测试报告
Jumper

修订历史

日期	版本	作者	描述
2014.8.7	V1	刘建安 邝媛媛	Jumper 的测试报告

1 测试计划

(1).判断如果 Jumper 前面第一个格子为空，而第二个格子不为空时，Jumper 该如何移动；

(2).判断如果 Jumper 要移动的目的格子在 grid 外时，Jumper 该如何移动；

(3).判断如果 Jumper 面临边界时，Jumper 该如何移动；

(4).判断如果有其他 Actor 占据着 Jumper 目的格子时，Jumper 该如何移动；

(5).判断如果 Jumper 和其他 Actor 相遇时，Jumper 该如何运动；

由于 Jumper 在 grid 内运动，途中可能会遇到边界或其他 Actor，而且当面临不同 Actor 时，Jumper 会有不同的运动。为了保证 Jumper 在面临不同的情况是否能够正确的运动，我们要分别测试 Jumper 面临上述情况时的运动情况，然后与期望值相比较，就可以判断 Jumper 是否正确运动了。

2 用例 1

Public void testRock() ;

测试当目的格子（前方第二格）被 Rock 占据着时，Jumper 是否会转向来避开 Rock.

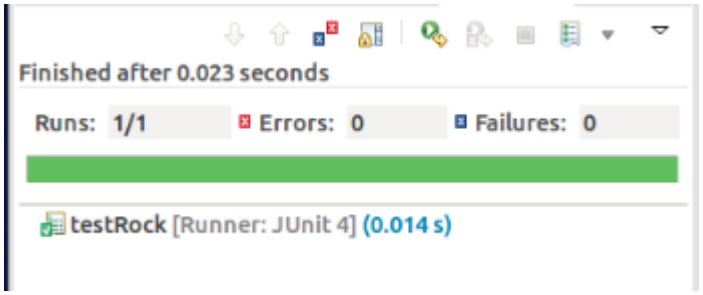
2.1 测试用例

```
@Test
public void testRock()
{
    //设置jumper的位置
    jumpLoc = new Location(8,2);
    //设置Rock的位置，该位置在jumper前面相隔一个位置处
    loc = new Location(6,2);
    //将jumper和Rock添加到world中
    world.add(loc, new Rock());
    world.add(jumpLoc, jumper);
    assertEquals(false, jumper.canMove());
}
```

设置 jump 位置为（8,2），Rock 位置为（6,2），把 jump 和 Rock 加到同一个 world 中，然后在调用 jump 的 canMove 方法来判断此时 jump 是否能够移动到（6,2）的位置。此时 canMove 函数返回值应该为 false。

2.2 测试结果

当目的位置被 Rock 占据着时，jumper 的 canMove 函数返回 false，与期望值相同，表示 Jumper 不能移动到 Rock 占据的位置。



2.3 结果分析

当目的位置（jumper 前方两格处）被 Rock 占据时，Jumper 是不能移动到该位置的，只能转向。

3 用例 2

Public void testEdge();

该用例用来测试当 Jumper 面向 grid 的边界时，Jumper 是否会移动到边界以外处。

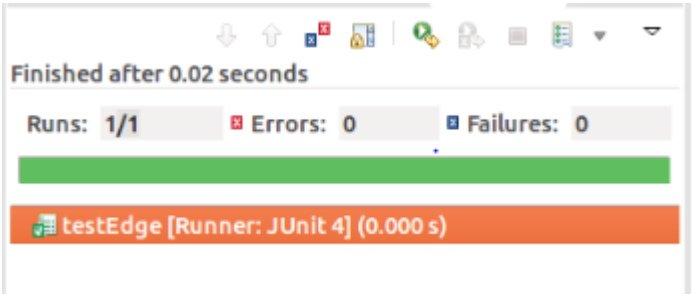
3.1 测试用例

```
@Test
public void testEdge()
{
    //设置jumper的位置，该位置在grid的边缘
    jumpLoc = new Location(1,3);
    world.add(jumpLoc,jumper);
    //判断canMove返回值是否与false相同
    assertEquals(false, jumper.canMove());
}
```

我们把 jumper 的位置设置为（1,3），此处位于 grid 的上方边界。然后我们在调用 jumper 的 canMove 方法，来判断 jumper 是否会移动到 grid 以外。

此时 canMove 方法的返回值应该为 false。

3.2 测试结果



Jumper 的 canMove 方法返回值为 false，与期望值相同。

3.3 结果分析

根据 canMove 返回值为 false，我们可以得到如下结论：Jumper 不能移动到 grid 以外的地方。

4.用例 3

Public void testFlower();

通过该用例来测试当目的位置被 Flower 占据时，Jumper 是否依然会移动到该位置，并移除 Flower.

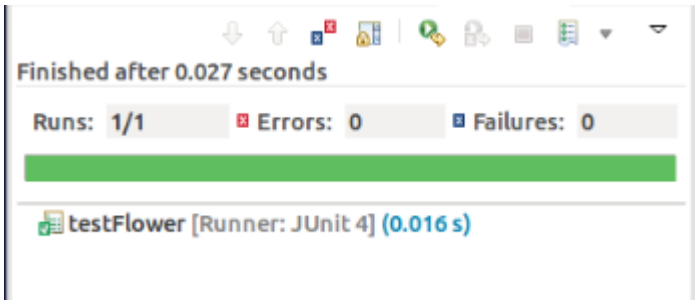
4.1 测试用例

```
@Test
public void testFlower()
{
    //设置jumper位置
    jumpLoc = new Location(8,2);
    //设置Flower位置，该位置在jumper位置前方相隔一个位置处
    loc = new Location(6,2);
    //将jumper和Flower添加到world中
    world.add(jumpLoc, jumper);
    world.add(loc, new Flower());
    assertEquals(true, jumper.canMove());
}
```

我们在（8,2）的位置放置一个 Jumper，在（6,2）的位置放置一个 Flower，然后调用 jumper 的 canMove 方法来判断当目的位置被 Flower 占据时，Jumper 是否能移动到该位置。

此时 canMove 方法的返回值应该为 true.

