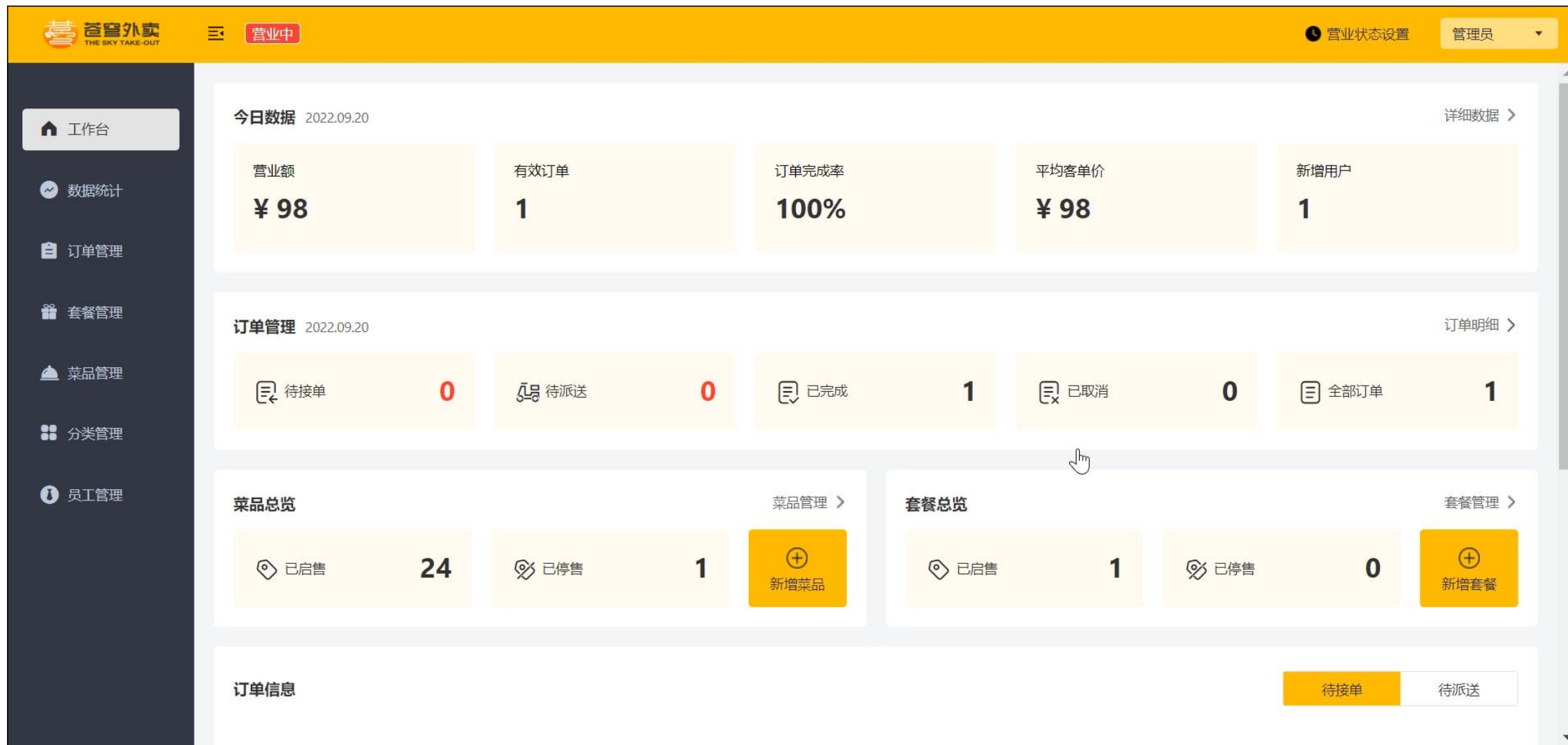


数据统计—Excel报表







目录

Contents

- ◆ 工作台
- ◆ Apache POI
- ◆ 导出运营数据Excel报表



工作台

- 需求分析和设计
- 代码导入
- 功能测试

需求分析和设计

工作台是系统运营的数据看板，并提供快捷操作入口，可以有效提高商家的工作效率。

工作台展示的数据：

- 今日数据
- 订单管理
- 菜品总览
- 套餐总览
- 订单信息



需求分析和设计

名词解释:

