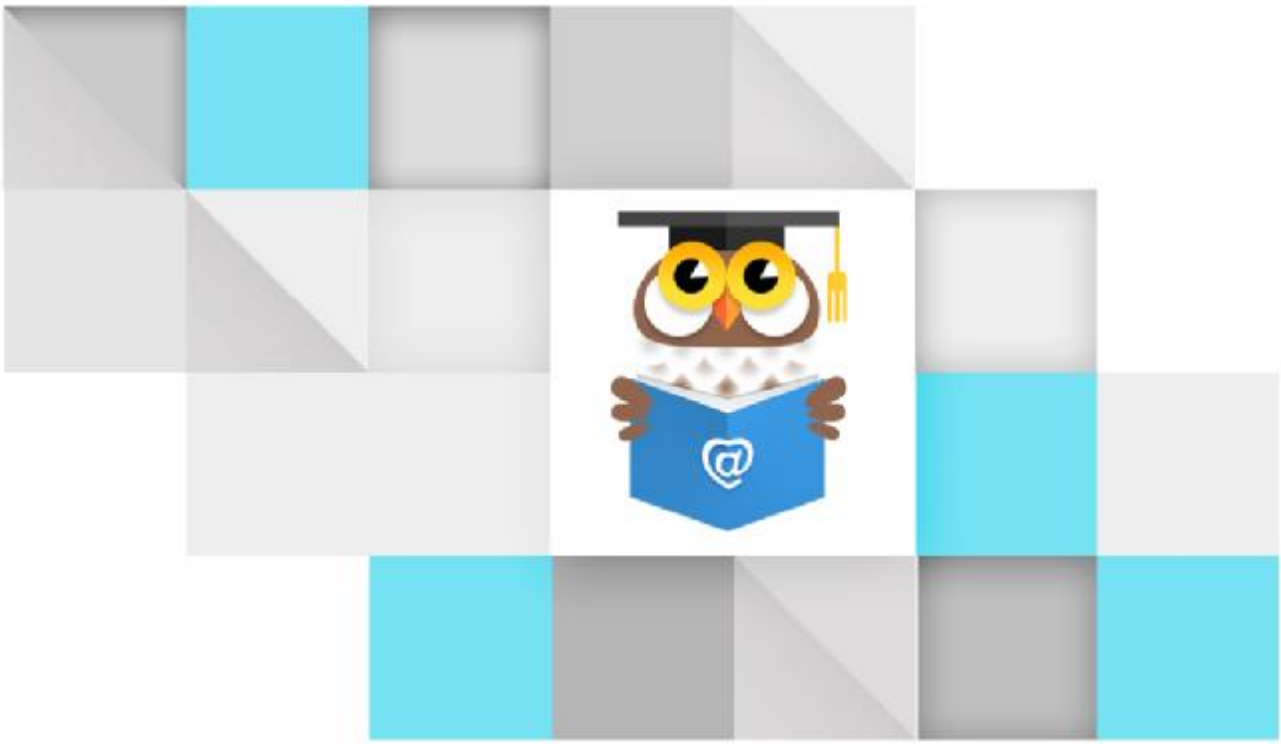
消灭泡泡糖（Java）

实训指导手册

实训场景 011 – 移动垂直方向的泡泡糖（二）



Campus Solution Group

目 录

一、任务编号：PRJ-BU2-JAVA-011

1、实训技能

- Java API 运用技能

2、涉及知识点

- List 实现类的特点
- List 中的方法

3、实现效果

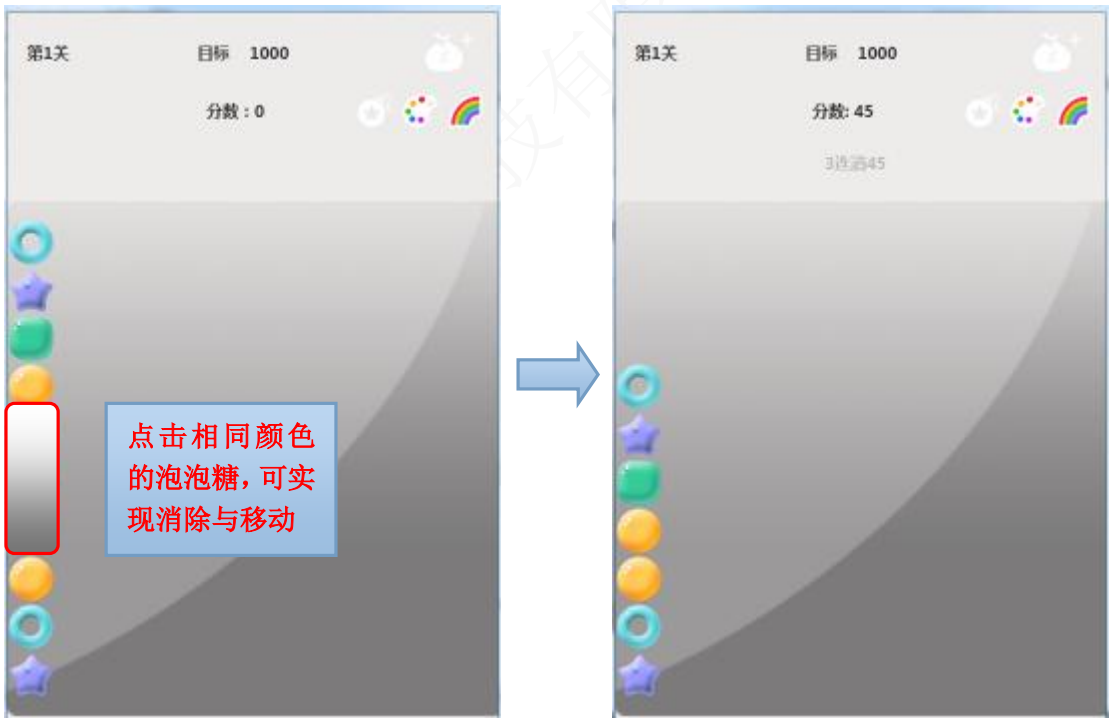


图 3-1

4、场景说明

1、业务说明：

场景**PRJ-BU2-JAVA-010**实现了单列泡泡糖消除与垂直移动“填补空隙”的功能。

场景**PRJ-BU2-JAVA-010**功能实现过程中使用了**StarList**类中的**lookup**和**existed**函数，本场景将需要对以上两个函数进行实现。

我们将把**lookup**和**existed**函数的实现存放于**StarList**，由于这两个函数均需操作**List**类型的集合，而**List**类型的集合并未提供相关函数，因此我们选择使用**StarList**继承（扩展）**ArrayList**，以此体现出组件继承与功能函数扩展的意义。

2、实现思路：

从场景**PRJ-BU2-JAVA-010**可知，“待消除泡泡糖”如选择**List**作为数据存储容器，则**List**的原始功能并不能满足我们的业务需求，因为我们还需要集合拥有**2**个功能：

2-1. 通过行、列值，来定位获取集合中的泡泡糖元素。

2-2. 通过行、列值判断泡泡糖对象是否存在于集合之中。

因此，我们通过**StarList**继承**ArrayList**的方式去承接集合原有的功能，然后在**StarList**中编写需要扩展的功能函数：**lookup**和**exist**来满足**2-1**和**2-2**的需求。

