

# 第二阶段项目--宠物小精灵

## 需求分析

冒险家携带宠物小精灵闯关，每一关卡都有对应的地图，地图随机生成，上面包含有宝箱、怪物和传送门，比例为 39:39:1，宝箱可以开出药品、装备或宠物小精灵，比例为 6 : 3 : 1。冒险家可以在地图上移动，每移动一步就会遇到宝箱、怪物或传送门。如果遇到宝箱，可以选择打开与否。如果遇到怪物，则需要选择携带的小精灵与之对抗，成功击杀怪物后会随机掉落装备、药品和宠物小精灵，掉落比例为 7 : 2 : 1。如果遇到传送门，则可以选择是否向传送门移动。如果向传送门移动，可以传送至对应的关卡。

### 说明

**药品：**可以用来恢复血量，每一关卡的药品不同，恢复的血量也不同，药品可以堆叠

**装备：**可以给携带的宠物小精灵穿戴，增加宠物小精灵的攻击、防御、血量

- 头盔：可以增加防御、血量
- 铠甲：可以增加防御、血量
- 护腿：可以增加防御、血量
- 鞋子：可以增加防御、血量
- 戒指：可以增加攻击、防御、血量
- 项链：可以增加攻击、防御、血量
- 手镯：可以增加攻击、防御、血量
- 武器：可以增加攻击、血量

所有装备的属性均根据关卡来随机生成，可能出现极品装备，也可能出现垃圾装备

**宠物小精灵：**可以穿戴装备，最多只能穿戴8件装备，每个部位只能穿戴一件

- 妙蛙种子：初级宠物小精灵
- 雷精灵：中级宠物小精灵
- 小火龙：高级宠物小精灵
- 比卡丘：究级宠物小精灵

所有同类宠物小精灵初始属性（攻击、防御、血量）相同，同类宠物小精灵可以进行融合升星，以提升宠物小精灵的初始属性。每次融合能提升初始属性的50%，同类宠物小精灵最多融合10次。

**怪物：**可以阻挡冒险家探索地图，冒险家需要使用携带的宠物小精灵与之对抗，将其击杀后方可继续探索地图。怪物被击杀后会掉落装备、药品或者宠物小精灵。

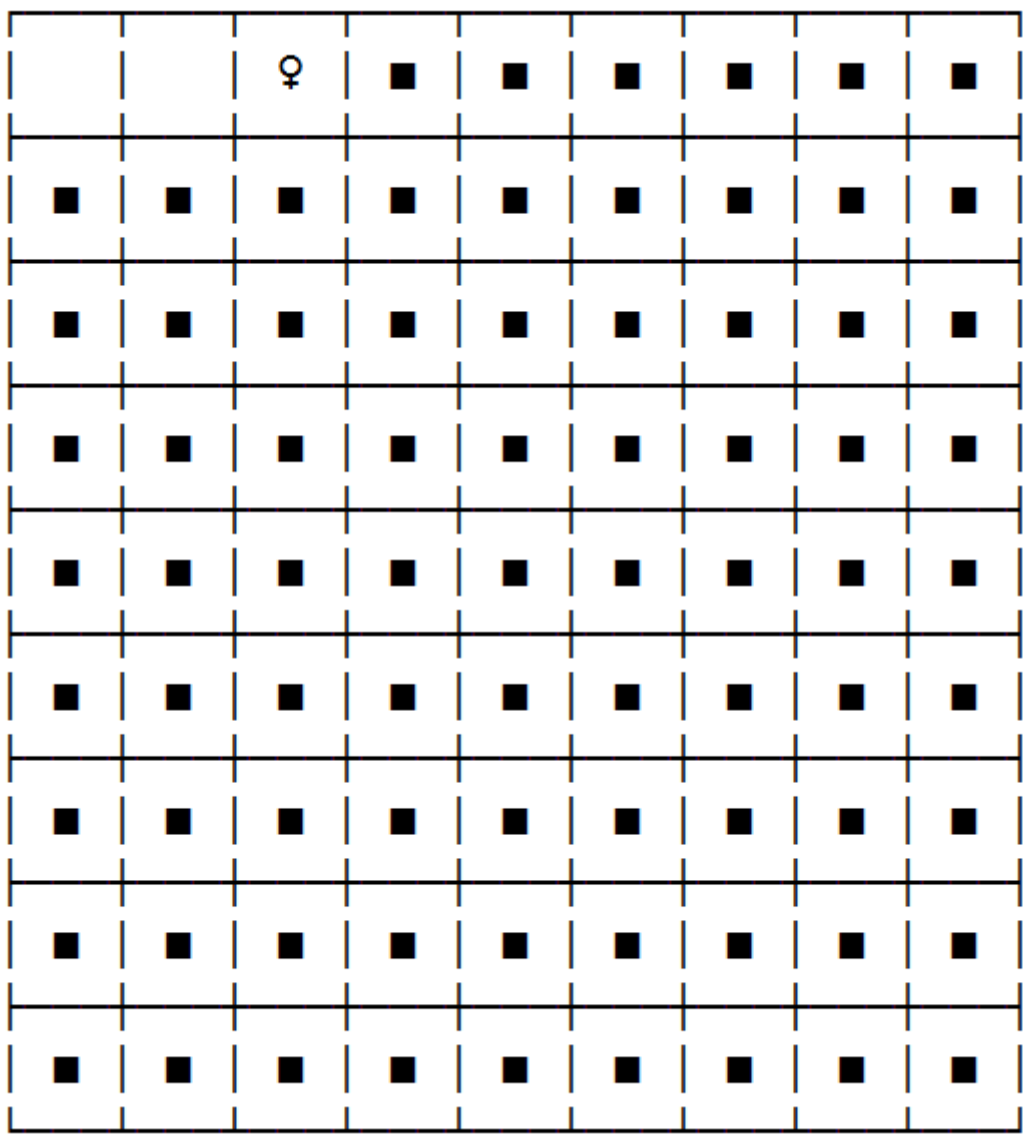
- 象牙猪：初级怪物
- 牛魔怪：中级怪物
- 铁炮鱼：高级怪物
- 火焰鸟：究级怪物

所有怪物的初始属性均根据关卡来随机生成

**传送门：**可以返回上一关卡，也可以前往下一关卡。

**地图：**地图大小为 9x9,共81个格子，第一关卡的地图上只有1个传送门，通往下一关卡；其余关卡均有两个传送门，返回上一关卡的传送门位于地图的第一个位置，前往下一关卡的传送门位置随机。冒险家闯关时，如果是第一关，则位于地图的第一个位置，第二个位置为初级怪物象牙猪；其余关卡，冒险家位于第二个位置。宝箱、怪物和传送门的比例为 39:39:1，地图为加密地图，冒险家不知道地图上到底

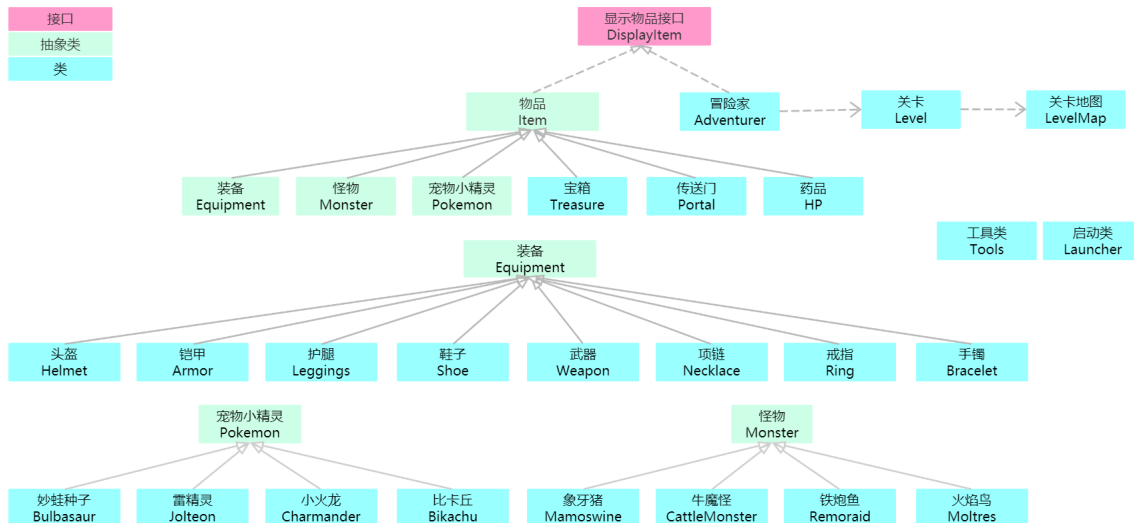
有什么。当冒险家探索某一位置地图时，该位置地图才具体显示。冒险家探索完成的地图显示为空白，以区分未探索的地图。如图：



冒险家探索完第二个位置后，第二个位置显示为空白

**冒险家：**冒险家携带背包闯关，背包中可以容纳装备、药品以及小精灵，默认有10个药品、1个妙蛙种子。冒险家闯关时可以使用 W(上)、A(左)、S(下)、D(右)、四个键在地图上移动，使用 E(退出)

类结构图展示



## 实现步骤

### 1. 接口设计

地图上物品信息会加密，被探索之后会显示出来，可以使用接口来完成约定

```

1  /**
2   * 物品显示接口
3   */
4  public interface DisplayItem {
5
6      /**
7       * 获取物品信息
8       * @return
9       */
10     String getItemInformation();
11
12 }
  
```

所有物品必须遵守这个约定

```

1  package com.cyx.pokemon.item;
2
3  import com.cyx.pokemon.DisplayItem;
4
5  /**
6   * 宝箱
7   */
8  public class Treasure implements DisplayItem {
9
10     @Override
11     public String getItemInformation() {
12         return "■";
13     }
14 }
15
16 package com.cyx.pokemon.item;
17
18 import com.cyx.pokemon.DisplayItem;
  
```

```

19
20 /**
21  * 怪物
22  */
23 public class Monster implements DisplayItem {
24
25     @Override
26     public String getItemInformation() {
27         return "■";
28     }
29 }
30
31 package com.cyx.pokemon.item;
32
33 import com.cyx.pokemon.DisplayItem;
34
35 /**
36  * 传送门
37  */
38 public class Portal implements DisplayItem {
39
40     @Override
41     public String getItemInformation() {
42         return "■";
43     }
44 }

```

## 2. 展示物品设计

冒险家探索地图时，物品信息会具体显示。换言之，冒险家探索到某物品时，该物品展示具体信息，探索完成后，该物品展示信息为空白。每一个物品都具有这样的特性，除此之外，物品大多数都与关卡有关，物品都有名称，因此可以利用继承的特性来实现。每一种物品展示的信息不一样，因此父类只能设计为抽象类。

```

1 package com.cyx.pokemon.item;
2
3 import com.cyx.pokemon.DisplayItem;
4
5 /**
6  * 物品
7  */
8 public abstract class Item implements DisplayItem {
9
10     /**
11      * 物品名称
12      */
13     protected String name;
14
15     /**
16      * 关卡编号
17      */
18     protected int levelNumber;
19
20     /**
21      * 是否被探索
22      */
23     protected boolean discovery;
24
25     public Item(String name) {

```

```

24         this.name = name;
25     }
26
27     public Item(String name, int levelNumber) {
28         this.name = name;
29         this.levelNumber = levelNumber;
30     }
31
32     public void setDiscovery(boolean discovery) {
33         this.discovery = discovery;
34     }
35 }
36
37 package com.cyx.pokemon.item;
38
39 import com.cyx.pokemon.DisplayItem;
40
41 /**
42  * 传送门
43  */
44 public class Portal extends Item {
45     /**
46      * 是否是通往下一关卡的传送门
47      */
48     private boolean next;
49
50     public Portal(boolean next) {
51         super("传送门");
52         this.next = next;
53     }
54
55     @Override
56     public String getItemInformation() {
57         if(discovery){
58             return next ? "→" : "←";
59         }
60         return "■";
61     }
62 }
63
64 package com.cyx.pokemon.item;
65
66 /**
67  * 怪物
68  */
69 public class Monster extends Item {
70
71     public Monster(String name, int levelNumber) {
72         super(name, levelNumber);
73     }
74
75     @Override
76     public String getItemInformation() {
77         return discovery ? name : "■";
78     }
79 }
80
81 package com.cyx.pokemon.item;

```

```

82
83  /**
84   * 宝箱
85   */
86  public class Treasure extends Item {
87
88      public Treasure(int levelNumber) {
89          super("宝箱", levelNumber);
90      }
91
92      @Override
93      public String getItemInformation() {
94          return discovery ? "🔍" : "■";
95      }
96  }

