People / Zhuofan Zhang / Home

CS231 Project 11

Added by Zhuofan Zhang, last edited by Zhuofan Zhang on Dec 12, 2015

Introduction

This time we're going to make the classic RPG game "Hunt the Wumpus" by implementing a grid-like graph structure, just like what I've done in last project...

General rules:

- The player plays the hunter, whose goal is to hunt the monster Wumpus.

- To lose: 1) fall into a pit; 2) walk straight into the Wumpus; 3) wasting the only arrow in hitting the wall and alarming the Wumpus.

- Hunter knows if a Wumpus is near or not by smelling. The smell is obvious within 2 block's range.

- Huge bats wander around; once spotting the hunter, they would take the hunter and drop it to a random place (including pits and the wumpus).

- The whole game's layout is a grid. Neighboring grids might not be connected.

In this project, the most basic requirement is to make the hunter, the wumpus, and a complete grid. Once the hunter shoots, one of them would die.

Solution

Several layers of classes: Cell, Vertex, Graph, Landscape, LandscapeDisplay, HuntTheWumpus (the game).

Cell class is an abstract class that has the basic format of all agents moving on the landscape.

Hunter and Wumpus are Cells. They have their locations and other properties stored.

Vertex is the unit that makes up the graph. One can connect at most 4 other vertices with the directions NORTH, WEST, EAST, and SOUTH. They also have draw methods that draws corresponding shapes with different images and colors based on different agents residing in the vertex.

Graph is, absolutely, the thing that contains all Vertices. Also, it can calculate the distance between any vertex and a specific vertex, so that we can calculate the shortest path. Here, we set the Wumpus' vertex as the "specific vertex", so that our hunter can know where is too near from the Wumpus.

Landscape contains all vertices and the hunter and the Wumpus. LandscapeDisplay is responsible for drawing everything in Landscape.

HuntTheWumpus is the game. It builds the graph, instructs the hunter, and entertains the user.

My illustrations are together with the extensions.

Extension

#1 Add user interface elements like a replay button that resets the game.

See demonstration [User Interface Demonstration]

As you can see, I also added the function that sees the whole map.

#2 Have your game generate interesting random graphs so you can replay a different game every time.

See [User Interface Demonstration] again.We have different maps each time.

Each time, the game would randomly change the properties of some vertices. For example, some might be cut off connections in all 4 directions, while some might just vertically or horizontally. In a few situations, the vertices are deliberately removed, while the connections through them are still kept, so you can see the "magic leap" in one of the demonstration videos. It's not a bug. It's just designed to play with the user.

NOTICE: To increase user experience, you can change line 191 & 192 in HuntTheWumpus.java into:

```
tempX = rr.nextInt(this.Col);
tempY = rr.nextInt(this.Row);
```

so that obstacles can appear in the lower right part of the game. I forgot this yesterday.

#3 Make the game visually interesting.

My visualization used emojis. The emoji pictures were downloaded from http://lechuck80.deviantart.com/art/Whatsapp-Emoji-Collection-357910493 . I used:

- arrow with heart as an extra bonus arrow

- mouse as bat

- poop as bat's sign

- pin as hunter

- multiple pins to represent multiple arrows that the hunter has

- Big Bastard Moon as pit

- water drops as pit's sign

- octopus as plain wumpus; dead smiley face as dead wumpus; the smiley face with a tongue out as wumpus with victory

- skeleton and red color as wumpus' sign

- hands as representations of aiming

Also, for each vertex, if visible, a square in the middle (not full) is the basic shape; for each connection with other vertices it has, it has a small rectangle on one of its edges representing the corresponding connection. a not-connected pair of vertices would have a continuous black line in between.

Please see the demonstrations below.

#4 Modify the rules so the game is more challenging.

As you see all those visualizations, you might wonder what are they:

- extra bonus arrow: appears with a small chance in each update; visible if in a visible vertex; can be picked up by the hunter to increase the amount of arrow by 1.

Use: The hunter only has one chance to shoot, because if the wumpus is not shot, it would be alarmed and find the hunter in a short time. However, in this one chance, additional arrows allow the hunter to shoot in different directions at the same time, to increase the chance of victory. Notice that, once the hunter puts down the arrow after shooting, there would be no more chance for shooting: Either the Wumpus is dead, or the hunter is eaten.

- bat: Just as introduced.

In this project, the captured hunter has a short time to escape from the bat before being taken to another place. Bat takes some time to rest. Also, when not spotting the hunter, bats would automatically move after some period of time.

- pit: Just as introduced.