3、核心组件介绍：

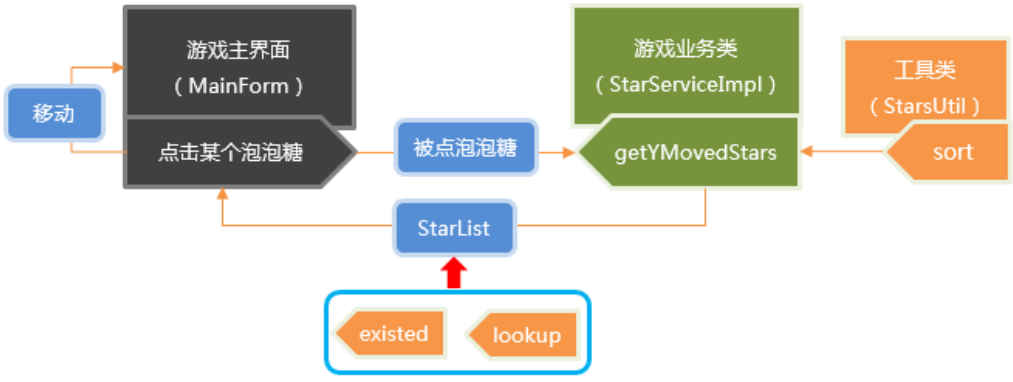


图 **4-1**

3-1. StarServiceImpl - 游戏业务类

负责游戏相关逻辑计算，例如：泡泡糖移动、消除、分数计算等操作。

3-2. StarList - 待移动泡泡糖（**MovedStar**）集合。

3-3. StarList类中的lookup方法：

该方法主要通过行、列值来定位获取集合中的泡泡糖元素。

从界面泡泡糖集合中获取某个泡泡糖需要使用该方法。

3-4. StarList类中的existed方法：

该方法主要通过行、列值判断泡泡糖对象是否存在于集合之中；

在获取“待移动泡泡糖”时，需要通过此方法判断普通泡泡糖的移动步长是否需要**+1**。

4、了解更多：

请参考《消灭泡泡糖 **-** 需求说明文档》

5、前置条件：

5-1. 前置场景：**PRJ-BU2-JAVA-010** – 移动垂直方向的泡泡糖（一）

5-2. 必备知识与技能：

5-2.1. Java开发工具 （**Eclipse**）。

5-2.2. Java面向对象编程技能（类的继承语法、**super**、方法重载）。

5、快速开始

1、 开发环境：

1-1. Oracle JDK8.x 以上版本

1-2. Eclipse Luna（**4.4.x**）以上版本

1-3. 工程包：**PRJ_BU2_JAVA_011**

2、 进入开发环境：

详见**SPOC**平台上《**PRJ-BU2-JAVA-011** 前置任务：进入开发环境》

图 **5-1**

6、任务 1 – 根据坐标查找泡泡糖

1、任务描述：

本任务要求对**ArrayList**进行扩展，实现根据行、列值查找泡泡糖的方法。

由于**List**类型的集合原本功能不能满足业务需求，因此这里我们需要使用自定义类**StarList**继承**List**类型的集合，然后在**StarList**中增加功能函数，从而实现对集合原始功能的扩展。

2、实现思路：

2-1. 场景定位

2-1.1. 定位到：**cn.campsg.practical.bubble.entity.StarList**类

2-2. 扩展**ArrayList**类的功能

2-2.1. 让**StarList**类继承**ArrayList<Star>**类

2-3. 创建【通过行、列值查询泡泡糖对象的】公有函数：**lookup**

2-3.1. 方法参数：目标行（整数）、目标列（整数）

2-3.2. 返回值：获取的泡泡糖对象

2-4. 循环遍历整个集合，每次循环中需做如下操作：

2-4.1. 判断从集合中获取的泡泡糖对象是否为**null**，如为**null**则直接进行下一次循环。

2-4.2. 如泡泡糖对象的行列值与函数入参的行列值相等，则表示已经找到对应的泡泡糖，函数需要直接返回该泡泡糖对象，而不再进行循环判断。

2-5. 循环结束后仍未找到满足行列值的泡泡糖，表明集合中无符合条件的泡泡糖，返回**null**。

+ 提示：

1） 获取集合的长度可以直接使用**size**函数，由于**ArrayList**是**StarList**的父类，因此可通过**super**关键字调用**size**获取集合长度。