```

### 3. 传送门分析

传送门有两个方向，一个是返回上一关卡，一个是前往下一关卡，，可以使用布尔类型的变量进行控制  
见 => 展示物品设计

### 4. 宝箱分析

可以开出药品、装备或者宠物小精灵、比例为 6 : 3 : 1，因为药品、装备、小精灵还未设计，所以暂时先写简单的逻辑实现。

```

1  /**
2   * 开启宝箱能够获得一个物品
3   * @return
4   */
5  public Item open(){
6      Random r = new Random();
7      //取[0, 10)范围内的随机数
8      int number = r.nextInt(10);
9      if(number == 0){//获得宠物小精灵
10
11      } else if(number <= 3){//获得装备
12
13      } else { //获得药品
14
15      }
16      return null;
17  }

```

### 5. 药品分析

恢复血量与关卡有关，药品可以堆叠

```

1  package com.cyx.pokemon.item;
2
3  /**
4   * 药品：回复血量
5   */
6  public class HP extends Item{
7
8      private int count;

```

```

9
10     public HP(int levelNumber, int count) {
11         super("天山雪莲", levelNumber);
12         this.count = count;
13     }
14
15     /**
16      * 使用药品可以回复血量
17      * @return
18      */
19     public int use(){
20         count--;
21         return levelNumber * 500;
22     }
23
24     /**
25      * 检测药品是否可以被销毁
26      * @return
27      */
28     public boolean canDestroy(){
29         return count == 0;
30     }
31
32     @Override
33     public String getItemInformation() {
34         return name;
35     }
36 }

```

## 6. 装备分析

装备有8种，因此装备可以设计为抽象类。每一种装备具有攻击、防御、血量三种属性，属性值与关卡有关且属性随机。

```

1     package com.cyx.pokemon.util;
2
3     import java.util.Random;
4
5     /**
6      * 工具类
7      */
8     public class Tools {
9         /**
10          * 随机数对象
11          */
12         private static final Random RANDOM = new Random();
13
14         /**
15          * 获取给定范围内的随机数
16          * @param min 最小值
17          * @param max 最大值
18          * @param levelNumber 关卡编号
19          * @return
20          */
21         public static int getRandomNumber(int min, int max, int levelNumber){
22             int diff = (max - min) * levelNumber;
23             return RANDOM.nextInt(diff) + min * levelNumber;

```

```

24     }
25
26     /**
27      * 获取给定范围内的随机数
28      * @param min 最小值
29      * @param max 最大值
30      * @return
31      */
32     public static int getRandomNumber(int min, int max){
33         return getRandomNumber(min, max, 1);
34     }
35
36     /**
37      * 获取给定范围内的随机数
38      * @param max 最大值
39      * @return
40      */
41     public static int getRandomNumber(int max){
42         return getRandomNumber(0, max);
43     }
44 }
45
46 package com.cyx.pokemon.item.equipment;
47
48 import com.cyx.pokemon.item.Item;
49
50 /**
51  * 装备
52  */
53 public abstract class Equipment extends Item {
54     /**
55      * 攻击力
56      */
57     protected int attack;
58     /**
59      * 防御力
60      */
61     protected int defense;
62     /**
63      * 生命值
64      */
65     protected int health;
66
67     public Equipment(String name, int levelNumber) {
68         super(name, levelNumber);
69     }
70
71     @Override
72     public String getItemInformation() {
73         return name + ": 攻击=" + attack + " 防御=" + defense + " 生命值=" +
health;
74     }
75 }
76
77 package com.cyx.pokemon.item.equipment;
78
79 import com.cyx.pokemon.util.Tools;
80

```



```
81  /**
82   * 头盔
83   */
84  public class Helmet extends Equipment{
85
86      public Helmet(int levelNumber) {
87          super("头盔", levelNumber);
88          this.attack = 0;
89          this.defense = Tools.getRandomNumber(20, 30, levelNumber);
90          this.health = Tools.getRandomNumber(100, 150, levelNumber);
91      }
92  }
93
94  package com.cyx.pokemon.item.equipment;
95
96  import com.cyx.pokemon.util.Tools;
97
98  /**
99   * 铠甲
100  */
101  public class Armor extends Equipment{
102
103      public Armor(int levelNumber) {
104          super("铠甲", levelNumber);
105          this.attack = 0;
106          this.defense = Tools.getRandomNumber(40, 50, levelNumber);
107          this.health = Tools.getRandomNumber(200, 250, levelNumber);
108      }
109  }
110
111  package com.cyx.pokemon.item.equipment;
112
113  import com.cyx.pokemon.util.Tools;
114
115  /**
116   * 护腿
117   */
118  public class Leggings extends Equipment{
119
120      public Leggings(int levelNumber) {
121          super("护腿", levelNumber);
122          this.attack = 0;
123          this.defense = Tools.getRandomNumber(30, 40, levelNumber);
124          this.health = Tools.getRandomNumber(150, 200, levelNumber);
125      }
126  }
127
128  package com.cyx.pokemon.item.equipment;
129
130  import com.cyx.pokemon.util.Tools;
131
132  /**
133   * 鞋子
134   */
135  public class Shoe extends Equipment{
136
137      public Shoe(int levelNumber) {
138          super("鞋子", levelNumber);
```

```

139         this.attack = 0;
140         this.defense = Tools.getRandomNumber(10, 20, levelNumber);
141         this.health = Tools.getRandomNumber(80, 100, levelNumber);
142     }
143 }
144
145 package com.cyx.pokemon.item.equipment;
146
147 import com.cyx.pokemon.util.Tools;
148
149 /**
150  * 武器
151  */
152 public class Weapon extends Equipment{
153
154     public Weapon(int levelNumber) {
155         super("武器", levelNumber);
156         this.attack = Tools.getRandomNumber(100, 150, levelNumber);
157         this.defense = 0;
158         this.health = Tools.getRandomNumber(250, 300, levelNumber);
159     }
160 }
161
162 package com.cyx.pokemon.item.equipment;
163
164 import com.cyx.pokemon.util.Tools;
165
166 /**
167  * 项链
168  */
169 public class Necklace extends Equipment{
170
171     public Necklace(int levelNumber) {
172         super("项链", levelNumber);
173         this.attack = Tools.getRandomNumber(25, 35, levelNumber);
174         this.defense = Tools.getRandomNumber(25, 25, levelNumber);
175         this.health = Tools.getRandomNumber(120, 180, levelNumber);
176     }
177 }
178
179 package com.cyx.pokemon.item.equipment;
180
181 import com.cyx.pokemon.util.Tools;
182
183 /**
184  * 戒指
185  */
186 public class Ring extends Equipment{
187
188     public Ring(int levelNumber) {
189         super("戒指", levelNumber);
190         this.attack = Tools.getRandomNumber(20, 30, levelNumber);
191         this.defense = Tools.getRandomNumber(20, 20, levelNumber);
192         this.health = Tools.getRandomNumber(100, 200, levelNumber);
193     }
194 }
195
196 package com.cyx.pokemon.item.equipment;

```

```

197
198 import com.cyx.pokemon.util.Tools;
199
200 /**
201  * 手镯
202  */
203 public class Bracelet extends Equipment{
204
205     public Bracelet(int levelNumber) {
206         super("手镯", levelNumber);
207         this.attack = Tools.getRandomNumber(20, 30, levelNumber);
208         this.defense = Tools.getRandomNumber(20, 20, levelNumber);
209         this.health = Tools.getRandomNumber(100, 200, levelNumber);
210     }
211 }

```

## 7.宠物小精灵分析

可以穿戴装备，最多只能穿戴8件装备，每个部位只能穿戴一件。宠物小精灵有4种类型，每一种都有名字，同类宠物小精灵都可以进行融合升星，升星提升初始属性50%。因此宠物小精灵应设计为抽象类。

```

1 package com.cyx.pokemon.item.pokemon;
2
3 import com.cyx.pokemon.item.Item;
4 import com.cyx.pokemon.item.equipment.*;
5
6 /**
7  * 宠物小精灵
8  */
9 public abstract class Pokemon extends Item {
10
11     /**
12      * 攻击力
13      */
14     protected int attack;
15     /**
16      * 防御力
17      */
18     protected int defense;
19     /**
20      * 生命值
21      */
22     protected int health;
23     /**
24      * 星级，默认1星
25      */
26     private int star = 1;
27     /**
28      * 宠物小精灵能够穿戴8件装备，默认是没有穿戴任何装备
29      * 穿戴顺序：头盔、铠甲、护腿、鞋子、武器、项链、戒指、手镯
30      */
31     private Equipment[] equipments = new Equipment[8];
32
33     public Pokemon(String name) {
34         super(name);
35     }
36

```

```

37
38     @Override
39     public String getItemInformation() {
40         return name + ": 攻击=" + attack + " 防御=" + defense + " 生命值=" +
health;
41     }
42     /**
43      * 与其他小精灵融合
44      * @param other 其他小精灵
45      */
46     public void merge(Pokemon other){
47         if(star == 10){
48             System.out.println(name + "星级已满，无法再融合升星");
49         } else {
50             this.attack += (other.attack >> 1);
51             this.defense += (other.defense >> 1);
52             this.health += (other.health >> 1);
53             star += 1;
54             System.out.println("融合成功");
55             System.out.println(getItemInformation());
56         }
57     }
58
59     /**
60      * 更换装备
61      * @param newEquipment 新装备
62      * @return
63      */
64     public Equipment changeEquipment(Equipment newEquipment){
65
66         int index;
67         if(newEquipment instanceof Helmet){//头盔
68             index = 0;
69         } else if(newEquipment instanceof Armor){//铠甲
70             index = 1;
71         } else if(newEquipment instanceof Leggings){//护腿
72             index = 2;
73         } else if(newEquipment instanceof Shoe){//鞋子
74             index = 3;
75         } else if(newEquipment instanceof weapon){//武器
76             index = 4;
77         } else if(newEquipment instanceof Necklace){//项链
78             index = 5;
79         } else if(newEquipment instanceof Ring){//戒指
80             index = 6;
81         } else {//手镯
82             index = 7;
83         }
84         //旧装备
85         Equipment old = equipments[index];
86         if(old == null){//未穿戴装备
87             equipments[index] = newEquipment;
88         } else {//已经穿戴装备
89
90         }
91         return old;
92     }
93 }

```

```
94
95 package com.cyx.pokemon.item.pokemon;
96
97 /**
98  * 妙蛙种子
99  */
100 public class Bulbasaur extends Pokemon{
101
102     public Bulbasaur() {
103         super("妙蛙种子");
104         this.attack = 60;
105         this.defense = 40;
106         this.health = 600;
107     }
108 }
109
110 package com.cyx.pokemon.item.pokemon;
111
112 /**
113  * 小火龙
114  */
115 public class Charmander extends Pokemon{
116
117     public Charmander() {
118         super("小火龙");
119         this.attack = 100;
120         this.defense = 80;
121         this.health = 1000;
122     }
123 }
124
125 package com.cyx.pokemon.item.pokemon;
126
127 /**
128  * 比卡丘
129  */
130 public class Bikachu extends Pokemon{
131
132     public Bikachu() {
133         super("比卡丘");
134         this.attack = 150;
135         this.defense = 100;
136         this.health = 2000;
137     }
138 }
139
140 package com.cyx.pokemon.item.pokemon;
141
142 /**
143  * 雷精灵
144  */
145 public class Jolteon extends Pokemon{
146
147     public Jolteon() {
148         super("雷精灵");
149         this.attack = 80;
150         this.defense = 60;
151         this.health = 800;
```

```
152     }
153 }
```

换装前应该比较新装备是否比旧装备好，因此需要在装备类 `Equipment` 中添加装备比较的方法

```
1  /**
2   * 是否比其他装备好
3   * @param other
4   * @return
5   */
6  public boolean isBetter(Equipment other){
7      //首先必须保证是同类型装备
8      if(this.getClass() == other.getClass()){
9          int total1 = this.attack + this.defense + this.health >> 1;
10         int total2 = other.attack + other.defense + other.health >> 1;
11         return total1 > total2;
12     }
13     return false;
14 }
```

宠物小精灵类 `Pokemon` 更换装备方法修改

```
1  /**
2   * 更换装备
3   * @param newEquipment 新装备
4   * @return
5   */
6  public Equipment changeEquipment(Equipment newEquipment){
7      int index;
8      if(newEquipment instanceof Helmet){//头盔
9          index = 0;
10     } else if(newEquipment instanceof Armor){//铠甲
11         index = 1;
12     } else if(newEquipment instanceof Leggings){//护腿
13         index = 2;
14     } else if(newEquipment instanceof Shoe){//鞋子
15         index = 3;
16     } else if(newEquipment instanceof weapon){//武器
17         index = 4;
18     } else if(newEquipment instanceof Necklace){//项链
19         index = 5;
20     } else if(newEquipment instanceof Ring){//戒指
21         index = 6;
22     } else { //手镯
23         index = 7;
24     }
25     //旧装备
26     Equipment old = equipments[index];
27     if(old == null){ //未穿戴装备
28         equipments[index] = newEquipment;
29     } else { //已经穿戴装备
30         //新装备比就装备好
31         if(newEquipment.isBetter(old)){
32             equipments[index] = newEquipment;
33         } else {
34             old = newEquipment;
35         }
36     }
37 }
```

```

35     }
36 }
37 return old;
38 }