Don't you think the Big Bastard Moon is very... disturbing? This whole game is a little bit disturbing so that it is more interesting... for example...

- poop: When a bat moves to one Vertex, it might leave a poop at one neighboring Vertex. The poop won't disappear by itself. It might indicate a nearby bat but is usually misleading, if not disturbing.

XD.

In the project, we have many pits and bats, so that the possibility of encounter is large. The numbers of these two depend on the size of the graph.

(#5 Develop your own implementations of one or more of the data structures in the project.)

I did develop a HashMap in last project, but I didn't use it in this week's project...

Demonstration

Bat Encounter: Disturbing bats were passing me like a stick in a race.

Dangerous Bat Encounter: Disturbing bats throwing me into a pit.

Extra Arrow Encounter: Demonstrates how you can use the extra arrow. Even if your first shot doesn't kill the Wumpus, you still have a second chance.

Normal Adventure: Boring Demonstration where the hunter successfully kills the Wumpus after a period of time neither long nor short.

Obstacle Encounter: Disturbing shape of the grid generated by the random removing method.

Pit Encounter: Disturbing (?) video about a stupid hunter walking right into its doom with no acts of wisdom at all.

Tunnel Digging: Disturbing (.) video about a hunter that keeps digging through the graph and nothing else.

UI demo: Disturbing (really) video because it has appeared earlier in this page.

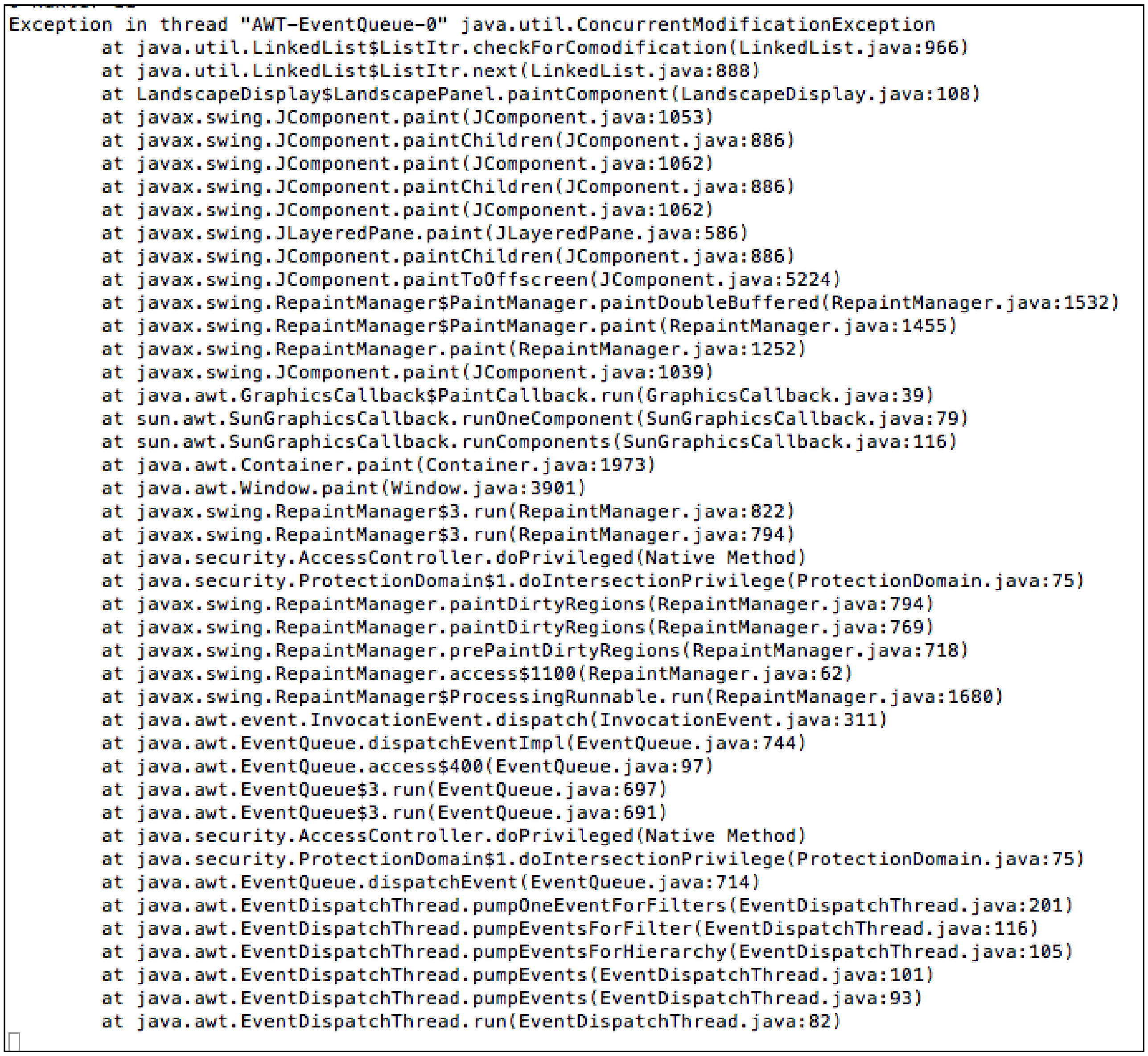
Wumpus Game Demo: Disturbing link. Enough said.