2） 同上，获取集合对象，可以使用**super**的**get**函数。

3） 在循环中如想直接终止当前函数可以直接使用**return**而非**break**。

3、验证与测试：

3-1. 在**StarList**类中创建程序入口函数**main**

3-2. 编写以下测试语句：

3-2.1. 创建**StarList**对象

3-2.2. 向**StarList**中添加，从（**0,0**）到（**9,0**）**10**个**Star**对象，泡泡糖类型可任意选择

3-3. 打印测试结果：

3-3.1. 通过**StarList**对象的**lookup**函数，打印出（**3,0**）位置的**Star**对象【该对象存在】

3-3.2. 通过**StarList**对象的**lookup**函数，打印出（**1,1**）位置的**Star**对象【该对象不存在】

3-3.3. 输出结果与下图一致：

```
(3,0)位置的泡泡糖：(3,0-BLUE)
(1,1)位置的泡泡糖：null
```

图 **6-1**

7、任务 2 – 根据位置查找泡泡糖

1、任务描述：

利用**Java**函数的重载机制，实现根据位置【**Position**】查找泡泡糖的方法。

2、实现思路：

2-1. 场景定位

2-1.1. 定位到：**cn.campsg.practical.bubble.entity.StarList**类

2-2. 创建【通过**Position**查询泡泡糖对象】的公有函数**lookup** （重载任务**1**的**lookup**）

2-2.1. 方法参数：**Position**对象

2-2.2. 返回值：获取的泡泡糖对象

2-3. 获取入参**Position**的行值与列值。

2-4. 将**2-3**步骤获取的数据传入任务**1**实现的**lookup**方法，以此获取**Star**对象。

2-5. 返回**Star**对象

3、验证与测试：

3-1. 定位到任务**1**中实现的**main**函数

3-2. 对**main**函数作以下调整：

3-2.1. 将打印输出函数中所使用的**lookup**函数的参数，从原本的行列值修改为通过行列值生成的**Position**对象

3-2.2. 输出结果与下图一致：

```
(3,0)位置的泡泡糖：(3,0-BLUE)
(1,1)位置的泡泡糖：null
```

图 **7-1**

8、任务 3 – 判断泡泡糖是否存在

1、任务描述：

利用任务**2**的**lookup**函数，实现判断泡泡糖是否存在于集合的函数**existed**

2、实现思路：

2-1. 场景定位

2-1.1. 定位到：**cn.campsg.practical.bubble.entity.StarList**类

2-2. 创建公有函数 **existed**

2-2.1. 方法参数：**Star**对象

2-2.2. 返回类型：**boolean**

2-3. 如果入参**Star**等于**null**，则无需进行任何业务判断，直接返回【假】

2-4. 通过**Star**对象的**Position**属性，从当前集合中寻找泡泡糖对象。

2-5. 如果集合返回的泡泡糖对象为**null**，则表示不存在返回【假】；否则表示存在返回【真】

> **+ 提示：**
>
> 步骤【**2-5**】推荐使用【**A ? B : C**】三目运算符实现。

3、验证与测试：

3-1. 运行项目工程，选择**cn.campsg.practical.bubble.MainClass**类作为程序入口

3-2. 程序会显示一列泡泡糖，其中可消除泡泡糖的个数和位置均为随机生成

3-3. 点击同色泡泡糖可实现消除与移动功能

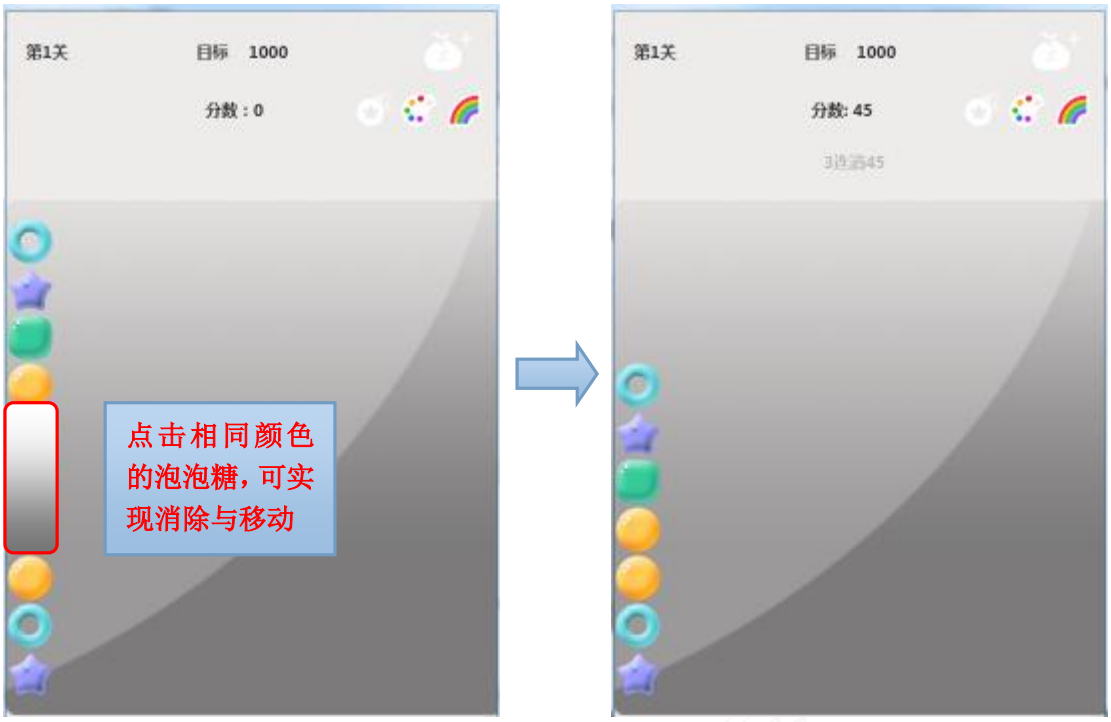


图 8-1

9、场景总结

Q1. 您能否重现项目中 Java 重载函数的运用场景？

1. 函数的入参在函数定义时往往是固定的，然而当不同业务均对该函数有调用需求时，我们往往需要为该函数传入不同类型或数量的参数，最终希望得到相同的业务计算需求，此时由于函数的功能没有发生变化，为满足不同调用需求定义两个函数是没有意义的，届时就可以考虑重载函数。

2. 例如：《消灭泡泡糖》游戏中，**lookup**“泡泡糖”搜索函数，既可以按“泡泡糖”位置对象搜索，也可以按“泡泡糖”的行列坐标搜索。

注意：名称相同，参数类型不同或参数数量不同的函数被称为重载函数（无需关注返回值）。

【说明】：合理利用重载函数可以大大减少重复代码，降低后期系统维护成本。

【说明】：重载属于“面向对象三要素”中的“多态性”。

重要提示：初学者切记，您编写的功能函数不仅仅为您一人服务，只有开发出”服务于他人“的函数才能不断进阶个人编程能力。

Q2. 何时定义私有函数？何时定义公有函数？

1. 公有函数往往是软件系统一个重要的功能（函数）。
2. 私有函数往往是软件系统重要功能的执行步骤。
3. 因此公有函数负责调用私有函数。
4. 独立调用功能函数的步骤是毫无意义的，应该通过 **private** 关键禁止。