```

换装后，宠物小精灵的攻击、防御、血量会发生改变，但不能改变宠物小精灵本身的属性。因此获取攻击、防御和血量时，应该将宠物小精灵本身的属性 + 所穿戴装备的属性。

```

1  public int getAttack() {
2      int totalAttack = attack;
3      for(Equipment equipment: equipments){
4          if(equipment != null)
5              totalAttack += equipment.getAttack();
6      }
7      return totalAttack;
8  }
9
10 public int getDefense() {
11     int totalDefense = defense;
12     for(Equipment equipment: equipments){
13         if(equipment != null)
14             totalDefense += equipment.getDefense();
15     }
16     return totalDefense;
17 }
18
19 public int getHealth() {
20     int totalHealth = health;
21     for(Equipment equipment: equipments){
22         if(equipment != null)
23             totalHealth += equipment.getHealth();
24     }
25     return totalHealth;
26 }
27
28 @Override
29 public String getItemInformation() {
30     return name + ": 攻击=" + getAttack() + " 防御=" + getDefense() + " 生命值"
31         + getHealth();
32 }

```

到此，宝箱 Treasure 打开的物品都已经设计，可以实现宝箱开启功能

```

1  /**
2   * 开启宝箱能够获得一个物品
3   * @return
4   */
5  public Item open(){
6      //取[0, 10)范围内的随机数
7      int number = Tools.getRandomNumber(10);
8      if(number == 0){//获得宠物小精灵，
9          //比例 初级：中级：高级：究级 = 80:15:4:1
10         int rate = Tools.getRandomNumber(100);
11         if(rate == 0){//究级宠物小精灵=>皮卡丘
12             return new Bikachu();
13         } else if(rate <= 4){//高级宠物小精灵 => 小火龙

```

```

14         return new Charmander();
15     } else if(rate <= 20){//中级宠物小精灵 => 雷精灵
16         return new Jolteon();
17     } else {//初级宠物小精灵
18         return new Bulbasaur();
19     }
20 } else if(number <= 3){//获得装备
21     //比例 武器: 项链: 戒指: 手镯: 头盔: 铠甲: 护腿: 鞋子 = 3:5:8:8:19:19:19:19
22     int rate = Tools.getRandomNumber(100);
23     if(rate < 3){//武器
24         return new Weapon(levelNumber);
25     } else if(rate < 8){//项链
26         return new Necklace(levelNumber);
27     } else if(rate < 16){//戒指
28         return new Ring(levelNumber);
29     } else if(rate < 24){//手镯
30         return new Bracelet(levelNumber);
31     } else if(rate < 43){//头盔
32         return new Helmet(levelNumber);
33     } else if(rate < 62){//铠甲
34         return new Armor(levelNumber);
35     } else if(rate < 81){//护腿
36         return new Leggings(levelNumber);
37     } else {//鞋子
38         return new Shoe(levelNumber);
39     }
40 } else {//获得药品
41     return new HP(levelNumber, 10);
42 }
43 }

```

## 8.怪物分析

可以阻挡冒险家探索地图。怪物会与宠物小精灵对抗，也就是会攻击宠物小精灵。

```

1  /**
2   * 攻击宠物小精灵
3   * @param pokemon 宠物小精灵
4   */
5  public void attackPokemon(Pokemon pokemon){
6      int minusHealth = this.attack * this.attack / pokemon.getDefense();
7      if(minusHealth == 0) {//伤害为0, 需要调整
8          minusHealth = 1; //调整伤害为1点
9      }
10     pokemon.setHealth(pokemon.getHealth() - minusHealth);
11     System.out.println(name + "对" + pokemon.getName() + "发动攻击, 造成了" +
12         minusHealth + "伤害");
12 }

```

这样设计存在问题：怪物攻击宠物小精灵时，宠物小精灵的血量总值一直在减少，但是却无法使用药品恢复血量。因此，需要在宠物小精灵类 `Pokemon` 中添加一个新的变量来记录宠物小精灵的当前血量，并提供获取和更改血量的方法。所有宠物小精灵类都需要做相应的修改

```

1  package com.cyx.pokemon.item.pokemon;
2
3  import com.cyx.pokemon.item.Item;

```



```

4  import com.cyx.pokemon.item.equipment.*;
5
6  /**
7   * 宠物小精灵
8   */
9  public abstract class Pokemon extends Item {
10
11     /**
12      * 攻击力
13      */
14     protected int attack;
15
16     /**
17      * 防御力
18      */
19     protected int defense;
20
21     /**
22      * 生命值
23      */
24     protected int health;
25
26     /**
27      * 当前生命值
28      */
29     protected int currentHealth;
30
31     /**
32      * 星级, 默认1星
33      */
34     private int star = 1;
35
36     /**
37      * 宠物小精灵能够穿戴8件装备, 默认是没有穿戴任何装备
38      * 穿戴顺序: 头盔、铠甲、护腿、鞋子、武器、项链、戒指、手镯
39      */
40     private Equipment[] equipments = new Equipment[8];
41
42     public Pokemon(String name) {
43         super(name);
44     }
45
46     public int getAttack() {
47         int totalAttack = attack;
48         for(Equipment equipment: equipments){
49             if(equipment != null)
50                 totalAttack += equipment.getAttack();
51         }
52         return totalAttack;
53     }
54
55     public int getDefense() {
56         int totalDefense = defense;
57         for(Equipment equipment: equipments){
58             if(equipment != null)
59                 totalDefense += equipment.getDefense();
60         }
61         return totalDefense;
62     }
63
64     public int getHealth() {
65         int totalHealth = health;

```

```

62         for(Equipment equipment: equipments){
63             if(equipment != null)
64                 totalHealth += equipment.getHealth();
65         }
66         return totalHealth;
67     }
68
69     public void setHealth(int health) {
70         this.health = health;
71     }
72
73     public int getCurrentHealth() {
74         return currentHealth;
75     }
76
77     public void setCurrentHealth(int currentHealth) {
78         this.currentHealth = currentHealth;
79     }
80
81     @Override
82     public String getItemInformation() {
83         return name + ": 攻击=" + getAttack() + " 防御=" + getDefense() + "
生命值=" + getHealth();
84     }
85     /**
86      * 与其他小精灵融合
87      * @param other 其他小精灵
88      */
89     public void merge(Pokemon other){
90         if(star == 10){
91             System.out.println(name + "星级已满，无法再融合升星");
92         } else {
93             this.attack += (other.attack >> 1);
94             this.defense += (other.defense >> 1);
95             this.health += (other.health >> 1);
96             star += 1;
97             System.out.println("融合成功");
98             System.out.println(getItemInformation());
99         }
100     }
101
102     /**
103      * 更换装备
104      * @param newEquipment 新装备
105      * @return
106      */
107     public Equipment changeEquipment(Equipment newEquipment){
108         int index;
109         if(newEquipment instanceof Helmet){//头盔
110             index = 0;
111         } else if(newEquipment instanceof Armor){//铠甲
112             index = 1;
113         } else if(newEquipment instanceof Leggings){//护腿
114             index = 2;
115         } else if(newEquipment instanceof Shoe){//鞋子
116             index = 3;
117         } else if(newEquipment instanceof weapon){//武器
118             index = 4;

```

```

119         } else if(newEquipment instanceof Necklace){//项链
120             index = 5;
121         } else if(newEquipment instanceof Ring){//戒指
122             index = 6;
123         } else {//手镯
124             index = 7;
125         }
126         //旧装备
127         Equipment old = equipments[index];
128         if(old == null){//未穿戴装备
129             equipments[index] = newEquipment;
130         } else {//已经穿戴装备
131             //新装备比就装备好
132             if(newEquipment.isBetter(old)){
133                 equipments[index] = newEquipment;
134             } else {
135                 old = newEquipment;
136             }
137         }
138         return old;
139     }
140 }
141
142 package com.cyx.pokemon.item.pokemon;
143
144 /**
145  * 小火龙
146  */
147 public class Charmander extends Pokemon{
148
149     public Charmander() {
150         super("小火龙");
151         this.attack = 100;
152         this.defense = 80;
153         this.health = 1000;
154         this.currentHealth = this.health;
155     }
156 }
157
158 package com.cyx.pokemon.item.pokemon;
159
160 /**
161  * 雷精灵
162  */
163 public class Jolteon extends Pokemon{
164
165     public Jolteon() {
166         super("雷精灵");
167         this.attack = 80;
168         this.defense = 60;
169         this.health = 800;
170         this.currentHealth = this.health;
171     }
172 }
173
174 package com.cyx.pokemon.item.pokemon;
175
176 /**

```

```
177  * 皮卡丘
178  */
179  public class Bikachu extends Pokemon{
180
181      public Bikachu() {
182          super("皮卡丘");
183          this.attack = 150;
184          this.defense = 100;
185          this.health = 2000;
186          this.currentHealth = this.health;
187      }
188  }
189
190  package com.cyx.pokemon.item.pokemon;
191
192  /**
193   * 妙蛙种子
194   */
195  public class Bulbasaur extends Pokemon{
196
197      public Bulbasaur() {
198          super("妙蛙种子");
199          this.attack = 60;
200          this.defense = 40;
201          this.health = 600;
202          this.currentHealth = this.health;
203      }
204  }
205
206  package com.cyx.pokemon.item;
207
208  import com.cyx.pokemon.DisplayItem;
209
210  /**
211   * 物品
212   */
213  public abstract class Item implements DisplayItem {
214
215      /**
216       * 物品名称
217       */
218      protected String name;
219
220      /**
221       * 关卡编号
222       */
223      protected int levelNumber;
224
225      /**
226       * 是否被探索
227       */
228      protected boolean discovery;
229
230      public Item(String name) {
231          this.name = name;
232      }
233
234      public Item(String name, int levelNumber) {
235          this.name = name;
236          this.levelNumber = levelNumber;
237      }
238  }
```

```

235     }
236
237     public void setDiscovery(boolean discovery) {
238         this.discovery = discovery;
239     }
240
241     public String getName() {
242         return name;
243     }
244 }
245
246 package com.cyx.pokemon.item.monster;
247
248 import com.cyx.pokemon.item.Item;
249 import com.cyx.pokemon.item.pokemon.Pokemon;
250
251 /**
252  * 怪物
253  */
254 public class Monster extends Item {
255
256     /**
257      * 攻击力
258      */
259     protected int attack;
260
261     /**
262      * 防御力
263      */
264     protected int defense;
265
266     /**
267      * 生命值
268      */
269     protected int health;
270
271     public Monster(String name, int levelNumber) {
272         super(name, levelNumber);
273     }
274
275     /**
276      * 攻击宠物小精灵
277      * @param pokemon 宠物小精灵
278      */
279     public void attackPokemon(Pokemon pokemon){
280         int minusHealth = this.attack * this.attack /
281         pokemon.getDefense();
282         if(minusHealth == 0) { //伤害为0, 需要调整
283             minusHealth = 1; //调整伤害为1点
284         } else if(minusHealth > pokemon.getCurrentHealth()) { //如果伤害比宠物
285             小精灵当前血量还要高
286             minusHealth = pokemon.getCurrentHealth(); //伤害就应该等于宠物小精
287             灵当前血量
288         }
289         //剩余血量
290         int restHealth = pokemon.getCurrentHealth() - minusHealth;
291         pokemon.setCurrentHealth(restHealth);
292         System.err.println(name + "对" + pokemon.getName() + "发动攻击, 造成
293         了" + minusHealth + "伤害");

```

```

289     }
290
291     @Override
292     public String getItemInformation() {
293         return discovery ? name : "■";
294     }
295 }

```

怪物会攻击宠物小精灵，宠物小精灵也会攻击怪物

```

1  package com.cyx.pokemon.item.monster;
2
3  import com.cyx.pokemon.item.Item;
4  import com.cyx.pokemon.item.pokemon.Pokemon;
5
6  /**
7   * 怪物
8   */
9  public class Monster extends Item {
10
11     /**
12      * 攻击力
13      */
14     protected int attack;
15
16     /**
17      * 防御力
18      */
19     protected int defense;
20
21     /**
22      * 生命值
23      */
24     protected int health;
25
26     /**
27      * 怪物当前血量
28      */
29     protected int currentHealth;
30
31
32
33     public Monster(String name, int levelNumber) {
34         super(name, levelNumber);
35     }
36
37     public int getDefense() {
38         return defense;
39     }
40
41     public int getCurrentHealth() {
42         return currentHealth;
43     }
44
45     public void setCurrentHealth(int currentHealth) {
46         this.currentHealth = currentHealth;
47     }
48
49     /**
50      * 攻击宠物小精灵
51      * @param pokemon 宠物小精灵

