Jumper

程序说明

version 1.0.0

修订历史

日期	版本	作者	描述
2015/8/21	试作版	江剑锋	地图初始时随机出现 1000 朵花，20 只 jumper。Jumper 每跳跃 2 次会随机转向，Jumper 之间不可以消灭，Jumper 可踩死 Flower。
2015/8/21	试作版	林在华	地图初始化时有若干只颜色不同的 Jumper，设置了若干定点障碍物。Jumper 不停跳跃，无法跳跃时右转直到可跳跃。
2015/8/21	试作版	苗佳欣	地图初始化时有若干只 bug 及 Jumper，设置若干定点障碍物。Bug 会沿路制造花朵，Jumper 不停跳跃，无法跳跃时右转直到可跳跃。
2015/8/21	试作版	张国伟	地图初始化时有若干只 Jumper，设置若干定点障碍物。Jumper 具有 Bug 以及跳跃特性，正常情况为 move 并留下花朵，遇到障碍物时才进行跳跃，无法移动和跳跃时右转直到可动。
2015/8/21	1.0.0	张国伟	此版本是对 0.9.5 张国伟版本的更改。向 TA 咨询后，剔除 Jumper 的 Bug 特性，并且修改了地图，同时封装了一个测试的辅助工具类。虽然最后的测试还是交给 JunitTest，但是利用封装的工具类简化了 JunitTest 的代码编写，降低了 JunitTest 编写测试样例时的出错率，同时便于修改测试样例。此版本也是本次团队作业的提交版本。

目录

1 程序功能

功能	描述
跳跃	Jumper 会前跳 2 步
转向	Jumper 在遇到边界时、跳跃落点位置有石头或 Jumper 时都会进行转向
进食	若跳跃的落点位置有花朵，就吃掉那朵花
自毁	Jumper 若被强制地丢出界外，会自动毁灭
友谊	Jumper 之间不会相互毁灭

2 实现过程

复写 act()函数，使动作的优先级为 jump，turn

```
public void act() {
    if (canJump()) {
        jump();
    } else {
        turn();
    }
}
```

canJump 函数

- **获得两步后的落点位置**

  Location next =

  loc.getAdjacentLocation(getDirection()).getAdjacentLocation(getDirection())

- **判断是否 next 在界外**

  调用 grid 的 isValid 来判断是否 next 在界外。

  1、 如果在界外，canJump 返回 false，

  2、 否则，就获取该位置的实体 neighbor。

  i. 当 neighbor 为空或者是 Flower 类的时候，canJump 返回 true，从而实现了“进食”功能，Jumper 会吃掉 Flower。

  ii. 否则，返回 false。从而实现了 “友谊”功能，Jumper 直接不会相互“踩死”。

jump 函数

- **获得两步后的落点位置（同 canJump）**
- **判断是否 next 在界外**

  调用 grid 的 isValid 来判断是否 next 在界外。

  1、如果在界外，就调用 removeSelfFromGrid()消除 jumper，从而实现了“自毁”功能（注意：此条件正常运行时不会发生，详情看 act()函数）。

  2、否则，说明 Jumper 在界内，就调用 moveTo(next)，使得 Jumper 直接到达两步后的位置，从而实现了“跳跃”功能。

turn 函数

- **转向**

  调用 setDirection(getDirection() + Location.HALF_RIGHT)，实现右转功能。

3 总结体会

第一次团队作业，也是团队第一次四人交流。我们小组在大约 9 点时，针对 Group Activity 上的每个问题进行了深入思考和讨论。发现对某些问题实在争执不下，比方说

Jumper 能不能 move，Jumper 的跳跃位置是花朵时该如何，障碍物出现的时机等等。

经过我们的分析，我们认为第一次团队任务基本的代码实现比较简单，一个完整的 Jumper 代码量及难度应该跟 Bug 类的实现差不多，都只是实现移动的方法以及一两个辅助方法来决定下一步的操作。所以，为了节省时间，大家决定先快速地用几分钟做了一个试作版。经过对比，发现最终效果的差别并不是很大。虽然编写代码前大家的想法有分歧，讨论完之后也还是存在分歧，但是事实上我们整个团队的思路在讨论后已经渐渐统一，作品的最终效果基本成型。因此，大家都觉得应该更好地体会此次团队作业，而不仅仅是体会打代码的乐趣。

我们最后决定采用国伟的试作版，因为当时国伟的试作版已经开始检验不同的场景。团队成员先 review 了一下试作版的代码，觉得没有大问题后，对 Ant 的编写，Junit 的编写，测试报告，程序说明报告以及试作版的修改进行了分工。其中 Ant 由林在华同学负责，测试报告由苗佳欣同学负责，程序说明报告由江剑锋同学负责，团队作业提交版的最终修改以及 JunitTest 的辅助测试类的封装由张国伟同学负责。林同学结合前两天的自学内容，编写 Ant 文件，并且利用 Ant 文件打开 Junit 测试，把 Junit 测试的最终结果以 html 的形式展现；江同学用自然语言对程序的关键代码进行描述，综合几个成员的工作情况以及心得体会；苗同学针对各种情景设定测试样例，并利用辅助类进行测试；张同学抽象出测试内容的一般性，封装了测试样例的辅助类。

总的来说，这次分工合理，大家各司其职，相辅相成，没有拖延工作，也没有相互推托工作，是一次非常愉快的团队合作经历。