4.2 测试结果



canMove 函数返回值为 true,与期望值相同。

4.3 结果分析

根据测试结果，我们可以看出：当 Jumper 的目的位置被 Flower 占据时，Jumper 依然会移动到该位置，并 eat 掉该位置上的 Flower.

5.用例

Public void testBug();

通过该用例来测试当 Jumper 的目的位置被 Bug 占据时，Jumper 会如何移动。

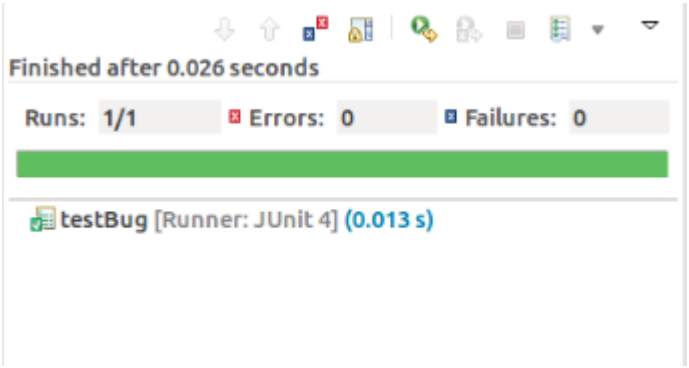
5.1 测试用例

```
@Test
public void testBug()
{
    //设置jumper位置
    jumpLoc = new Location(5, 5);
    //设置Bug位置，该位置在jumper位置前方相隔一个位置处
    loc = new Location(3, 5);
    //将jumper和Bug添加到world中
    world.add(jumpLoc, jumper);
    world.add(loc, new Bug());

    assertEquals(false, jumper.canMove());
}
```

我们在（5,5）位置上放置一个 Jumper，在（3,5）的位置上放置一个 Bug，然后调用 jumper 的 canMove 函数，测试此时 jumper 的移动方向。

5.2 测试结果



canMove 方法的返回值为 false，与期望值相同。

5.3 结果分析

通过测试我们得知，当 Jumper 的目的位置被 Bug 占据时，Jumper 不会移动到该位置。

6.用例

Public void testOverRock();

通过该方法来测试当 Jumper 前面为 Rock 时，Jumper 是否会越过

Rock，移动到目的位置去。

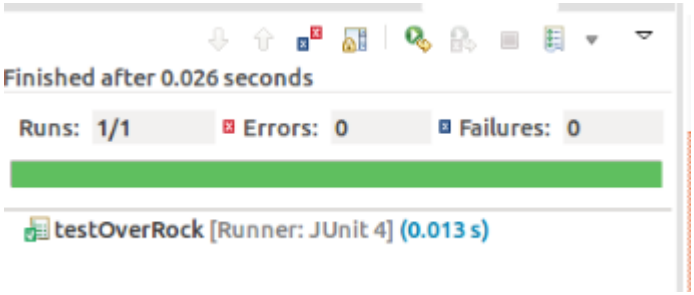
6.1 测试用例

```
@Test
public void testOverRock()
{
    //设置jumper位置
    jumpLoc = new Location(3, 2);
    //设置Rock位置，该位置在jumper位置前方
    loc = new Location(2, 2);
    //将jumper和Rock添加到world中
    world.add(jumpLoc, jumper);
    world.add(loc, new Rock());

    assertEquals(true, jumper.canMove());
}
```

我们在（3,2）位置上放置一个 Jumper，并在 Jumper 前方（2,2）位置上放置一个 Rock，然后调用 jumper 的 canMove 方法来判断 jumper 是否会越过 Rock 移动到 Rock 前方的目的位置上。

6.2 测试结果



canMove 方法返回值为 true，与期望值相同。

6.3 结果分析

通过测试结果，可以看出当 Jumper 被 Rock 挡住时，Jumper 会越过 Rock 移动到目的位置上。

7.用例

Public void testMove();

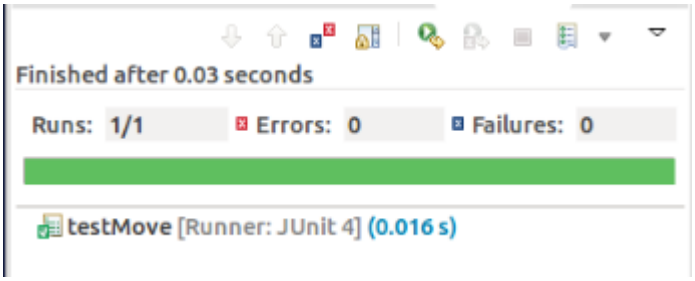
通过该方法来测试 Jumper 是否会向前移动两步

7.1 测试用例

```
@Test
public void testMove()
{
    //设置jumper位置
    jumpLoc = new Location(5,2);
    //将jumper添加到world中
    world.add(jumpLoc, jumper);
    //调用jumper的act函数，以移动jumper
    jumper.act();
    //期望目的位置
    Location newloc = new Location(3, 2);
    assertEquals(newloc, jumper.getLocation());
}
```

我们在（5,2）位置上放置一个 Jumper，然后调用 jumper 的 act 方法，再将移动之后的 jumper 的位置与期望位置（3,2）比较，看其是否相同。

7.2 测试结果



getLocation 方法返回的值与 newLoc 的值相同。

7.3 结果分析

由测试结果可得，Jumper 每次会向前移动两步。