```

```

48     */
49     public void attackPokemon(Pokemon pokemon){
50         int minusHealth = this.attack * this.attack /
pokemon.getDefense();
51         if(minusHealth == 0) { //伤害为0, 需要调整
52             minusHealth = 1; //调整伤害为1点
53         } else if(minusHealth > pokemon.getCurrentHealth()) { //如果伤害比宠物
小精灵当前血量还要高
54             minusHealth = pokemon.getCurrentHealth(); //伤害就应该等于宠物小精
灵当前血量
55         }
56         //剩余血量
57         int restHealth = pokemon.getCurrentHealth() - minusHealth;
58         pokemon.setCurrentHealth(restHealth);
59         System.err.println(name + "对" + pokemon.getName() + "发动攻击, 造成
了" + minusHealth + "伤害");
60     }
61
62     @Override
63     public String getItemInformation() {
64         return discovery ? name : "■";
65     }
66 }
67
68 package com.cyx.pokemon.item.monster;
69
70 import com.cyx.pokemon.util.Tools;
71
72 /**
73  * 牛魔怪
74  */
75 public class CattleMonster extends Monster{
76
77     public CattleMonster(int levelNumber) {
78         super("牛魔怪", levelNumber);
79         this.attack = Tools.getRandomNumber(50, 60, levelNumber);
80         this.defense = Tools.getRandomNumber(40, 50, levelNumber);
81         this.health = Tools.getRandomNumber(700, 900, levelNumber);
82         this.currentHealth = this.health;
83     }
84 }
85
86 package com.cyx.pokemon.item.monster;
87
88 import com.cyx.pokemon.util.Tools;
89
90 /**
91  * 铁炮鱼
92  */
93 public class Ramoraid extends Monster{
94
95     public Ramoraid(int levelNumber) {
96         super("铁炮鱼", levelNumber);
97         this.attack = Tools.getRandomNumber(60, 70, levelNumber);
98         this.defense = Tools.getRandomNumber(50, 60, levelNumber);
99         this.health = Tools.getRandomNumber(900, 1100, levelNumber);
100         this.currentHealth = this.health;
101     }

```

```

102 }
103
104 package com.cyx.pokemon.item.monster;
105
106 import com.cyx.pokemon.util.Tools;
107
108 /**
109  * 火焰鸟
110  */
111 public class Moltres extends Monster{
112
113     public Moltres(int levelNumber) {
114         super("火焰鸟", levelNumber);
115         this.attack = Tools.getRandomNumber(80, 100, levelNumber);
116         this.defense = Tools.getRandomNumber(70, 90, levelNumber);
117         this.health = Tools.getRandomNumber(1400, 1800, levelNumber);
118         this.currentHealth = this.health;
119     }
120 }
121
122 package com.cyx.pokemon.item.monster;
123
124 import com.cyx.pokemon.util.Tools;
125
126 /**
127  * 象牙猪
128  */
129 public class Mamoswine extends Monster {
130
131     public Mamoswine(int levelNumber) {
132         super("象牙猪", levelNumber);
133         this.attack = Tools.getRandomNumber(45, 55, levelNumber);
134         this.defense = Tools.getRandomNumber(35, 45, levelNumber);
135         this.health = Tools.getRandomNumber(600, 800, levelNumber);
136         this.currentHealth = this.health;
137     }
138 }
139
140 package com.cyx.pokemon.item.pokemon;
141
142 import com.cyx.pokemon.item.Item;
143 import com.cyx.pokemon.item.equipment.*;
144 import com.cyx.pokemon.item.monster.Monster;
145
146 /**
147  * 宠物小精灵
148  */
149 public abstract class Pokemon extends Item {
150
151     /**
152      * 攻击力
153      */
154     protected int attack;
155     /**
156      * 防御力
157      */
158     protected int defense;
159     /**

```



```

160     * 生命值
161     */
162     protected int health;
163     /**
164     * 当前生命值
165     */
166     protected int currentHealth;
167     /**
168     * 星级，默认1星
169     */
170     private int star = 1;
171     /**
172     * 宠物小精灵能够穿戴8件装备，默认是没有穿戴任何装备
173     * 穿戴顺序：头盔、铠甲、护腿、鞋子、武器、项链、戒指、手镯
174     */
175     private Equipment[] equipments = new Equipment[8];
176
177     public Pokemon(String name) {
178         super(name);
179     }
180
181
182     public int getAttack() {
183         int totalAttack = attack;
184         for(Equipment equipment: equipments){
185             if(equipment != null)
186                 totalAttack += equipment.getAttack();
187         }
188         return totalAttack;
189     }
190
191     public int getDefense() {
192         int totalDefense = defense;
193         for(Equipment equipment: equipments){
194             if(equipment != null)
195                 totalDefense += equipment.getDefense();
196         }
197         return totalDefense;
198     }
199
200     public int getHealth() {
201         int totalHealth = health;
202         for(Equipment equipment: equipments){
203             if(equipment != null)
204                 totalHealth += equipment.getHealth();
205         }
206         return totalHealth;
207     }
208
209     public void setHealth(int health) {
210         this.health = health;
211     }
212
213     public int getCurrentHealth() {
214         return currentHealth;
215     }
216
217     public void setCurrentHealth(int currentHealth) {

```

```

218         this.currentHealth = currentHealth;
219     }
220
221     @Override
222     public String getItemInformation() {
223         return name + ": 攻击=" + getAttack() + " 防御=" + getDefense() + "
生命值=" + getHealth();
224     }
225
226     /**
227      * 宠物小精灵攻击怪物
228      * @param monster 怪物
229      */
230     public void attackMonster(Monster monster){
231         int minusHealth = this.attack * this.attack /
monster.getDefense();
232         if(minusHealth == 0) { //伤害为0, 需要调整
233             minusHealth = 1; //调整伤害为1点
234         } else if(minusHealth > monster.getCurrentHealth()) { //如果伤害比怪物
当前血量还要高
235             minusHealth = monster.getCurrentHealth(); //伤害就应该等于怪物当前
血量
236         }
237         //剩余血量
238         int restHealth = monster.getCurrentHealth() - minusHealth;
239         monster.setCurrentHealth(restHealth);
240         System.out.println(name + "对" + monster.getName() + "发动攻击, 造成
了" + minusHealth + "伤害");
241     }
242
243     /**
244      * 与其他小精灵融合
245      * @param other 其他小精灵
246      */
247     public void merge(Pokemon other){
248         if(star == 10){
249             System.out.println(name + "星级已满, 无法再融合升星");
250         } else {
251             this.attack += (other.attack >> 1);
252             this.defense += (other.defense >> 1);
253             this.health += (other.health >> 1);
254             star += 1;
255             System.out.println("融合成功");
256             System.out.println(getItemInformation());
257         }
258     }
259
260     /**
261      * 更换装备
262      * @param newEquipment 新装备
263      * @return
264      */
265     public Equipment changeEquipment(Equipment newEquipment){
266         int index;
267         if(newEquipment instanceof Helmet) { //头盔
268             index = 0;
269         } else if(newEquipment instanceof Armor) { //铠甲
270             index = 1;

```

```

271     } else if(newEquipment instanceof Leggings){//护腿
272         index = 2;
273     } else if(newEquipment instanceof Shoe){//鞋子
274         index = 3;
275     } else if(newEquipment instanceof weapon){//武器
276         index = 4;
277     } else if(newEquipment instanceof Necklace){//项链
278         index = 5;
279     } else if(newEquipment instanceof Ring){//戒指
280         index = 6;
281     } else {//手镯
282         index = 7;
283     }
284     //旧装备
285     Equipment old = equipments[index];
286     if(old == null){//未穿戴装备
287         equipments[index] = newEquipment;
288     } else {//已经穿戴装备
289         //新装备比就装备好
290         if(newEquipment.isBetter(old)){
291             equipments[index] = newEquipment;
292         } else {
293             old = newEquipment;
294         }
295     }
296     return old;
297 }
298 }

```

怪物死后会掉落装备、药品或宠物小精灵。

```

1  package com.cyx.pokemon.util;
2
3  import com.cyx.pokemon.item.HP;
4  import com.cyx.pokemon.item.Item;
5  import com.cyx.pokemon.item.equipment.*;
6  import com.cyx.pokemon.item.pokemon.Bikachu;
7  import com.cyx.pokemon.item.pokemon.Bulbasaur;
8  import com.cyx.pokemon.item.pokemon.Charmander;
9  import com.cyx.pokemon.item.pokemon.Jolteon;
10
11  import java.util.Random;
12
13  /**
14   * 工具类
15   */
16  public class Tools {
17      /**
18       * 随机数对象
19       */
20      private static final Random RANDOM = new Random();
21
22      /**
23       * 获取给定范围内的随机数
24       * @param min 最小值
25       * @param max 最大值
26       * @param levelNumber 关卡编号

```

```

27     * @return
28     */
29     public static int getRandomNumber(int min, int max, int levelNumber){
30         int diff = (max - min) * levelNumber;
31         return RANDOM.nextInt(diff) + min * levelNumber;
32     }
33
34     /**
35     * 获取给定范围内的随机数
36     * @param min 最小值
37     * @param max 最大值
38     * @return
39     */
40     public static int getRandomNumber(int min, int max){
41         return getRandomNumber(min, max, 1);
42     }
43
44     /**
45     * 获取给定范围内的随机数
46     * @param max 最大值
47     * @return
48     */
49     public static int getRandomNumber(int max){
50         return getRandomNumber(0, max);
51     }
52
53     /**
54     * 获取一个随机物品
55     * @param levelNumber 关卡编号
56     * @return
57     */
58     public static Item getRandomItem(int levelNumber){
59         //取[0, 10)范围内的随机数
60         int number = Tools.getRandomNumber(10);
61         if(number == 0){//获得宠物小精灵,
62             //比例 初级: 中级: 高级: 究级 = 80:15:4:1
63             int rate = Tools.getRandomNumber(100);
64             if(rate == 0){//究级宠物小精灵=>皮卡丘
65                 return new Bikachu();
66             } else if(rate <= 4){//高级宠物小精灵 => 小火龙
67                 return new Charmander();
68             } else if(rate <= 20){//中级宠物小精灵 => 雷精灵
69                 return new Jolteon();
70             } else {//初级宠物小精灵
71                 return new Bulbasaur();
72             }
73         } else if(number <= 3){//获得装备
74             //比例 武器: 项链: 戒指: 手镯: 头盔: 铠甲: 护腿: 鞋子 =
75             3:5:8:8:19:19:19:19
76             int rate = Tools.getRandomNumber(100);
77             if(rate < 3){//武器
78                 return new Weapon(levelNumber);
79             } else if(rate < 8){//项链
80                 return new Necklace(levelNumber);
81             } else if(rate < 16){//戒指
82                 return new Ring(levelNumber);
83             } else if(rate < 24){//手镯
84                 return new Bracelet(levelNumber);

```

```

84         } else if(rate < 43){//头盔
85             return new Helmet(levelNumber);
86         } else if(rate < 62){//铠甲
87             return new Armor(levelNumber);
88         } else if(rate < 81){//护腿
89             return new Leggings(levelNumber);
90         } else {//鞋子
91             return new Shoe(levelNumber);
92         }
93     } else {//获得药品
94         return new HP(levelNumber, 10);
95     }
96 }
97 }
98
99 package com.cyx.pokemon.item;
100
101 import com.cyx.pokemon.item.equipment.*;
102 import com.cyx.pokemon.item.pokemon.Bikachu;
103 import com.cyx.pokemon.item.pokemon.Bulbasaur;
104 import com.cyx.pokemon.item.pokemon.Charmander;
105 import com.cyx.pokemon.item.pokemon.Jolteon;
106 import com.cyx.pokemon.util.Tools;
107
108 /**
109  * 宝箱
110  */
111 public class Treasure extends Item {
112
113     public Treasure(int levelNumber) {
114         super("宝箱", levelNumber);
115     }
116
117     /**
118      * 开启宝箱能够获得一个物品
119      * @return
120      */
121     public Item open(){
122         return Tools.getRandomItem(levelNumber);
123     }
124
125     @Override
126     public String getItemInformation() {
127         return discovery ? "◉" : "■";
128     }
129 }
130
131 package com.cyx.pokemon.item.monster;
132
133 import com.cyx.pokemon.item.Item;
134 import com.cyx.pokemon.item.pokemon.Pokemon;
135 import com.cyx.pokemon.util.Tools;
136
137 /**
138  * 怪物
139  */
140 public class Monster extends Item {
141

