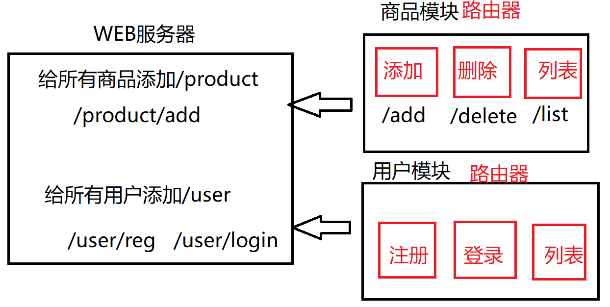
练习：添加路由(get /login)，响应login.html，点击提交向服务器发请求(post /mylogin)，响应'登录成功'



|  |  |  |
| --- | --- | --- |
| 传参方式 | 格式 | 路由获取 |
| post传递 | URL中不可见 | req.on('data',(c)=>{  c 获取的数据，格式为buffer，需要转字符串，转后为查询字符串，还需要进一步转为对象  }) |

一.路由器

路由器用来管理路由，将同一个模块下的路由统一管理；可以给路由器下所有路由添加同一个的前缀，防止URL相同



|  |
| --- |
| 路由器  //1.创建路由器对象  const r=express.Router();  //2.添加路由  //3.暴露路由器对象  module.exports=r; |
| WEB服务器  //1.引入路由器  //2.使用路由器，将所有路由器下的路由挂在到WEB服务器下，同时添加前缀  app.use( 添加的前缀，引入的路由器 ) |

练习：创建用户模块路由器user.js，创建路由器对象，添加用户列表路由(get /list)，暴露路由器对象；在app.js在引入路由器，使用路由，添加前缀/user

二.中间件

用于拦截对服务器的请求

中间件分为应用级中间件、路由级中间件、内置中间件、第三方中间件、错误处理中间件

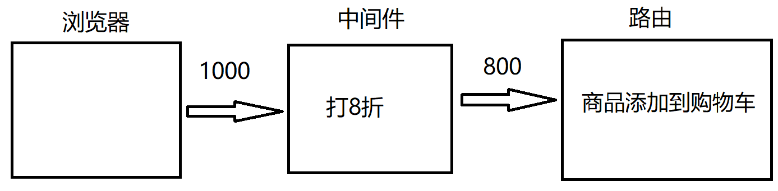
1.应用级中间件

是一个函数，一旦拦截到自动调用这个函数

函数(req,res,next) //next() 往后执行，可能是下一个中间件或者路由

app.use( 拦截的URL, 函数 )

练习：添加路由(get /shopping)，用于将商品加入到购物车，传递商品的价格price，添加中间件实现对价格打8折；最后在路由中响应 '商品价格：xx'



2.路由级中间件

就是路由器的使用

当拦截到以后到指定的路由器下寻找路由

3.内置中间件

(1)托管静态资源

客户端请求静态资源(html、css、js、图像...)，不需要通过路由去响应，而是自动的到指定的目录查找。

app.use( express.static('要托管的目录') )

练习：编写文件05\_static.js，创建WEB服务器，托管静态资源到public目录，目录中创建文件login.html，点击提交向服务器发请求(post /mylogin)，响应'登录成功'



(2)将post传递参数转为对象

|  |
| --- |
| app.use( express.urlencoded({  extended:false //内部如何转为对象  }) ) |
| req.body //路由中获取对象 |

复习数据库，xampp查看是否能用；把01\_tedu.sql提交到数据库服务器，MySQL使用命令

练习：创建WEB服务器，托管静态资源到public目录，包含文件add.html，点击提交向服务器发请求(post /myadd)，响应'员工添加成功，姓名：xx'

