Graphical Display of a Physics Simulation
Project Code

Steven Oetjen

June 12, 2007

1 Driver

main(String args) Instantiates two JFrames, one containing `GraphicsPanel` and one containing `GUIPanel`.

2 GraphicsPanel

GraphicsPanel(int l, int w) Creates initially blank, white panel with dimensions (l, w), and a timer that fires FREQ times every second with a Listener.

xc(double x) Converts x-coordinates for an object in meters to x-coordinates on the panel in pixels.

yc(double y) Converts y-coordinates for an object in meters to y-coordinates on the panel in pixels.

xrc(double r) Converts the radius of a projectile in meters to an x distance on the panel in pixels.

yrc(double r) Converts the radius of a projectile in meters to a y distance on the panel in pixels.

draw() Repaints white background over previous image, then loops over all ramps and projectiles, drawing them in their correct positions in their correct colors.

checkCollision(Projectile a, Projectile b) Checks if two objects are overlapping, or occupying the same space. If so, changes their velocities, conserving momentum and kinetic energy to represent an elastic collision, using vector equations that produce the angular effects of non-head-on collisions.

sumForces(Projectile p) Applies gravitational force if necessary. Loops over all projectiles, calling `checkCollision()` for each combination of two without duplicating checks. Loops over all ramps to check for contact with projectile. This check has three cases. If the angle of the ramp is less than 45°, the projectile is within the x bounds of the ramp, and within the vertical $mx + b$ bounds, there is a collision. If the angle of the ramp is greater than 45°, the projectile is within the y bounds of the ramp, and within the vertical $\frac{1}{m}(y-b)$ bounds, there is also a collision. Finally, if there is a vertical ramp, which has equal x-coordinates, the projectile is within the y bounds of ramp, and the distance between the ramp and the projectile is less than the radius, a collision occurs. If any of these cases is satisfied, it checks if the projectile has negative perpendicular velocity, adjusting velocity to make the projectile bounce if necessary, and applies force exerted on projectile due to ramp, including normal force and frictional force. Calculates forces due to all springs attached to the projectile, due to gravity from all other objects, and due to electric charge from all other objects. Calculates acceleration from the sum of the forces on the projectile.

paintComponent(Graphics g) Used by Panel to draw images.

Listener Performs action whenever timer fires: loops through all projectiles, calling their `step()` methods, resets their sum of force variables, then calls `sumForces()`; calls `draw()`, updates simulation time, calls the GUI's `getValues()`, and calls `repaint()`.

3 GUIPanel

GUIPanel(int l, int w) Creates panel with dimensions (l, w), ArrayLists of all projectiles, ramps, and input panels for both. Sets up three subpanels in a BorderLayout - main, inputs, and scales - containing JTextFields, JButtons, JLabels, JRadioButtons, a JCheckBox, and JTabbedPanes.

setValues() Sets values of all projectiles and ramps to the values retrieved from text fields in the GUI. Calls `ProjectileInputs`'s, `RampInputs`'s, and `SpringInputs`'s `updateMatrix()` methods to calculate cartesian to polar coordinates or vice versa before setting objects' values.

setScales() Sets scales for the `GraphicsPanel` window to the values retrieved from the text fields in the GUI.

getValues() Retrieves values from all projectiles, ramps and springs, and updates all text fields on the GUI to show these values. Checks whether coordinate or polar mode is enabled to display correct values.

deleteProj(int index) Deletes all record of a projectile, including the actual `Projectile` object, its `ProjectileInputs` object, and its slot on the tabbed pane.

deleteRamp(int index) Deletes all record of a ramp, including the actual `Ramp` object, its `RampInputs` object, and its slot on the tabbed pane.

deleteSpring(int index) Deletes all record of a spring, including the actual `Spring` object, its `SpringInputs` object, and its slot on the tabbed pane.

openFile(String f) Closes current file by deleting all ramps and projectiles. Reads tags from a text file with filename ”f” signaling a section of data belonging to a certain object of that tag type. Creates objects of all tags in the file, assigning their appropriate values to those obejcts. Catches `FileNotFoundException` and `IOException`.

saveFile(String f) Prints to text file with filename ”f” for all projectiles, ramps, and scales, printing all values in order in correct sections of the text file. Catches `FileNotFoundException`.

RunListener Calls `setValues()` and `setScales()`, instructs the `GraphicsPanel`’s timer to start, disables all buttons except for stop.

StopListener Instructs the `GraphicsPanel`’s timer to stop, enables all buttons except for stop.

ResetListener Resets the time to 0, and if there is a file opened, reopens that file to its original set-up.

AddProjListener Prompts user for a name, creates a `Projectile` object and a `ProjectileInputs` object under that name, and adds it to the tabbed pane.