```

```

142     /**
143      * 攻击力
144      */
145     protected int attack;
146     /**
147      * 防御力
148      */
149     protected int defense;
150     /**
151      * 生命值
152      */
153     protected int health;
154     /**
155      * 怪物当前血量
156      */
157     protected int currentHealth;
158
159
160     public Monster(String name, int levelNumber) {
161         super(name, levelNumber);
162     }
163
164     public int getDefense() {
165         return defense;
166     }
167
168     public int getCurrentHealth() {
169         return currentHealth;
170     }
171
172     public void setCurrentHealth(int currentHealth) {
173         this.currentHealth = currentHealth;
174     }
175
176     /**
177      * 攻击宠物小精灵
178      * @param pokemon 宠物小精灵
179      */
180     public void attackPokemon(Pokemon pokemon){
181         int minusHealth = this.attack * this.attack /
pokemon.getDefense();
182         if(minusHealth == 0) { //伤害为0，需要调整
183             minusHealth = 1; //调整伤害为1点
184         } else if(minusHealth > pokemon.getCurrentHealth()) { //如果伤害比宠物
小精灵当前血量还要高
185             minusHealth = pokemon.getCurrentHealth(); //伤害就应该等于宠物小精
灵当前血量
186         }
187         //剩余血量
188         int restHealth = pokemon.getCurrentHealth() - minusHealth;
189         pokemon.setCurrentHealth(restHealth);
190         System.err.println(name + "对" + pokemon.getName() + "发动攻击，造成
了" + minusHealth + "伤害");
191     }
192
193     /**
194      * 怪物掉落装备
195      * @return

```

```

196     */
197     public Item drop(){
198         return Tools.getRandomItem(levelNumber);
199     }
200
201     @Override
202     public String getItemInformation() {
203         return discovery ? name : "■";
204     }
205 }

```

## 9. 关卡设计

关卡有编号、有地图。

```

1  package com.cyx.pokemon.level;
2
3  /**
4   * 关卡
5   */
6  public class Level {
7      /**
8       * 关卡编号
9       */
10     private int number;
11     /**
12      * 关卡地图
13      */
14     private LevelMap map;
15
16     private Level prevLevel;
17
18     private Level nextLevel;
19
20     public Level(Level prevLevel, int number, Level nextLevel) {
21         this.number = number;
22         this.prevLevel = prevLevel;
23         this.nextLevel = nextLevel;
24         this.map = new LevelMap(number);
25     }
26
27     public int getNumber() {
28         return number;
29     }
30
31     public LevelMap getMap() {
32         return map;
33     }
34
35     public Level getPrevLevel() {
36         return prevLevel;
37     }
38
39     public void setPrevLevel(Level prevLevel) {
40         this.prevLevel = prevLevel;
41     }
42

```

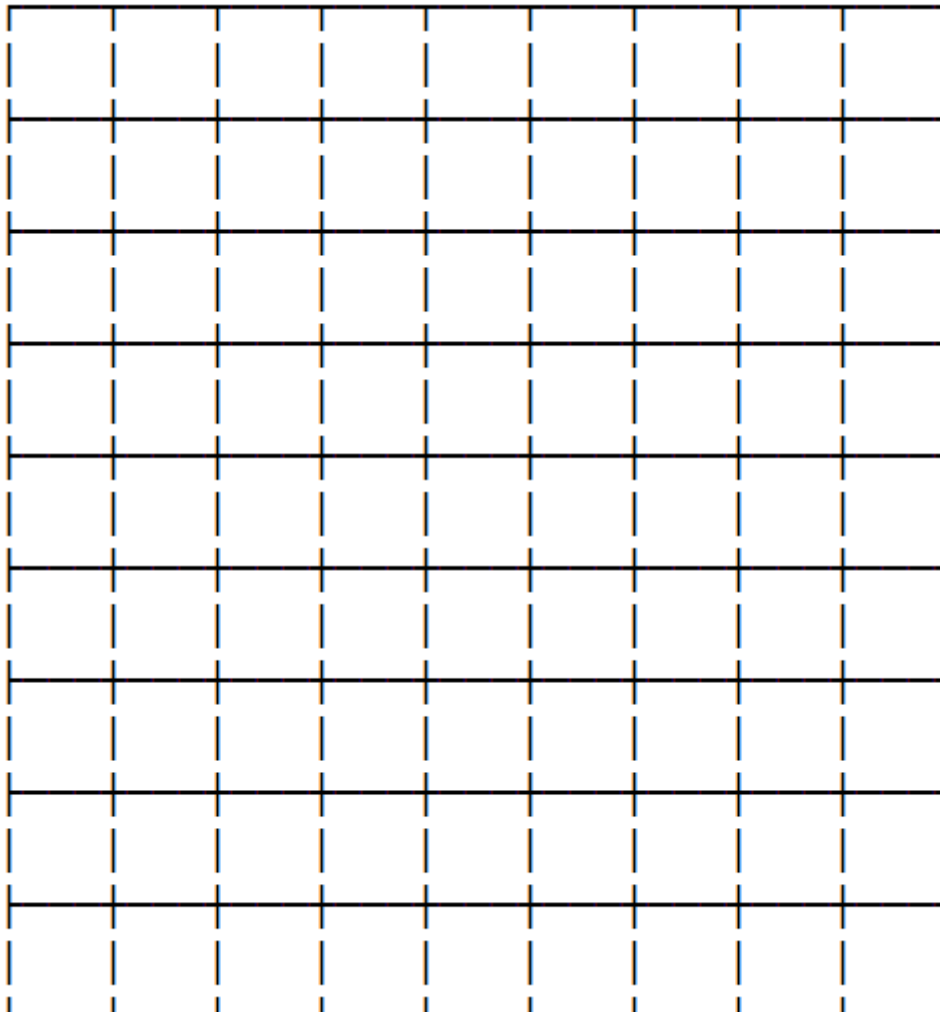
```

43     public Level getNextLevel() {
44         return nextLevel;
45     }
46
47     public void setNextLevel(Level nextLevel) {
48         this.nextLevel = nextLevel;
49     }
50 }
51
52 package com.cyx.pokemon.level;
53
54 /**
55  * 关卡地图
56  */
57 public class LevelMap {
58
59     private int number;
60
61     public LevelMap(int number) {
62         this.number = number;
63     }
64 }

```

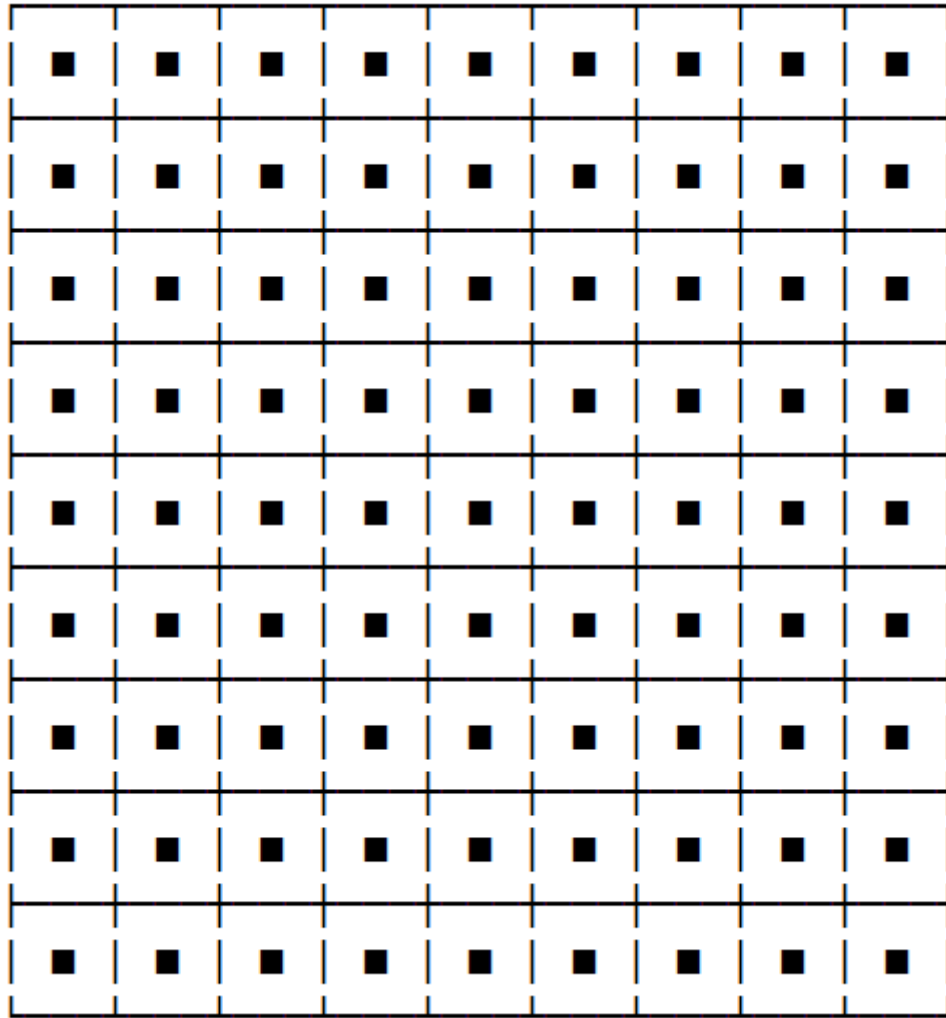
## 10. 地图分析

地图网格绘制，如图

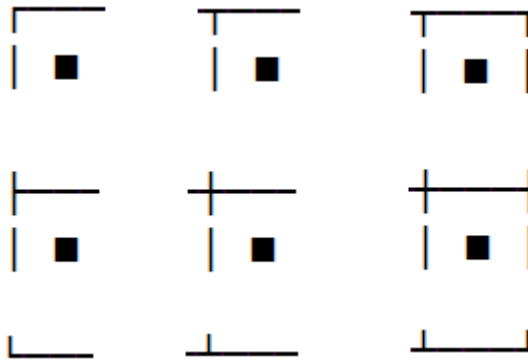




每一个网格中都有物品



可以看作图形组合



每一个物品有两行信息，上面一行纯网格线，下面一行有网格线，也有物品信息

```
1 package com.cyx.pokemon.level;
2
3 import com.cyx.pokemon.DisplayItem;
4 import com.cyx.pokemon.item.monster.Mamoswine;
5
6 /**
7  * 关卡地图
8  */
9 public class LevelMap {
10     /**
11      * 关卡编号
```

```

12     */
13     private int number;
14     /**
15      * 地图上的物品: 9x9
16      */
17     private final DisplayItem[][] items = new DisplayItem[9][9];
18
19     public LevelMap(int number) {
20         this.number = number;
21         generate();
22     }
23
24     /**
25      * 生成地图
26      */
27     private void generate(){
28         for(int i=0; i<items.length; i++){
29             for(int j=0; j<items[i].length; j++){
30                 items[i][j] = new Mamoswine(number);
31             }
32         }
33     }
34
35     /**
36      * 展示地图
37      */
38     public void show(){
39         for(int i=0; i<items.length; i++){
40             String line1 = "", line2 = "";
41             for(int j=0; j<items[i].length; j++){
42                 if(i == 0){//第一行
43                     if(j == 0){//第一列
44                         line1 += "┌───";
45                         line2 += "| " + items[i][j].getItemInformation() +
46 " ";
47                     } else if(j == items[i].length-1){//最后一列
48                         line1 += "└───";
49                         line2 += "| " + items[i][j].getItemInformation() +
50 " |";
51                     } else {
52                         line1 += "├───";
53                         line2 += "| " + items[i][j].getItemInformation() +
54 " ";
55                     }
56                 } else if(j == 0){//第一列
57                     line1 += "┌───";
58                     line2 += "| " + items[i][j].getItemInformation() +
59 " ";
60                 } else if(j == items[i].length-1){//最后一列
61                     line1 += "└───";
62                     line2 += "| " + items[i][j].getItemInformation() +
63 " |";
64                 } else {
65                     line1 += "├───";
66                     line2 += "| " + items[i][j].getItemInformation() +
67 " ";
68                 }
69             }
70         }
71     }

```

```

64         }
65     }
66     System.out.println(line1);
67     System.out.println(line2);
68 }
69 String lastLine = ""; //最后一行网格线
70 for(int i=0; i<items[0].length; i++){
71     if(i==0){ //第一列
72         lastLine += "┌───";
73     } else if(i == items[0].length -1){ //最后一列
74         lastLine += "└───";
75     } else {
76         lastLine += "│───";
77     }
78 }
79 System.out.println(lastLine);
80 }
81 }
82
83 package com.cyx.pokemon;
84
85 import com.cyx.pokemon.level.LevelMap;
86
87 /**
88  * 启动类
89  */
90 public class Launcher {
91
92     public static void main(String[] args) {
93         LevelMap map = new LevelMap(1);
94         map.show();
95     }
96 }

