1.中间件

用户处理（过滤）请求，最终要为路由提供服务

应用级中间件（自定义） 路由级中间件 内置中间件 第三方中间件

（1）应用级中间件

App.use(url,function(req,res,next))

url表示中间件要给哪一个路由的url使用，对应路由的url

在中间件中间件中可以拦截到请求，也可以作出响应；

Next是一个函数，表示执行下一个中间件或者路由

练习：创建路由，方法：get url:/shopping;传递商品的价格price，要求在中间件中将商品的价格加100

‘商品价格为：\*\*\*’

（2）路由级中间件

路由器的使用

（3）内置中间件

Express当前只有一个内置中间件

把所有的静态资源托管到指定的目录，如果浏览器请求文件，自动到该目录下寻找，不需要创建路由来响应文件。

app.use(express.static(‘目录’));

练习：托管静态资源到files目录，如果files目录和public目录有相同名称的文件，查看会显示哪一个

练习：创建web服务器，托管静态资源到plublic目录，创建文件login.html;点击提交，向服务器发送post请求，请求方法post,请求的url:/mylogin,在路由中获取post请求的数据

（4）第三方中间件body-parser使用

|  |
| --- |
| //1.引入body-parser中间件模块  const bodyParser=require('body-parser');  //2.使用body-parser中间件，将post请求的数据解析为对象  app.use(bodyParser.urlencoded({  extended:false //使用querystring模块(原始模块)，而不是用第三方的qs模块  }）；  //3.在路由中获取post请求数据  req.body |

2.mysql模块

Mysql使用

INSERT INTO emp VALUES(..);

DELEFT FEOM

|  |
| --- |
| //创建连接对象  Var connection=mysql.createConnection({})  //执行连接  Connection.Connect()  //执行SQL语句  Connection.query(sql,callback)  Sql要执行的SQL语句  Callback 获取执行的结果  Err 可能产生的错误  Result SQL语句的结果  //关闭连接  Connection.end() |

查询：数组

插入：对象 affectedRows大于0说明插入成功了

修改：对象 affectedRows大于0说明修改成功了

删除：对象 affectedRows大于0说明删除成功了

练习：往员工表插入一条数据

使用占位符（?），对数据进行过滤

|  |
| --- |
| 使用连接池  Var pool=mysql.createpool({})  //connectionLimit 用于设置连接池大小  Pool.query(sql,callback) //执行SQL语句 |

练习：创建web服务器，托管静态资源到public目录，在public目录下创建user\_reg.html

课后任务

（1）复习今天内容，整理思维导图

（2）课后练习

创建web服务器，托管静态资源到public目录，创建文件add.html,点击提交，向服务器请求（/add get）,获取数据，把数据插入到tedu下的dept表中，响应‘部门添加成功’

