

# express

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

- 模板解析
- 静态文件服务
- 中间件
- 路由控制

## 路由控制

- get方法 —— 根据请求 **路径** 来处理客户端发出的 **GET** 请求

```
app.get(path, function(request, response));
```

- path为请求的 **路径**
- 第二个参数为处理请求的 **回调函数**，有两个参数分别是request和response，代表请求信息和响应信息。
- app.all()函数可以匹配所有的HTTP **动词**，也就是说它可以匹配所有路径的请求，

## 中间件

- 中间件就是处理HTTP请求的 **函数**，用来完成各种特定的任务，比如检查用户是否登录、添加公共方法。
- 它最大的特点就是，一个中间件处理完，可以把相应数据再传递给下一个中间件。
- 如果调用回调函数的 **next** 参数表示将请求数据传递给下一个中间件。

```
app.use([path], function(request, response, next){}); //可选参数path默认为"/"
```



## 获取请求参数

- req.host返回请求头里取的 **主机名** (不包含端口号)。
- req.path返回请求的URL的 **路径名**。
- req.query是一个可获取客户端get请求 **查询字符串** 转成的对象，默认为{}。
- req.params是一个由 **路径参数** 组成的对象。

# send

send()方法向浏览器发送响应，并可以智能处理不同类型的数据。并地输出响应时会自动进行一些设置，比如header信息、http缓存支持等等。

- 当参数为一个String时，Content-Type默认设置为"text/html"。

```
res.send([body|status], [body]);
```

- 当参数为Array或Object时，Express会返回一个JSON

```
res.send({ user: 'tobi' }); //{"user":"tobi"}
```

- 不能使用数字作为参数，如果要返回入码要用 `res.sendStatus` 方法



# 模板

## 1. 指定渲染模板引擎

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

## 2. 设置放模板文件的目录

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

3. render函数，对网页模板进行渲染 在渲染模板时 **locals** 可为其模板传入变量值，在模板中就可以调用所传变量了，

```
: res.render(view, [locals], callback);
```

## 4. 原理

```
var tmpl = '<h1>{{name}}</h1><h1>{{age}}</h1>';  
var data = {name: 'zfx', age: 30};  
var html = tmpl.replace(/\{\{(\w+)\}\}/g, function(input, group) {  
    return data[group];  
})
```

## 静态文件服务中间件

`express.static` 是 Express 内置的唯一一个中间件,负责托管 Express 应用内的静态资源。

- 如果要在网页中加载静态文件 ( `css`、`js`、`img` ) , 就需要另外指定一个存放静态文件的 **目录**
- 项目目录下添加一个存放静态文件的目录为 `public`
- 在`public`目录下在添加三个存放 `js`、`css`、`img` 的目录,把相关文件放到相应的目录下
- 当浏览器发出文件请求时, 服务器端就会到这个目录下去寻找相关文件

```
app.use(express.static(require('path').join(__dirname, 'public')), {options});
```

# post方法

根据请求路径来处理客户端发出的Post请求

```
var bodyParser = require('body-parser');  
app.use(bodyParser.urlencoded({extended:true}));  
app.post(path,function(req, res));
```

`req.body` 属性解析客户端的 **post** 请求参数，通过它可获取请求路径的参数值。