```

地图随机生成，上面包含有宝箱、怪物和传送门，宝箱可以开出药品、装备或宠物小精灵，比例为 6 : 3 : 1。如果是第一关，第一个位置为冒险家进入地图的位置，第二个位置为初级怪物象牙猪。如果是其他关卡，第一个位置为返回上一关卡的传送门，第二个位置为冒险家进入地图的位置。

```

1  package com.cyx.pokemon.level;
2
3  import com.cyx.pokemon.DisplayItem;
4  import com.cyx.pokemon.item.Portal;
5  import com.cyx.pokemon.item.Treasure;
6  import com.cyx.pokemon.item.monster.CattleMonster;
7  import com.cyx.pokemon.item.monster.Mamoswine;
8  import com.cyx.pokemon.item.monster.Moltres;
9  import com.cyx.pokemon.item.monster.Ramoraid;
10 import com.cyx.pokemon.util.Tools;
11
12 /**
13  * 关卡地图
14  */
15 public class LevelMap {
16     /**
17     * 关卡编号
18     */
19     private int number;

```

```

20  /**
21  * 地图上的物品： 9x9
22  */
23  private final DisplayItem[][] items = new DisplayItem[9][9];
24
25  public LevelMap(int number) {
26      this.number = number;
27      generate();
28  }
29
30  /**
31  * 生成地图=> 宝箱：怪物：传送门 = 39:39:1
32  * 第一个位置和第二个位置不能使用
33  */
34  private void generate(){
35      if(number == 1){//第一关卡
36          //第二个位置为初级怪物象牙猪
37          items[0][1] = new Mamoswine(number);
38          items[0][0] = new Mamoswine(number);
39      } else {//其他关卡
40          //第一个位置为返回上一层的传送门
41          items[0][0] = new Portal(false);
42          items[0][1] = new Portal(false);
43      }
44      //记录生成的宝箱数量
45      int generatedTreasure = 0;
46      //记录生成的怪物数量
47      int generatedMonster1 = 0;//记录生成的初级怪物数量
48      int generatedMonster2 = 0;//记录生成的中级怪物数量
49      int generatedMonster3 = 0;//记录生成的高级怪物数量
50      int generatedMonster4 = 0;//记录生成的究级怪物数量
51      //记录生成的宝箱数量
52      int generatedPortal = 0;
53      while (generatedTreasure < 39
54          || (generatedMonster1 + generatedMonster2 +
55          generatedMonster3 + generatedMonster4) < 39
56          || generatedPortal == 0){
57          //获取随机坐标
58          int index = Tools.getRandomNumber(2, 81);
59          //计算行和列
60          int row = index / items[0].length;
61          int col = index % items[0].length;
62          //目标位置已经有物品存在
63          if(items[row][col] != null) continue;
64          //获取一个随机数
65          int rate = Tools.getRandomNumber(79);
66          if(rate == 0){//传送门
67              //传送门已经生成了，直接跳过
68              if(generatedPortal == 1) continue;
69              items[row][col] = new Portal(true);
70              generatedPortal += 1;
71          } else if(rate < 40){//宝箱
72              //宝箱已经全部生成完毕，直接跳过
73              if(generatedTreasure == 39) continue;
74              items[row][col] = new Treasure(number);
75              generatedTreasure += 1;
76          } else {//怪物 初级：中级：高级：究级 = 18:12:6:3
77              int num = Tools.getRandomNumber(39);

```

```

77         if(num < 3){//究级怪物
78             //究级怪物已经全部生成完毕，直接跳过
79             if(generatedMonster4 == 3) continue;
80             items[row][col] = new Moltres(number);
81             generatedMonster4 += 1;
82         } else if(num < 9){//高级怪物
83             //高级怪物已经全部生成完毕，直接跳过
84             if(generatedMonster3 == 6) continue;
85             items[row][col] = new Ramoraid(number);
86             generatedMonster3 += 1;
87         } else if(num < 21){//中级怪物
88             //中级怪物已经全部生成完毕，直接跳过
89             if(generatedMonster2 == 12) continue;
90             items[row][col] = new CattleMonster(number);
91             generatedMonster2 += 1;
92         } else { //初级怪物
93             //初级怪物已经全部生成完毕，直接跳过
94             if(generatedMonster1 == 18) continue;
95             items[row][col] = new Mamoswine(number);
96             generatedMonster1 += 1;
97         }
98     }
99 }
100
101 }
102
103 /**
104  * 展示地图
105  */
106 public void show(){
107     for(int i=0; i<items.length; i++){
108         string line1 = "", line2 = "";
109         for(int j=0; j<items[i].length; j++){
110             if(i == 0){//第一行
111                 if(j == 0){//第一列
112                     line1 += "┌───";
113                     line2 += "| " + items[i][j].getItemInformation() +
114 " ";
115                 } else if(j == items[i].length-1){//最后一列
116                     line1 += "└───";
117                     line2 += "| " + items[i][j].getItemInformation() +
118 " |";
119                 } else {
120                     line1 += "├───";
121                     line2 += "| " + items[i][j].getItemInformation() +
122 " ";
123                 }
124             } else {
125                 if(j == 0){//第一列
126                     line1 += "┌───";
127                     line2 += "| " + items[i][j].getItemInformation() +
128 " ";
129                 } else if(j == items[i].length-1){//最后一列
130                     line1 += "└───";
131                     line2 += "| " + items[i][j].getItemInformation() +
132 " |";
133                 } else {
134                     line1 += "├───";
135                     line2 += "| " + items[i][j].getItemInformation() +
136 " ";
137                 }
138             }
139         }
140         System.out.println(line1);
141         System.out.println(line2);
142     }
143 }

```

```

130         line2 += "|" + items[i][j].getItemInformation() +
    " ";
131     }
132 }
133 }
134 System.out.println(line1);
135 System.out.println(line2);
136 }
137 String lastLine = ""; //最后一行网格线
138 for(int i=0; i<items[0].length; i++){
139     if(i==0){ //第一列
140         lastLine += "┌───";
141     } else if(i == items[0].length -1){ //最后一列
142         lastLine += "└───";
143     } else {
144         lastLine += "│───";
145     }
146 }
147 System.out.println(lastLine);
148 }
149 }
150
151
152 package com.cyx.pokemon.item.monster;
153
154 import com.cyx.pokemon.item.Item;
155 import com.cyx.pokemon.item.pokemon.Pokemon;
156 import com.cyx.pokemon.util.Tools;
157
158 /**
159  * 怪物
160  */
161 public abstract class Monster extends Item {
162
163     /**
164      * 攻击力
165      */
166     protected int attack;
167
168     /**
169      * 防御力
170      */
171     protected int defense;
172
173     /**
174      * 生命值
175      */
176     protected int health;
177
178     /**
179      * 怪物当前血量
180      */
181     protected int currentHealth;
182
183     public Monster(String name, int levelNumber) {
184         super(name, levelNumber);
185     }
186
187     public int getDefense() {
188         return defense;
189     }

```

```

187     }
188
189     public int getCurrentHealth() {
190         return currentHealth;
191     }
192
193     public void setCurrentHealth(int currentHealth) {
194         this.currentHealth = currentHealth;
195     }
196
197     /**
198      * 攻击宠物小精灵
199      * @param pokemon 宠物小精灵
200      */
201     public void attackPokemon(Pokemon pokemon){
202         int minusHealth = this.attack * this.attack /
pokemon.getDefense();
203         if(minusHealth == 0) { //伤害为0，需要调整
204             minusHealth = 1; //调整伤害为1点
205         } else if(minusHealth > pokemon.getCurrentHealth()) { //如果伤害比宠物
小精灵当前血量还要高
206             minusHealth = pokemon.getCurrentHealth(); //伤害就应该等于宠物小精
灵当前血量
207         }
208         //剩余血量
209         int restHealth = pokemon.getCurrentHealth() - minusHealth;
210         pokemon.setCurrentHealth(restHealth);
211         System.err.println(name + "对" + pokemon.getName() + "发动攻击，造成
了" + minusHealth + "伤害");
212     }
213
214     /**
215      * 怪物掉落装备
216      * @return
217      */
218     public Item drop(){
219         return Tools.getRandomItem(levelNumber);
220     }
221 }
222
223 package com.cyx.pokemon.item.monster;
224
225 import com.cyx.pokemon.util.Tools;
226
227 /**
228  * 火焰鸟
229  */
230 public class Moltres extends Monster{
231
232     public Moltres(int levelNumber) {
233         super("火焰鸟", levelNumber);
234         this.attack = Tools.getRandomNumber(80, 100, levelNumber);
235         this.defense = Tools.getRandomNumber(70, 90, levelNumber);
236         this.health = Tools.getRandomNumber(1400, 1800, levelNumber);
237         this.currentHealth = this.health;
238     }
239
240     @Override

```

```

241     public String getItemInformation() {
242         return discovery ? "D" : "■";
243     }
244 }
245
246 package com.cyx.pokemon.item.monster;
247
248 import com.cyx.pokemon.util.Tools;
249
250 /**
251  * 象牙猪
252  */
253 public class Mamoswine extends Monster {
254
255     public Mamoswine(int levelNumber) {
256         super("象牙猪", levelNumber);
257         this.attack = Tools.getRandomNumber(45, 55, levelNumber);
258         this.defense = Tools.getRandomNumber(35, 45, levelNumber);
259         this.health = Tools.getRandomNumber(600, 800, levelNumber);
260         this.currentHealth = this.health;
261     }
262
263     @Override
264     public String getItemInformation() {
265         return discovery ? "A" : "■";
266     }
267 }
268
269 package com.cyx.pokemon.item.monster;
270
271 import com.cyx.pokemon.util.Tools;
272
273 /**
274  * 牛魔怪
275  */
276 public class CattleMonster extends Monster{
277
278     public CattleMonster(int levelNumber) {
279         super("牛魔怪", levelNumber);
280         this.attack = Tools.getRandomNumber(50, 60, levelNumber);
281         this.defense = Tools.getRandomNumber(40, 50, levelNumber);
282         this.health = Tools.getRandomNumber(700, 900, levelNumber);
283         this.currentHealth = this.health;
284     }
285
286     @Override
287     public String getItemInformation() {
288         return discovery ? "B" : "■";
289     }
290 }
291
292 package com.cyx.pokemon.item.monster;
293
294 import com.cyx.pokemon.util.Tools;
295
296 /**
297  * 铁炮鱼
298  */

```



```

299 public class Ramoraid extends Monster{
300
301     public Ramoraid(int levelNumber) {
302         super("铁炮鱼", levelNumber);
303         this.attack = Tools.getRandomNumber(60, 70, levelNumber);
304         this.defense = Tools.getRandomNumber(50, 60, levelNumber);
305         this.health = Tools.getRandomNumber(900, 1100, levelNumber);
306         this.currentHealth = this.health;
307     }
308
309     @Override
310     public String getItemInformation() {
311         return discovery ? "C" : "■";
312     }
313 }

```

## 11. 冒险家分析

冒险家携带背包闯关，背包中可以容纳装备、药品以及小精灵，默认有10个药品、1个妙蛙种子

```

1  package com.cyx.pokemon;
2
3  import com.cyx.pokemon.item.HP;
4  import com.cyx.pokemon.item.Item;
5  import com.cyx.pokemon.item.equipment.Equipment;
6  import com.cyx.pokemon.item.pokemon.Bulbasaur;
7  import com.cyx.pokemon.item.pokemon.Pokemon;
8
9  /**
10   * 冒险家
11   */
12  public class Adventurer implements DisplayItem{
13      /**
14       * 装备背包
15       */
16      private Equipment[] equipments = {};
17      /**
18       * 药品背包
19       */
20      private HP[] medicines = {
21          new HP(1, 10)
22      };
23      /**
24       * 宠物背包
25       */
26      private Pokemon[] pokemons = {
27          new Bulbasaur()
28      };
29      /**
30       * 总背包
31       */
32      private Item[][] packageItems = {
33          equipments,
34          medicines,
35          pokemons
36      };
37

```

```

38     @Override
39     public String getItemInformation() {
40         return "♀";
41     }
42 }

```

冒险家开始闯关时会进入关卡地图，因此需要在关卡地图 `LevelMap` 中添加冒险家

```

1  /**
2   * 添加冒险家
3   * @param adventurer 冒险家
4   */
5  public void addAdventurer(Adventurer adventurer){
6      if(number == 1){//第一关
7          items[0][0] = adventurer;
8      } else {
9          items[0][1] = adventurer;
10     }
11 }

```

完善冒险家 `Adventurer` 闯关方法

```

1  /**
2   * 开始闯关
3   */
4  public void start(){
5      Level level = new Level(null, 1, null);
6      LevelMap map = level.getMap();
7      //冒险家进入地图
8      map.addAdventurer(this);
9      map.show();
10 }

