| /\* |  |
| --- | --- |
|  | \* AP(r) Computer Science GridWorld Case Study: |
|  | \* Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com) |
|  | \* |
|  | \* This code is free software; you can redistribute it and/or modify |
|  | \* it under the terms of the GNU General Public License as published by |
|  | \* the Free Software Foundation. |
|  | \* |
|  | \* This code is distributed in the hope that it will be useful, |
|  | \* but WITHOUT ANY WARRANTY; without even the implied warranty of |
|  | \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the |
|  | \* GNU General Public License for more details. |
|  | \* |
|  | \* @author Cay Horstmann |
|  | \*/ |
|  |  |
|  | package info.gridworld.actor; |
|  |  |
|  | import info.gridworld.grid.Grid; |
|  | import info.gridworld.grid.Location; |
|  | import info.gridworld.world.World; |
|  |  |
|  | import java.util.ArrayList; |
|  |  |
|  | /\*\* |
|  | \* An <code>ActorWorld</code> is occupied by actors. <br /> |
|  | \* This class is not tested on the AP CS A and AB exams. |
|  | \*/ |
|  |  |
|  | public class ActorWorld extends World<Actor> |
|  | { |
|  | private static final String DEFAULT\_MESSAGE = "Click on a grid location to construct or manipulate an actor."; |
|  |  |
|  | /\*\* |
|  | \* Constructs an actor world with a default grid. |
|  | \*/ |
|  | public ActorWorld() |
|  | { |
|  | } |
|  |  |
|  | /\*\* |
|  | \* Constructs an actor world with a given grid. |
|  | \* @param grid the grid for this world. |
|  | \*/ |
|  | public ActorWorld(Grid<Actor> grid) |
|  | { |
|  | super(grid); |
|  | } |
|  |  |
|  | public void show() |
|  | { |
|  | if (getMessage() == null) |
|  | setMessage(DEFAULT\_MESSAGE); |
|  | super.show(); |
|  | } |
|  |  |
|  | public void step() |
|  | { |
|  | Grid<Actor> gr = getGrid(); |
|  | ArrayList<Actor> actors = new ArrayList<Actor>(); |
|  | for (Location loc : gr.getOccupiedLocations()) |
|  | actors.add(gr.get(loc)); |
|  |  |
|  | for (Actor a : actors) |
|  | { |
|  | // only act if another actor hasn't removed a |
|  | if (a.getGrid() == gr) |
|  | a.act(); |
|  | } |
|  | } |
|  |  |
|  | /\*\* |
|  | \* Adds an actor to this world at a given location. |
|  | \* @param loc the location at which to add the actor |
|  | \* @param occupant the actor to add |
|  | \*/ |
|  | public void add(Location loc, Actor occupant) |
|  | { |
|  | occupant.putSelfInGrid(getGrid(), loc); |
|  | } |
|  |  |
|  | /\*\* |
|  | \* Adds an occupant at a random empty location. |
|  | \* @param occupant the occupant to add |
|  | \*/ |
|  | public void add(Actor occupant) |
|  | { |
|  | Location loc = getRandomEmptyLocation(); |
|  | if (loc != null) |
|  | add(loc, occupant); |
|  | } |
|  |  |
|  | /\*\* |
|  | \* Removes an actor from this world. |
|  | \* @param loc the location from which to remove an actor |
|  | \* @return the removed actor, or null if there was no actor at the given |
|  | \* location. |
|  | \*/ |
|  | public Actor remove(Location loc) |
|  | { |
|  | Actor occupant = getGrid().get(loc); |
|  | if (occupant == null) |
|  | return null; |
|  | occupant.removeSelfFromGrid(); |
|  | return occupant; |
|  | } |
|  | } |