应结束  
})  
//创建服务并监听  
http.createServer(app).listen(3000,function(err){  
   if(!err)  
    console.log('listening....')  //当监听成功会执行  
})

**3、总结：**   
1）、express就是一个web开发框架，基于nodejs，   
2）、express封装了很多nodejs的原生方法用于操作请求，响应数据等   
3）、app===express();

**三、express路由**

3.1 路由是什么？   
路由：router，是指如何定义应用 （请求的url）的端点urls以及如何响应客户端的请求，（路由实际就是url的分发，根据请求的url不同，执行不同的业务逻辑。）   
作用：分发数据，根据请求的不同，分发响应不同的内容；这里写图片描述

中间件  
  
在express中，中间件的实质就是函数，它们有一定的作用（如：增、删、改、查），起到过滤请求和响应的作用。  
>中间件：Middleware，本质为一个函数，它可以访问请求对象request(res),响应对象response（res），以及在web应用上处于请求-响应循环中的中间件。  
**4.1 功能**1、执行任何代码  
2、修改请求和响应对象  
3、终结请求-响应循环  
4、调用堆栈中的下一个中间件  
【注意】：如果当前中间件没有结束请求-响应循环，则必须调用next（）方法将控制权交给下一个中间伯，否则请求就会挂起。  
**4.2中间件的分类**  
1、应用级中间件——自定义中间件

1. express内置中间件（Static是express内置的唯一的中间件，它上面有很多方法）
2. 第三方中间件（如：body-parser）第三方中间件是在express的基础之上开发出来的，在使用之间要先安装相应的中间件，npm install body-parser(需求：拿到用post方式提交的表单数据)

**4.3、项目中的views层**  
在项目中，views层一般用来存放模板的（html），router层用来放路由模块的，public层用来存放一些静态的模板（如：css，img，js等）；   
【需求】用res.render(‘htmlpath’)的方式来渲染一个请求的页面  
**4.4、 总结**  
1、app.set(‘views’,path.join(\_\_dirname,’views’));——的作用是设置html文件模板的来源是views文件夹，当下面要发送views文件夹里的html文件到客户端时，只需要用’index.html’的方法，即不需要再写html相对于当前js文件的路径来查找要发送的html。   
2、使用模板引擎的步骤：   
1）、引入模板引擎require(’ejs‘）   
2）、定义一个模板引擎app.engine(‘html’,ejs.\_\_express);   
3）、设置该模板引擎的模板路径（用到path.join来获取根目录到views的绝对路径）   
app.set(‘views’,path.join(\_\_dirname,’views’));   
4）、设置模板引擎   
app.set(‘views engine’,’html’);   
5）、用新方式来发送html模板到客户端   
app.get(‘/’,function(req,res){   
res.render(‘index.html’)   
})；   
5、模板引擎的作用就在于方便在发送数据时动态的去响应页面

### **五、Express方法**

### **5.1、all方法和HTTP动词方法**

针对不同的请求，Express提供了use方法的一些别名。

var express = require("express");var http = require("http");var app = express();

app.all("\*", function(request, response, next) {

response.writeHead(200, { "Content-Type": "text/plain" });

next();});

app.get("/", function(request, response) {

response.end("Welcome to the homepage!");});

app.get("/about", function(request, response) {

response.end("Welcome to the about page!");});

app.get("\*", function(request, response) {

response.end("404!");});

http.createServer(app).listen(1337);

上面代码的all方法表示，所有请求都必须通过该中间件，参数中的“\*”表示对所有路径有效。get方法则是只有GET动词的HTTP请求通过该中间件，它的第一个参数是请求的路径。由于get方法的回调函数没有调用next方法，所以只要有一个中间件被调用了，后面的中间件就不会再被调用了。

除了get方法以外，Express还提供post、put、delete方法，即HTTP动词都是Express的方法。

这些方法的第一个参数，都是请求的路径。除了绝对匹配以外，Express允许模式匹配。

app.get("/hello/:who", function(req, res) {

res.end("Hello, " + req.params.who + ".");});

上面代码将匹配“/hello/alice”网址，网址中的alice将被捕获，作为req.params.who属性的值。需要注意的是，捕获后需要对网址进行检查，过滤不安全字符，上面的写法只是为了演示，生产中不应这样直接使用用户提供的值。

如果在模式参数后面加上问号，表示该参数可选。

app.get('/hello/:who?',function(req,res) {

if(req.params.id) {

res.end("Hello, " + req.params.who + ".");

}

else {

res.send("Hello, Guest.");

}});

下面是一些更复杂的模式匹配的例子。

app.get('/forum/:fid/thread/:tid', middleware)

// 匹配/commits/71dbb9c// 或/commits/71dbb9c..4c084f9这样的git格式的网址app.get(/^\/commits\/(\w+)(?:\.\.(\w+))?$/, function(req, res){

var from = req.params[0];

var to = req.params[1] || 'HEAD';

res.send('commit range ' + from + '..' + to);});

### **5.2、set方法**

set方法用于指定变量的值。

app.set("views", \_\_dirname + "/views");

app.set("view engine", "jade");

上面代码使用set方法，为系统变量“views”和“view engine”指定值。

### **5.3、response对象**

****（1）response.redirect方法****

response.redirect方法允许网址的重定向。

response.redirect("/hello/anime");response.redirect("http://www.example.com");response.redirect(301, "http://www.example.com");

****（2）response.sendFile方法****

response.sendFile方法用于发送文件。

response.sendFile("/path/to/anime.mp4");

****（3）response.render方法****

response.render方法用于渲染网页模板。

app.get("/", function(request, response) {

response.render("index", { message: "Hello World" });});

上面代码使用render方法，将message变量传入index模板，渲染成HTML网页。

### **5.4、requst对象**

****（1）request.ip****

request.ip属性用于获得HTTP请求的IP地址。

****（2）request.files****

request.files用于获取上传的文件。

### **5.5、搭建HTTPs服务器**

使用Express搭建HTTPs加密服务器，也很简单。

var fs = require('fs');var options = {

key: fs.readFileSync('E:/ssl/myserver.key'),

cert: fs.readFileSync('E:/ssl/myserver.crt'),

passphrase: '1234'};

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

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

res.send('Hello World Expressjs');});

var server = https.createServer(options, app);server.listen(8084);console.log('Server is running on port 8084');

**六、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。