```

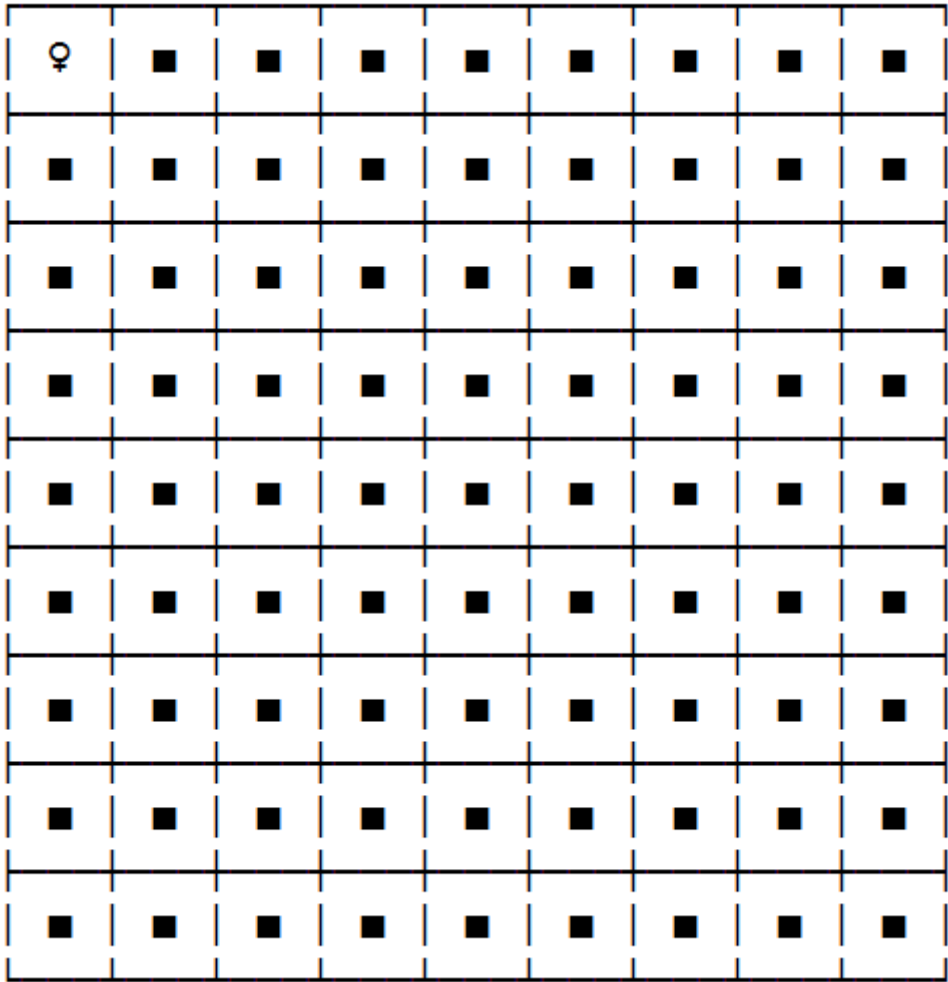
编写启动类，冒险家开始闯关，检测显示是否正常

```

1  package com.cyx.pokemon;
2
3  /**
4   * 启动类
5   */
6  public class Launcher {
7
8      public static void main(String[] args) {
9          Adventurer adventurer = new Adventurer();
10         adventurer.start();
11     }
12 }

```

执行结果如图所示则为正常



冒险家闯关时可以使用w(上)、A(左)、S(下)、D(右)四个键在地图上移动，移动时需要先探索，如果遇到怪物，则需要先击败怪物，才能移动；如果遇到宝箱，需要打开后才能移动。如果遇到传送门，可以选择直接移动。因此需要在冒险家 `Adventurer` 中添加按方向探索 and 移动的方法

冒险家探索和移动时都需要冒险家在地图上的位置，因此，在地图中应该记录冒险家的位置。冒险家移动后，能够发现地图上的物品，因此，还需要在地图中添加获取给定方向位置物品的方法。

#### 完善冒险家 `Adventurer` 探索方法

冒险家完成探索后，可能移动至探索位置，移动后，冒险家原来的位置再没有物品。因此，地图中需要添加冒险家位置变更的方法。

#### 完善冒险家 `Adventurer` 移动方法

冒险家需要从控制台输入移动方向，移动可以反复执行，因此需要完善开始闯关的方法

#### 宝箱处理

```

1 package com.cyx.pokemon.level;
2
3 import com.cyx.pokemon.Adventurer;
4 import com.cyx.pokemon.DisplayItem;
5 import com.cyx.pokemon.item.Item;
6 import com.cyx.pokemon.item.Portal;
7 import com.cyx.pokemon.item.Treasure;
8 import com.cyx.pokemon.item.monster.CattleMonster;
9 import com.cyx.pokemon.item.monster.Mamoswine;
10 import com.cyx.pokemon.item.monster.Moltres;
11 import com.cyx.pokemon.item.monster.Ramoraid;
12 import com.cyx.pokemon.util.Tools;
13
14 /**
15  * 关卡地图
16  */
17 public class LevelMap {
18     /**
19      * 关卡编号
20      */
21     private int number;
22     /**
23      * 地图上的物品: 9x9
24      */
25     private final DisplayItem[][] items = new DisplayItem[9][9];
26     /**
27      * 记录冒险家在地图中的位置
28      */
29     private int currentRow, currentCol;
30
31     public LevelMap(int number) {
32         this.number = number;
33         generate();
34     }
35
36     /**
37      * 生成地图=> 宝箱: 怪物: 传送门 = 39:39:1
38      * 第一个位置和第二个位置不能使用
39      */
40     private void generate(){
41         if(number == 1){//第一关卡
42             //第二个位置为初级怪物象牙猪
43             items[0][1] = new Mamoswine(number);
44             items[0][0] = new Mamoswine(number);
45         } else {//其他关卡
46             //第一个位置为返回上一层的传送门
47             items[0][0] = new Portal(false);
48             items[0][1] = new Portal(false);
49         }
50         //记录生成的宝箱数量
51         int generatedTreasure = 0;
52         //记录生成的怪物数量
53         int generatedMonster1 = 0;//记录生成的初级怪物数量
54         int generatedMonster2 = 0;//记录生成的中级怪物数量
55         int generatedMonster3 = 0;//记录生成的高级怪物数量
56         int generatedMonster4 = 0;//记录生成的究级怪物数量
57         //记录生成的宝箱数量
58         int generatedPortal = 0;

```

```

59         while (generatedTreasure < 39
60             || (generatedMonster1 + generatedMonster2 +
generatedMonster3 + generatedMonster4) < 39
61             || generatedPortal == 0){
62             //获取随机坐标
63             int index = Tools.getRandomNumber(2, 81);
64             //计算行和列
65             int row = index / items[0].length;
66             int col = index % items[0].length;
67             //目标位置已经有物品存在
68             if(items[row][col] != null) continue;
69             //获取一个随机数
70             int rate = Tools.getRandomNumber(79);
71             if(rate == 0){//传送门
72                 //传送门已经生成了，直接跳过
73                 if(generatedPortal == 1) continue;
74                 items[row][col] = new Portal(true);
75                 generatedPortal += 1;
76             } else if(rate < 40){//宝箱
77                 //宝箱已经全部生成完毕，直接跳过
78                 if(generatedTreasure == 39) continue;
79                 items[row][col] = new Treasure(number);
80                 generatedTreasure += 1;
81             } else {//怪物 初级：中级：高级：究级 = 18:12:6:3
82                 int num = Tools.getRandomNumber(39);
83                 if(num < 3){//究级怪物
84                     //究级怪物已经全部生成完毕，直接跳过
85                     if(generatedMonster4 == 3) continue;
86                     items[row][col] = new Moltres(number);
87                     generatedMonster4 += 1;
88                 } else if(num < 9){//高级怪物
89                     //高级怪物已经全部生成完毕，直接跳过
90                     if(generatedMonster3 == 6) continue;
91                     items[row][col] = new Ramoraid(number);
92                     generatedMonster3 += 1;
93                 } else if(num < 21){//中级怪物
94                     //中级怪物已经全部生成完毕，直接跳过
95                     if(generatedMonster2 == 12) continue;
96                     items[row][col] = new CattleMonster(number);
97                     generatedMonster2 += 1;
98                 } else {//初级怪物
99                     //初级怪物已经全部生成完毕，直接跳过
100                     if(generatedMonster1 == 18) continue;
101                     items[row][col] = new Mamoswine(number);
102                     generatedMonster1 += 1;
103                 }
104             }
105         }
106
107     }
108
109     /**
110     * 获取给定方向位置的物品信息
111     * @param direct 方向
112     * @return
113     */
114     public DisplayItem getPositionItem(char direct){
115         int targetRow = currentRow, targetCol = currentCol;

```

```

116         switch (direct){
117             case 'w': //向上
118                 if(targetRow == 0){
119                     return null;
120                 }
121                 targetRow -= 1;
122                 break;
123             case 'A'://向左
124                 if(targetCol == 0){
125                     return null;
126                 }
127                 targetCol -= 1;
128                 break;
129             case 's'://向下
130                 if(targetRow == items.length - 1){
131                     return null;
132                 }
133                 targetRow += 1;
134                 break;
135             case 'D'://向右
136                 if(targetCol == items[currentRow].length -1){
137                     return null;
138                 }
139                 targetCol += 1;
140                 break;
141         }
142         return items[targetRow][targetCol];
143     }
144
145     /**
146     * 向给定方向移动冒险家的位置
147     * @param direct
148     */
149     public void move(char direct){
150         int oldRow = currentRow,oldCol = currentCol;
151         //获取冒险家
152         DisplayItem adventurer = items[oldRow][oldCol];
153         switch (direct){
154             case 'w': //向上
155                 if(currentRow == 0){
156                     System.err.println("非法移动");
157                     Tools.lazy(300L);
158                     return;
159                 }
160                 currentRow -= 1;
161                 break;
162             case 'A'://向左
163                 if(currentCol == 0){
164                     System.err.println("非法移动");
165                     Tools.lazy(300L);
166                     return;
167                 }
168                 currentCol -= 1;
169                 break;
170             case 's'://向下
171                 if(currentRow == items.length - 1){
172                     System.err.println("非法移动");
173                     Tools.lazy(300L);

```

```

174         return;
175     }
176     currentRow += 1;
177     break;
178     case 'D': //向右
179         if(currentCol == items[currentRow].length - 1){
180             System.err.println("非法移动");
181             Tools.lazy(300L);
182             return;
183         }
184         currentCol += 1;
185         break;
186     }
187     //冒险家新的位置
188     items[currentRow][currentCol] = adventurer;
189     //原来的位置就不存在物品了
190     items[oldRow][oldCol] = null;
191 }
192
193 /**
194  * 添加冒险家
195  * @param adventurer 冒险家
196  */
197 public void addAdventurer(Adventurer adventurer){
198     currentRow = 0;
199     if(number == 1){ //第一关
200         currentCol = 0;
201     } else {
202         currentCol = 1;
203     }
204     items[currentRow][currentCol] = adventurer;
205 }
206
207 /**
208  * 展示地图
209  */
210 public void show(){
211     System.out.println("宠物小精灵第" + number + "关: ");
212     for(int i=0; i<items.length; i++){
213         String line1 = "", line2 = "";
214         for(int j=0; j<items[i].length; j++){
215             String info = " ";
216             if(items[i][j] != null){
217                 info = items[i][j].getItemInformation();
218             }
219             if(i == 0){ //第一行
220                 if(j == 0){ //第一列
221                     line1 += "┌";
222                     line2 += "| " + info + " ";
223                 } else if(j == items[i].length-1){ //最后一列
224                     line1 += "└";
225                     line2 += "| " + info + " |";
226                 } else {
227                     line1 += "├";
228                     line2 += "| " + info + " ";
229                 }
230             } else {
231                 if(j == 0){ //第一列

```

```

232         line1 += "├──";
233         line2 += "| " + info + " ";
234     } else if(j == items[i].length-1){//最后一列
235         line1 += "├──|";
236         line2 += "| " + info + " |";
237     } else {
238         line1 += "├──";
239         line2 += "| " + info + " ";
240     }
241 }
242 }
243 System.out.println(line1);
244 System.out.println(line2);
245 }
246 String lastLine = "";//最后一行网格线
247 for(int i=0;i<items[0].length; i++){
248     if(i==0){//第一列
249         lastLine += "┌──";
250     } else if(i == items[0].length -1){//最后一列
251         lastLine += "┌──|";
252     } else {
253         lastLine += "┌──";
254     }
255 }
256 System.out.println(lastLine);
257 }
258 }
259
260 package com.cyx.pokemon;
261
262 import com.cyx.pokemon.item.HP;
263 import com.cyx.pokemon.item.Item;
264 import com.cyx.pokemon.item.Portal;
265 import com.cyx.pokemon.item.Treasure;
266 import com.cyx.pokemon.item.equipment.Equipment;
267 import com.cyx.pokemon.item.monster.Monster;
268 import com.cyx.pokemon.item.pokemon.Bulbasaur;
269 import com.cyx.pokemon.item.pokemon.Pokemon;
270 import com.cyx.pokemon.level.Level;
271 import com.cyx.pokemon.level.LevelMap;
272 import com.cyx.pokemon.util.Tools;
273
274 import java.util.Arrays;
275
276 /**
277  * 冒险家
278  */
279 public class Adventurer implements DisplayItem{
280     /**
281      * 装备背包
282      */
283     private Equipment[] equipments = {};
284     /**
285      * 药品背包
286      */
287     private HP[] medicines = {
288         new HP(1, 10)
289     };