AddRampListener Prompts user for a name, creates a `Ramp` object and a `RampInputs` object under that name, and adds it to the tabbed pane.

AddSpringListener Prompts user for a name, creates a `Spring` object and a `SpringInputs` object under that name, and adds it to the tabbed pane.

OpenListener Prompts the user for a filename, calls `openFile()` if valid filename.

SaveListener Prompts the user for a filename, calls `saveFile()` if valid filename.

TextListener When focus of a text field is gained, selects all characters of contents of that text field.

RadioListener Loops through all `ProjectileInputs` and `RampInputs`, updating all cartesian cordinates to polar, or vice versa, depending which radio button is selected.

4 Projectile

Projectile(String n) Creates projectile object with name, color, radius, an array of scalar values, and a matrix with components of all vector values.

equals(String n) Returns true if n equals the name of the projectile, false otherwise.

step(double t) Updates the projectile's position (using its velocity) and its velocity (using its acceleration). This integration is done by storing the previous two acceleration points, finding the equation for a parabola through the three points, $(-t, a_{-2})$, $(0, a_{-1})$, and (t, a_0), and integrates from 0 to t. Converts to both coordinate and polar.

5 ProjectileInputs

ProjectileInputs(String n, boolean isCoord) Creates an input panel for a projectile, storing its name and color. Contains JButtons for changing color and deleting, and JLabels and JTextFields for the values of the projectile.

equals(String n) Returns true if n equals the name of the projectile, false otherwise.

updateMatrix(boolean isCoord) Loops through each vector and updates the coordinates in polar, if coordinate mode is enabled, or in coordinate, if polar mode is enabled.

TextListener When focus of a text field is gained, selects all characters of contents of that text field.

ColorListener Opens a JColorChooser dialog window, changes background of button and color of projectile to new color selected by user.

DeleteListener Finds the index of the projectile by comparing this `ProjectileInputs` to all in the ArrayList. Calls the `GUIPanel`'s `deleteProj()` on that index.

6 Ramp

Ramp(String n) Creates a ramp object with a name, color, a matrix of two endpoint values, and a thickness.

equals(String n) Returns true if n equals the name of the ramp, false otherwise.

7 RampInputs

RampInputs(String n, boolean isCoord) Creates an input panel for a ramp, storing its name and color. Contains JButtons for changing color and deleting, and JLabels and JTextFields for the values of the ramp.

equals(String n) Returns true if n equals the name of the ramp, false otherwise.

updateMatrix(boolean isCoord) Loops through each coordinate and updates the coordinates in polar, if coordinate mode is enabled, or in coordinate, if polar mode is enabled.

TextListener When focus of a text field is gained, selects all characters of contents of that text field.

ColorListener Opens a JColorChooser dialog window, changes background of button and color of ramp to new color selected by user.

DeleteListener Finds the index of the ramp by comparing this `RampInputs` to all in the ArrayList. Calls the `GUIPanel`'s `deleteRamp()` on that index.

8 Spring

Spring(String n) Creates a spring object with a name, color, a matrix of two end-point values, a stiffness constant, a natural length, a damping factor, and two `Projectile`s attached to its ends (`null` if fixed to nothing).

equals(String n) Returns true if n equals the name of the spring, false otherwise.

getIndex() Returns index of the spring in the tabbed pane and lists of springs.

setCoord() Updates the coordinates of the spring to match those of an attached projectile. Does nothing if the end is fixed (not attached to any projectile).

9 SpringInputs

SpringInputs(String n, boolean isCoord) Creates an input panel for a spring, storing its name and color. Contains JButtons for changing color and deleting, and JLabels and JTextFields for the values of the spring.

equals(String n) Returns true if n equals the name of the spring, false otherwise.

getIndex() Returns index of the spring in the tabbed pane and lists of springs.

updateMatrix(boolean isCoord) Loops through each coordinate and updates the coordinates in polar, if coordinate mode is enabled, or in coordinate, if polar mode is enabled.

TextListener When focus of a text field is gained, selects all characters of contents of that text field.

ColorListener Opens a JColorChooser dialog window, changes background of button and color of ramp to new color selected by user.

DeleteListener Finds the index of the spring by comparing this `SpringInputs` to all in the `ArrayList`. Calls the `GUIPanel`'s `deleteSpring()` on that index.

ObjectListener(int n) Argument n is either 1 or 2, corresponding to an end of the spring. Calls a `JOptionPane` window to ask which projectile to attach that end to the spring to. Removes the spring from the list of attached springs of the old projectile and adds the spring to the list of the new projectile. Sets the text of the button to the name of the selected projectile.