- 营业额: 已完成订单的总金额
- 有效订单: 已完成订单的数量
- 订单完成率: $\text{有效订单数} / \text{总订单数} * 100\%$
- 平均客单价: $\text{营业额} / \text{有效订单数}$
- 新增用户: 新增用户的数量

需求分析和设计

接口设计:

- 今日数据接口
- 订单管理接口
- 菜品总览接口
- 套餐总览接口
- 订单搜索（已完成）
- 各个状态的订单数量统计（已完成）

苍穹外卖LOGO

营业中

营业状态 99+ 管理员

今日数据 (2022年4月22日) 4 查看详细数据 >

营业额 ¥ 200.13	有效订单 12	订单完成率 16	平均客单价 30	新增用户 18
-----------------	------------	-------------	-------------	------------

工作台

订单管理

分类管理

菜品管理

套餐管理

数据统计

员工管理

订单管理 5 查看订单明细 >

12 待接单	10 待派送	18 已完成	1 已取消	41 全部订单
-----------	-----------	-----------	----------	------------

菜品/套餐总览 6 查看菜品管理 >

12 已启售	2 已停售	+ 新增菜品
12 已启售	2 已停售	+ 新增套餐

查看套餐管理 >

订单信息 7

待接单 (12) 待派送 (10)

订单号	订单菜品	地址	预计送达时间	实收金额	备注	操作
2021010200001	宫保鸡丁* 3; 红烧带鱼* 2; 农家小炒肉* 1;	金燕龙办公楼 (建材城西路9号) 四层 (一一宾馆北侧办公楼)	2021-01-02 11:11:11	40.00	不要香菜	接单 拒单 查看
2021010200001	宫保鸡丁* 3; 红烧带鱼* 2; 农家小炒肉* 1;	金燕龙办公楼 (建材城西路9号) 四层 (一一宾馆北侧办公楼)	2021-01-02 11:11:11	40.00	不要香菜	接单 拒单 查看
2021010200001	宫保鸡丁* 3; 红烧带鱼* 2; 农家小炒肉* 1;	金燕龙办公楼 (建材城西路9号) 四层 (一一宾馆北侧办公楼)	2021-01-02 11:11:11	40.00		接单 拒单 查看
2021010200001	宫保鸡丁* 3; 红烧带鱼* 2; 农家小炒肉* 1;	金燕龙办公楼 (建材城西路9号) 四层 (一一宾馆北侧办公楼)	2021-01-02 11:11:11	40.00	不要香菜	接单 拒单 查看
2021010200001	宫保鸡丁* 3; 红烧带鱼* 2; 农家小炒肉* 1;	金燕龙办公楼 (建材城西路9号) 四层 (一一宾馆北侧办公楼)	2021-01-02 11:11:11	40.00	不要香菜	接单 拒单 查看

需求分析和设计

今日数据的接口设计：

基本信息

Path: /admin/workspace/businessData

Method: GET

接口描述:

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
├ newUsers	integer	必须		新增用户数	format: int32
├ orderCompletionRate	number	必须		订单完成率	format: double
├ turnover	number	必须		营业额	format: double
├ unitPrice	number	必须		平均客单价	format: double
├ validOrderCount	integer	必须		有效订单数	format: int32
msg	string	非必须			

需求分析和设计

订单管理的接口设计：

基本信息

Path: /admin/workspace/overviewOrders

Method: GET

接口描述:

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
└─ allOrders	integer	必须		全部订单	format: int32
└─ cancelledOrders	integer	必须		已取消数量	format: int32
└─ completedOrders	integer	必须		已完成数量	format: int32
└─ deliveredOrders	integer	必须		待派送数量	format: int32
└─ waitingOrders	integer	必须		待接单数量	format: int32
msg	string	非必须			