```



```

290     /**
291      * 宠物背包
292      */
293     private Pokemon[] pokemons = {
294         new Bulbasaur()
295     };
296     /**
297      * 总背包
298      */
299     private Item[][] packageItems = {
300         equipments,
301         medicines,
302         pokemons
303     };
304
305     private Level currentLevel;
306
307     /**
308      * 开始闯关
309      */
310     public void start(){
311         currentLevel = new Level(null, 1, null);
312         LevelMap map = currentLevel.getMap();
313         //冒险家进入地图
314         map.addAdventurer(this);
315         while (true){
316             currentLevel.getMap().show();
317             System.out.println("请选择移动方向: W(上)、A(左)、S(下)、D(右)、E(退出)");
318             char direct = Tools.getInputChar();
319             if(direct == 'E'){//退出
320                 System.out.println("确定要退出吗? Y/N");
321                 char quit = Tools.getInputChar();
322                 if(Character.toUpperCase(quit) == 'Y'){
323                     System.out.println("感谢使用宠物小精灵闯关");
324                     break;
325                 }
326             } else {
327                 Item item = discovery(direct);
328                 if(item != null){
329                     //物品被发现
330                     item.setDiscovery(true);
331                     currentLevel.getMap().show();
332                 }
333                 if(item instanceof Treasure){//宝箱
334                     processTreasure((Treasure) item, direct);
335                 } else if(item instanceof Monster){//怪物
336                     processMonster((Monster) item, direct);
337                 } else if(item instanceof Portal){//传送门
338                     System.out.println("发现传送门, 是否通过? Y/N");
339                     char pass = Tools.getInputChar();
340                     if(Character.toUpperCase(pass) == 'Y'){
341                         if(((Portal) item).isNext()){//通往下一关卡的传送门
342                             //获取当前关卡的下一关卡
343                             Level nextLevel = currentLevel.getNextLevel();
344                             if(nextLevel == null){//下一关卡为空, 则需要创建
345                                 nextLevel = new Level(currentLevel,
currentLevel.getNumber() + 1, null);

```

```

346         //将冒险家加载至地图中
347         nextLevel.getMap().addAdventurer(this);
348         //当前关卡的下一关卡即为新创建的关卡
349         currentLevel.setNextLevel(nextLevel);
350     }
351     //经过传送门后，下一关卡即为当前关卡
352     currentLevel = nextLevel;
353 } else { //通往上一关卡的传送门
354     Level prevLevel = currentLevel.getPrevLevel();
355     if(prevLevel == null){
356         System.out.println("非法操作");
357     } else {
358         currentLevel = prevLevel;
359     }
360 }
361 }
362 } else { //其他情况
363     move(direct);
364 }
365 }
366 }
367
368 }
369
370 /**
371  * 处理怪物
372  * @param monster 怪物
373  * @param direct 方向
374  */
375 private void processMonster(Monster monster, char direct){
376     System.out.println("发现 " + monster.getName() + "，是否清除？ Y/N");
377     char clear = Tools.getInputChar();
378     if(Character.toUpperCase(clear) == 'Y'){
379         for(int i=0; i<pokemons.length; i++){
380             System.out.println((i+1) + "\t" +
pokemons[i].getItemInformation());
381         }
382         System.out.println("请选择出战宠物小精灵：");
383         int number = Tools.getInputNumber(1, pokemons.length);
384         Pokemon pokemon = pokemons[number - 1];
385         while (monster.getCurrentHealth() > 0 &&
pokemon.getCurrentHealth() > 0){
386             //获取宠物小精灵的剩余生命值的比例
387             double rate = pokemon.getHealthPercent();
388             if(rate < 0.5){ //生命值低于50%，询问是否使用药品
389                 System.out.println(pokemon.getName() + "生命值低于50%，
是否使用药品？ Y/N");
390                 char eatHp = Tools.getInputChar();
391                 if(Character.toUpperCase(eatHp) == 'Y'){
392                     HP hp =
getCurrentLevelHP(currentLevel.getNumber());
393                     if(hp == null){
394                         System.out.println("背包中没有可用药品，请探索其他地
图");
395                     } else {
396                         //如果药品可以被销毁，说明没有可用数量
397                         if(hp.canDestroy()){
398                             int index = -1;

```

```

399         for(int i=0; i<medicines.length; i++){
400             if(hp.getLevelNumber() ==
medicines[i].getLevelNumber()){
401                 index = i;
402                 break;
403             }
404         }
405         System.arraycopy(medicines, index+1,
medicines, index, medicines.length - index -1);
406         System.out.println("药品已经使用完毕");
407     } else {
408         int health = hp.use();
409         pokemon.setCurrentHealth(
pokemon.getCurrentHealth() + health);
410     }
411 }
412 }
413 }
414 Tools.lazy(300L);
415 pokemon.attackMonster(monster);
416 Tools.lazy(300L);
417 monster.attackPokemon(pokemon);
418 Tools.lazy(300L);
419 }
420 //怪物已被击败
421 if(monster.getCurrentHealth() == 0){
422     System.out.println("怪物已被击败");
423     //怪物掉落物品
424     Item dropItem = monster.drop();
425     //展示获取的物品信息
426     System.out.println("怪物已被击败，掉落" +
dropItem.getItemInformation());
427     processItem(dropItem);
428     //怪物被击败后
429     move(direct);
430 } else { //宠物小精灵被击败
431     monster.resume(); //怪物回血
432     System.out.println(pokemon.getName() + "已被击败");
433 }
434 }
435 }
436
437 /**
438  * 获取当前关卡使用的药品，如果当前关卡的药品已经使用完，那么可以使用上一关卡的药品，依次类推
439  * @param levelNumber
440  * @return
441  */
442 private HP getCurrentLevelHP(int levelNumber){
443     if(levelNumber == 0) return null;
444     HP hp = null;
445     for(int i=0; i<medicines.length; i++){
446         if(medicines[i].getLevelNumber() == levelNumber){
447             hp = medicines[i];
448             break;
449         }
450     }
451     if(hp == null){

```

```

452         return getCurrentLevelHP(levelNumber - 1);
453     } else {
454         return hp;
455     }
456 }
457
458 /**
459  * 处理获得物品
460  * @param item
461  */
462 private void processItem(Item item){
463
464     if(item instanceof HP){//药品
465         for(HP hp: medicines){
466             if(hp.getLevelNumber() == item.getLevelNumber()){
467                 hp.addCount(((HP) item).getCount());
468                 break;
469             }
470         }
471     } else if(item instanceof Equipment){//装备
472         System.out.println("发现新的装备，是否给宠物小精灵更换？ Y/N");
473         char change = Tools.getInputChar();
474         if(Character.toUpperCase(change) == 'Y'){
475             Equipment old = null;
476             for(Pokemon pokemon: pokemons){
477                 //小精灵更换装备
478                 old = pokemon.changeEquipment((Equipment) item);
479                 //如果换下来的装备为空，说明后面的小精灵不需要再看
480                 if(old == null) break;
481             }
482             //如果换下来的旧装备不为空，直接放入背包中
483             if(old != null){
484                 equipments = Arrays.copyOf(equipments,
485 equipments.length + 1);
486                 equipments[equipments.length - 1] = old;
487             }
488         } else{//宠物小精灵
489             int index = -1;
490             for(int i=0; i<pokemons.length; i++){
491                 if(item.getClass() == pokemons[i].getClass()){
492                     index = i;
493                     break;
494                 }
495             }
496             //不存在同类型宠物小精灵
497             if(index == -1){
498                 pokemons = Arrays.copyOf(pokemons, pokemons.length + 1);
499                 pokemons[pokemons.length - 1] = (Pokemon) item;
500             } else{//存在同类型宠物小精灵
501                 System.out.println("发现可融合宠物小精灵，是否融合？ Y/N");
502                 char merge = Tools.getInputChar();
503                 if(Character.toUpperCase(merge) == 'Y'){
504                     pokemons[index].merge((Pokemon) item);
505                 } else{//不融合，直接放入背包
506                     pokemons = Arrays.copyOf(pokemons, pokemons.length +
507 1);
508                     pokemons[pokemons.length - 1] = (Pokemon) item;

```

```

508     }
509     }
510 }
511 }
512
513 /**
514  * 处理宝箱
515  * @param treasure 宝箱
516  */
517 private void processTreasure(Treasure treasure, char direct){
518     System.out.println("发现宝箱，是否打开？ Y/N");
519     char open = Tools.getInputChar();
520     if(Character.toUpperCase(open) == 'Y'){
521         //开启宝箱获得一个物品
522         Item item = treasure.open();
523         //展示获取的物品信息
524         System.out.println("获得" + item.getItemInformation());
525         processItem(item);
526         //宝箱处理后，冒险家移动至宝箱的位置
527         move(direct);
528     }
529 }
530
531 /**
532  * 探索给定方向地图位置
533  * @param direct 方向
534  * @return
535  */
536 private Item discovery(char direct){
537     return (Item)
currentLevel.getMap().getPositionItem(Character.toUpperCase(direct));
538 }
539 /**
540  * 向给定方向地图位置移动
541  */
542 private void move(char direct){
543     currentLevel.getMap().move(Character.toUpperCase(direct));
544 }
545
546
547 @Override
548 public String getItemInformation() {
549     return "♀";
550 }
551 }
552
553 package com.cyx.pokemon.item.monster;
554
555 import com.cyx.pokemon.item.Item;
556 import com.cyx.pokemon.item.pokemon.Pokemon;
557 import com.cyx.pokemon.util.Tools;
558
559 /**
560  * 怪物
561  */
562 public abstract class Monster extends Item {
563
564     /**

```

```

565     * 攻击力
566     */
567     protected int attack;
568     /**
569     * 防御力
570     */
571     protected int defense;
572     /**
573     * 生命值
574     */
575     protected int health;
576     /**
577     * 怪物当前血量
578     */
579     protected int currentHealth;
580
581
582     public Monster(String name, int levelNumber) {
583         super(name, levelNumber);
584     }
585
586     public int getDefense() {
587         return defense;
588     }
589
590     public int getCurrentHealth() {
591         return currentHealth;
592     }
593
594     public void setCurrentHealth(int currentHealth) {
595         this.currentHealth = currentHealth;
596     }
597
598     /**
599     * 怪物恢复
600     */
601     public void resume(){
602         currentHealth = health;
603     }
604
605     /**
606     * 攻击宠物小精灵
607     * @param pokemon 宠物小精灵
608     */
609     public void attackPokemon(Pokemon pokemon){
610         int minusHealth = this.attack * this.attack /
pokemon.getDefense();
611         if(minusHealth == 0) { //伤害为0, 需要调整
612             minusHealth = 1; //调整伤害为1点
613         } else if(minusHealth > pokemon.getCurrentHealth()) { //如果伤害比宠物
小精灵当前血量还要高
614             minusHealth = pokemon.getCurrentHealth(); //伤害就应该等于宠物小精
灵当前血量
615         }
616         //剩余血量
617         int restHealth = pokemon.getCurrentHealth() - minusHealth;
618         pokemon.setCurrentHealth(restHealth);

```

```

619         System.err.println(name + "对" + pokemon.getName() + "发动攻击，造成
了" + minusHealth + "伤害");
620     }
621
622     /**
623      * 怪物掉落装备
624      * @return
625      */
626     public Item drop(){
627         return Tools.getRandomItem(levelNumber);
628     }
629 }
630 package com.cyx.pokemon.item;
631
632 import com.cyx.pokemon.DisplayItem;
633
634 /**
635  * 传送门
636  */
637 public class Portal extends Item {
638     /**
639      * 是否是通往下一关卡的传送门
640      */
641     private boolean next;
642
643     public Portal(boolean next) {
644         super("传送门");
645         this.next = next;
646     }
647
648     public boolean isNext() {
649         return next;
650     }
651
652     @Override
653     public String getItemInformation() {
654         if(discovery){
655             return next ? "→" : "←";
656         }
657         return "■";
658     }
659 }

```