Wumpus Encounter: Disturbing hunter meets a disturbing wumpus and disturbingly walks straight into its mouth, giving a disturbing ending.

Problem:

1 The window keeps flashing once the first image gets drawn

2 an exception would occur in some rare situations, with no indication of where had went wrong. This won't end the game, though.



```
Exception in thread "AWT-EventQueue-0" java.util.ConcurrentModificationException
	at java.util.LinkedList$ListItr.checkForComodification(LinkedList.java:966)
	at java.util.LinkedList$ListItr.next(LinkedList.java:888)
	at LandscapeDisplay$LandscapePanel.paintComponent(LandscapeDisplay.java:108)
	at javax.swing.JComponent.paint(JComponent.java:1053)
	at javax.swing.JComponent.paintChildren(JComponent.java:886)
	at javax.swing.JComponent.paint(JComponent.java:1062)
	at javax.swing.JComponent.paintChildren(JComponent.java:886)
	at javax.swing.JComponent.paint(JComponent.java:1062)
	at javax.swing.JLayeredPane.paint(JLayeredPane.java:586)
	at javax.swing.JComponent.paintChildren(JComponent.java:886)
	at javax.swing.JComponent.paintToOffscreen(JComponent.java:5224)
	at javax.swing.RepaintManager$PaintManager.paintDoubleBuffered(RepaintManager.java:1532)
	at javax.swing.RepaintManager$PaintManager.paint(RepaintManager.java:1455)
	at javax.swing.RepaintManager.paint(RepaintManager.java:1252)
	at javax.swing.JComponent.paint(JComponent.java:1039)
	at java.awt.GraphicsCallback$PaintCallback.run(GraphicsCallback.java:39)
	at sun.awt.SunGraphicsCallback.runOneComponent(SunGraphicsCallback.java:79)
	at sun.awt.SunGraphicsCallback.runComponents(SunGraphicsCallback.java:116)
	at java.awt.Container.paint(Container.java:1973)
	at java.awt.Window.paint(Window.java:3901)
	at javax.swing.RepaintManager$3.run(RepaintManager.java:822)
	at javax.swing.RepaintManager$3.run(RepaintManager.java:794)
	at java.security.AccessController.doPrivileged(Native Method)
	at java.security.ProtectionDomain$1.doIntersectionPrivilege(ProtectionDomain.java:75)
	at javax.swing.RepaintManager.paintDirtyRegions(RepaintManager.java:794)
	at javax.swing.RepaintManager.paintDirtyRegions(RepaintManager.java:769)
	at javax.swing.RepaintManager.prePaintDirtyRegions(RepaintManager.java:718)
	at javax.swing.RepaintManager.access$1100(RepaintManager.java:62)
	at javax.swing.RepaintManager$ProcessingRunnable.run(RepaintManager.java:1680)
	at java.awt.event.InvocationEvent.dispatch(InvocationEvent.java:311)
	at java.awt.EventQueue.dispatchEventImpl(EventQueue.java:744)
	at java.awt.EventQueue.access$400(EventQueue.java:97)
	at java.awt.EventQueue$3.run(EventQueue.java:697)
	at java.awt.EventQueue$3.run(EventQueue.java:691)
	at java.security.AccessController.doPrivileged(Native Method)
	at java.security.ProtectionDomain$1.doIntersectionPrivilege(ProtectionDomain.java:75)
	at java.awt.EventQueue.dispatchEvent(EventQueue.java:714)
	at java.awt.EventDispatchThread.pumpOneEventForFilters(EventDispatchThread.java:201)
	at java.awt.EventDispatchThread.pumpEventsForFilter(EventDispatchThread.java:116)
	at java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:105)
	at java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:101)
	at java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:93)
	at java.awt.EventDispatchThread.run(EventDispatchThread.java:82)
```

Acknowledgement

I give thanks to all the websites that might have helped me...

Like Be the first to like this