需求分析和设计

菜品总览的接口设计：

基本信息

Path: /admin/workspace/overviewDishes

Method: GET

接口描述：

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
└ discontinued	integer	必须		已停售菜品数量	format: int32
└ sold	integer	必须		已启售菜品数量	format: int32
msg	string	非必须			

需求分析和设计

套餐总览的接口设计：

基本信息

Path: /admin/workspace/overviewSetmeals

Method: GET

接口描述：

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
└ discontinued	integer	必须		已停售套餐数量	format: int32
└ sold	integer	必须		已启售套餐数量	format: int32
msg	string	非必须			










工作台

- 需求分析和设计
- 代码导入
- 功能测试

代码导入

直接导入课程资料中的工作台模块功能代码即可：

-  DishMapper.java
-  DishMapper.xml
-  SetmealMapper.java
-  SetmealMapper.xml
-  WorkspaceController.java
-  WorkspaceService.java
-  WorkspaceServiceImpl.java



工作台

- 需求分析和设计
- 代码导入
- 功能测试

功能测试

可以通过如下方式进行测试：

- 通过接口文档测试
- 前后端联调测试



目录

Contents

- ◆ 工作台
- ◆ Apache POI
- ◆ 导出运营数据Excel报表



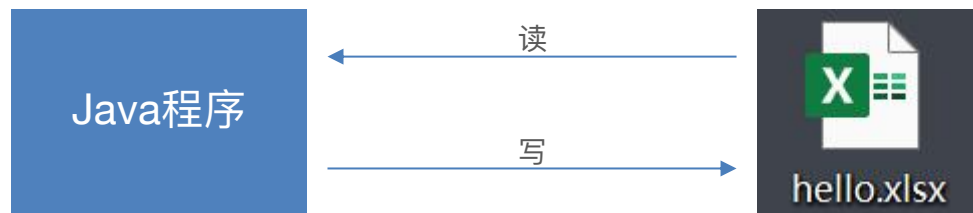
Apache POI

- 介绍
- 入门案例

介绍

Apache POI 是一个处理Microsoft Office各种文件格式的开源项目。简单来说就是，我们可以使用 POI 在 Java 程序中对Microsoft Office各种文件进行读写操作。

一般情况下，POI 都是用于操作 Excel 文件。



为什么要在Java程序中操作Excel文件呢?

介绍

Apache POI 的应用场景：

- 银行网银系统导出交易明细
- 各种业务系统导出Excel报表
- 批量导入业务数据

信息统计表							
序号	姓名	性别	民族	身份证号	年龄	联系方式	备注

介绍

例子程序运行效果展示：

D:\\itcast.xlsx

	A	B	C	D
1		姓名	爱好	
2		张三	篮球	
3		李四	足球	
4		王五	羽毛球	

```
11 public class POITest {
12     /**
13      * 基于POI读取Excel文件
14      * @throws Exception
15      */
16     public static void main(String[] args) throws Exception {
17         FileInputStream in = new FileInputStream(new File( pathname: "D:\\itcast.xlsx"));
18         //通过输入流读取指定的Excel文件
19         XSSFWorkbook excel = new XSSFWorkbook(in);
20         //获取Excel文件的第1个Sheet页
21         XSSFSheet sheet = excel.getSheetAt( index: 0);
22
23         //获取Sheet页中的最后一行的行号
24         int lastRowNum = sheet.getLastRowNum();
25
26         for (int i = 0; i <= lastRowNum; i++) {
27             //获取Sheet页中的行

```

Run: POITest x

姓名 爱好
张三 篮球
李四 足球
王五 羽毛球

Process finished with exit code 0

Git Run TODO Problems Statistic Terminal Profiler Endpoints Build Spring

Build completed successfully in 5 sec, 569 ms (2 minutes ago)



