Newton's Playground

By Samarth Shah and Samrudh Shenoy
May 2019

Our program is a visual physics simulator, specifically designed for building and running user-built kinematic ball and obstacle configurations. Our program is very customizable, letting the user set their personal preference for aspects such as downward gravitational acceleration, obstacles in the simulation, ball weight, etc.

The overall goal of the program is to allow the user to test various and infinitely many scenarios to experiment with how the physics in our world affect the way that objects move. The program can be used in both a scientific setting, as well as simply for entertainment. Due to the inability or inconvenience to change certain real-life variables (gravity, a singular object's mass, etc.), our program aims to allow users to have the power to do so at their fingertips.

One of our main inspirations motivating the creation of Newton's Playground is our knowledge of Physics. We both have knowledge in Physics A and B, and wanted to create an application at the crossroads of Computer Science as well as another science. Additionally, during class, our physics teacher often mentions that she wishes we could take our experiments farther, but that our limitations to alter physical qualities stand in the way. Through this online program, students can conduct such experiments in the manner preferred, including our physics teacher who we hope to show this program to.

Regarding usability, our designated target market are those who are involved in physics, so curious students or physics departments at schools could use the program provide an easy to comprehend visualization of common word problems students tend to have trouble on, or to explain the meanings behind formulas.

Instructions:
Once program is launched:
- Press "Exit" to exit the program
- Press "Help" to view a detailed set of instructions
- Press "Start" to enter the simulator

* Note: Our program assumes a perfect physics world, so there is no decrease of the ball's total velocity because total mechanical energy is conserved

- Left side bar:
    - Save/Load an experiment with a user-determined name
    - Play/Pause an experiment at a desired time
    - View specific data collected within the time that the game was played and paused
    - Create a new obstacle with a given coordinate start point, an angle, and a length
    - Remove all obstacles currently displayed on the screen
- Right side bar:
    - Set the gravitational acceleration of the world
    - Set the (X, Y) coordinates of the ball
    - Set the radius of the ball
    - Set the mass of the ball
    - Set the current x-directional velocity of the ball
    - Set the current y-directional velocity of the ball

Class List:
- Main
- Obstacle
- Ball
- Force
- World
- MainScreenController
- MenuScreenController
- ObjectAdderPanelController
- WorldEditorPanelController
- LoadScreenController
- SaveScreenController

Features:

Must Have:
- Balls can be let go from different place
- Gravity and ball weight can be set to custom values
- Ability to set initial x and y velocity
- Pause and play options
- Ability add new planes into the simulation, as well as reset them
- Data collection during experiments

Want to Have:
- Ability to save experiments
- 3D User Interface
- Ability to set air pressure and resistance
- Launching the ball out of cannons/catapults
- Inclusion of water mechanics
- Ability to set speed of process (slow mo or fast forward, or even rewind)

Stretch:
- POV from object's standpoint
- Inclusion of electricity and/or magnetism
- Different objects (spherical shell, box, pole)
- Parachute
- Ability to record

Credits:

- Samrudh Shenoy: Calculation of angles and forces, theme css file, readme file
- Samarth Shah: User interface, writing of most model classes, UML diagram
- Mrs. Shreve: Help with angles and forces, because we did not have much direction (pun intended) to start with
- Websites we used:
    Collisions: https://stackoverflow.com/questions/849211/shortest-distance-between-a-point-and-a-line-segment
    Animations/Graphics: https://netopyr.com/2012/06/14/using-the-javafx-animationtimer/
    Canvas: https://docs.oracle.com/javafx/2/canvas/jfxpub-canvas.htm

Feedback

* Move "Different objects" to Want to Have
* Perhaps use different materials
  * Such as a rubber ball or metal ball
* Change the readme to flow better and "sell" your program

UML
* Add depends on relationships
* Nothing is going into ball, ramp, wall, and platform