Apache POI

- 介绍
- 入门案例

入门案例

Apache POI的maven坐标：

```
<dependency>
  <groupId>org.apache.poi</groupId>
  <artifactId>poi</artifactId>
  <version>3.16</version>
</dependency>
<dependency>
  <groupId>org.apache.poi</groupId>
  <artifactId>poi-ooxml</artifactId>
  <version>3.16</version>
</dependency>
```

入门案例

将数据写入Excel文件：

```
//在内存中创建一个Excel文件对象
XSSFWorkbook excel = new XSSFWorkbook();
//创建Sheet页
XSSFSheet sheet = excel.createSheet("itcast");

//在Sheet页中创建行，0表示第1行
XSSFRow row1 = sheet.createRow(0);
//创建单元格并在单元格中设置值，单元格编号也是从0开始，1表示第2个单元格
row1.createCell(1).setCellValue("姓名");
row1.createCell(2).setCellValue("城市");

XSSFRow row2 = sheet.createRow(1);
row2.createCell(1).setCellValue("张三");
row2.createCell(2).setCellValue("北京");

XSSFRow row3 = sheet.createRow(2);
row3.createCell(1).setCellValue("李四");
row3.createCell(2).setCellValue("上海");

FileOutputStream out = new FileOutputStream(new File("D:\\itcast.xlsx"));
//通过输出流将内存中的Excel文件写入到磁盘上
excel.write(out);

//关闭资源
out.flush();
out.close();
excel.close();
```


入门案例

读取Excel文件中的数据：

```
FileInputStream in = new FileInputStream(new File("D:\\itcast.xlsx"));
//通过输入流读取指定的Excel文件
XSSFWorkbook excel = new XSSFWorkbook(in);
//获取Excel文件的第1个Sheet页
XSSFSheet sheet = excel.getSheetAt(0);

//获取Sheet页中的最后一行的行号
int lastRowNum = sheet.getLastRowNum();

for (int i = 0; i <= lastRowNum; i++) {
    //获取Sheet页中的行
    XSSFRow titleRow = sheet.getRow(i);
    //获取行的第2个单元格
    XSSFCell cell1 = titleRow.getCell(1);
    //获取单元格中的文本内容
    String cellValue1 = cell1.getStringCellValue();
    //获取行的第3个单元格
    XSSFCell cell2 = titleRow.getCell(2);
    //获取单元格中的文本内容
    String cellValue2 = cell2.getStringCellValue();
    System.out.println(cellValue1 + " " + cellValue2);
}

//关闭资源
in.close();
excel.close();
```



目录

Contents

- ◆ 工作台
- ◆ Apache POI
- ◆ 导出运营数据Excel报表



导出运营数据Excel报表

- 需求分析和设计
- 代码开发
- 功能测试

需求分析和设计

产品原型:



需求分析和设计

导出的Excel报表格式：

运营数据报表					
概览数据					
营业额		订单完成率		新增用户数	
有效订单		平均客单价			
明细数据					
日期	营业额	有效订单	订单完成率	平均客单价	新增用户数

业务规则：

- 导出Excel形式的报表文件
- 导出最近30天的运营数据

需求分析和设计

接口设计：

基本信息

Path: /admin/report/export

Method: GET

接口描述：

请求参数

返回数据

注意：当前接口没有返回数据，因为报表导出功能本质上是文件下载，

服务端会通过输出流将Excel文件下载到客户端浏览器



导出运营数据Excel报表

- 需求分析和设计
- 代码开发
- 功能测试

代码开发

实现步骤:

- ① 设计Excel模板文件
- ② 查询近30天的运营数据
- ③ 将查询到的运营数据写入模板文件
- ④ 通过输出流将Excel文件下载到客户端浏览器

运营数据报表					
时间: 2022-05-28至2022-06-26					
概览数据					
营业额	¥2,506.00	订单完成率	76.00%	新增用户数	3
有效订单	19	平均客单价	¥131.89		
明细数据					
日期	营业额	有效订单	订单完成率	平均客单价	新增用户数
2022-05-28	¥0.00	0	0.00%	¥0.00	0
2022-05-29	¥0.00	0	0.00%	¥0.00	0
2022-05-30	¥0.00	0	0.00%	¥0.00	0
2022-05-31	¥0.00	0	0.00%	¥0.00	0
2022-06-01	¥0.00	0	0.00%	¥0.00	1
2022-06-02	¥0.00	0	0.00%	¥0.00	1
2022-06-03	¥0.00	0	0.00%	¥0.00	0
2022-06-04	¥0.00	0	0.00%	¥0.00	0

下载



代码开发

根据接口定义，在ReportController中创建export方法：

```
/**
 * 导出运营数据报表
 * @param response
 */
@GetMapping("/export")
@ApiOperation("导出运营数据报表")
public void export(HttpServletResponse response){
    reportService.exportBusinessData(response);
}
```

代码开发

在ReportService接口中声明导出运营数据报表的方法：

```
/**  
 * 导出近30天的运营数据报表  
 * @param response  
 */  
void exportBusinessData(HttpServletResponse response);
```

代码开发

在ReportServiceImpl实现类中实现导出运营数据报表的方法（第1部分）：

```
/**
 * 导出近30天的运营数据报表
 * @param response
 */
public void exportBusinessData(HttpServletResponse response) {
    LocalDate begin = LocalDate.now().minusDays(30);
    LocalDate end = LocalDate.now().minusDays(1);
    //查询概览运营数据，提供给Excel模板文件
    BusinessDataVO businessData = workspaceService.getBusinessData(LocalDateTime.of(begin, LocalTime.MIN), LocalDateTime.of(end, LocalTime.MAX));

    InputStream inputStream = this.getClass().getClassLoader().getResourceAsStream("template/运营数据报表模板.xlsx");
    try {
        //基于提供好的模板文件创建一个新的Excel表格对象
        XSSFWorkbook excel = new XSSFWorkbook(inputStream);

        //获得Excel文件中的一个Sheet页
        XSSFSheet sheet = excel.getSheet("Sheet1");
```

代码开发

在ReportServiceImpl实现类中实现导出运营数据报表的方法（第2部分）：

```
sheet.getRow(1).getCell(1).setCellValue(begin + "至" + end);

//获得第4行
XSSFRow row = sheet.getRow(3);
//获取单元格
row.getCell(2).setCellValue(businessData.getTurnover());
row.getCell(4).setCellValue(businessData.getOrderCompletionRate());
row.getCell(6).setCellValue(businessData.getNewUsers());

row = sheet.getRow(4);
row.getCell(2).setCellValue(businessData.getValidOrderCount());
row.getCell(4).setCellValue(businessData.getUnitPrice());

for (int i = 0; i < 30; i++) {
    LocalDate date = begin.plusDays(i);
    //准备明细数据
    businessData = workspaceService.getBusinessData(LocalDateTime.of(date, LocalTime.MIN), LocalDateTime.of(date, LocalTime.MAX));
    row = sheet.getRow(7 + i);
```

代码开发

在ReportServiceImpl实现类中实现导出运营数据报表的方法（第3部分）：

```
row.getCell(1).setCellValue(date.toString());
row.getCell(2).setCellValue(businessData.getTurnover());
row.getCell(3).setCellValue(businessData.getValidOrderCount());
row.getCell(4).setCellValue(businessData.getOrderCompletionRate());
row.getCell(5).setCellValue(businessData.getUnitPrice());
row.getCell(6).setCellValue(businessData.getNewUsers());
}

//通过输出流将文件下载到客户端浏览器中
ServletOutputStream out = response.getOutputStream();
excel.write(out);

//关闭资源
out.flush();
out.close();
excel.close();
} catch (IOException e) {
    e.printStackTrace();
}
```



导出运营数据Excel报表

- 需求分析和设计
- 代码开发
- 功能测试

功能测试

可以通过如下方式进行测试：

- 前后端联调测试



传智教育旗下高端IT教育品牌