Cs32

Solar Helpsession

Claudiu Saftoiu

Alex Tuteur

Introduction

- Solar is a pretty simple project, designed to give you some introductory experience with software engineering, including:
  - Designing a project
  - Using Eclipse
  - Developing and executing a testing plan
- The trickiest part of Solar is probably the physics involved.

Getting Started

- Getting the stencil code is easy – just run the install script.
- Getting the project to run in Eclipse is a bit harder.
- There are detailed instructions in the handout which describe step by step how to edit Solar in Eclipse.
- We're encouraging you to use Eclipse for this project because it is a valuable tool that will make your life **much** easier down the road.

The Physics

- Solar is a physics simulator, so the biggest part of the project is really getting the physics right.
- The physics equations might seem intimidating at first, but they're actually pretty simple once you look at them for a bit.

The Main Loop

- For each planet p_1
  - For each planet p_2
    - Calculate force exerted by p_2 on p_1, using gravitational force equation in handout. Keep track of total force on p_1.
  - Use total force on p_1 to calculate p_1's new position and velocity using the equations in the handout.
- For each planet p_1
  - Update the planet's position and velocity using the previously calculated values.

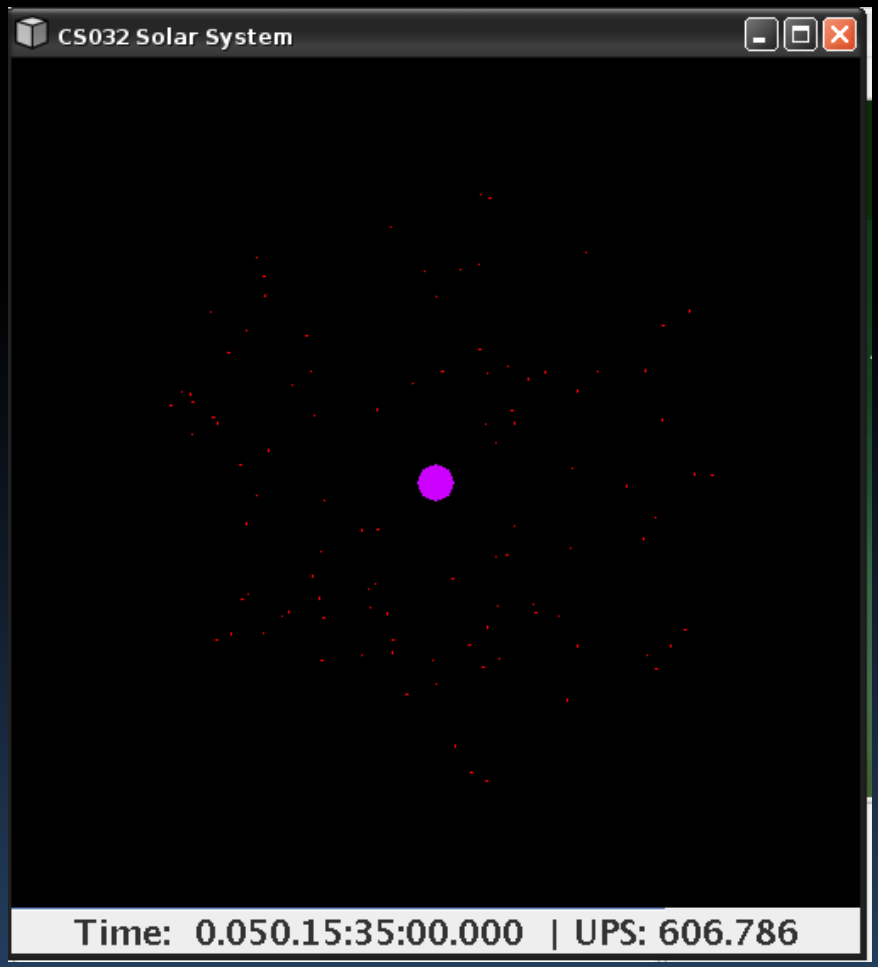
Physics In the Main Loop

- All that's really required here is four basic equations!

- Gravitational force: $F = \frac{Gm_1m_2}{d^2}$

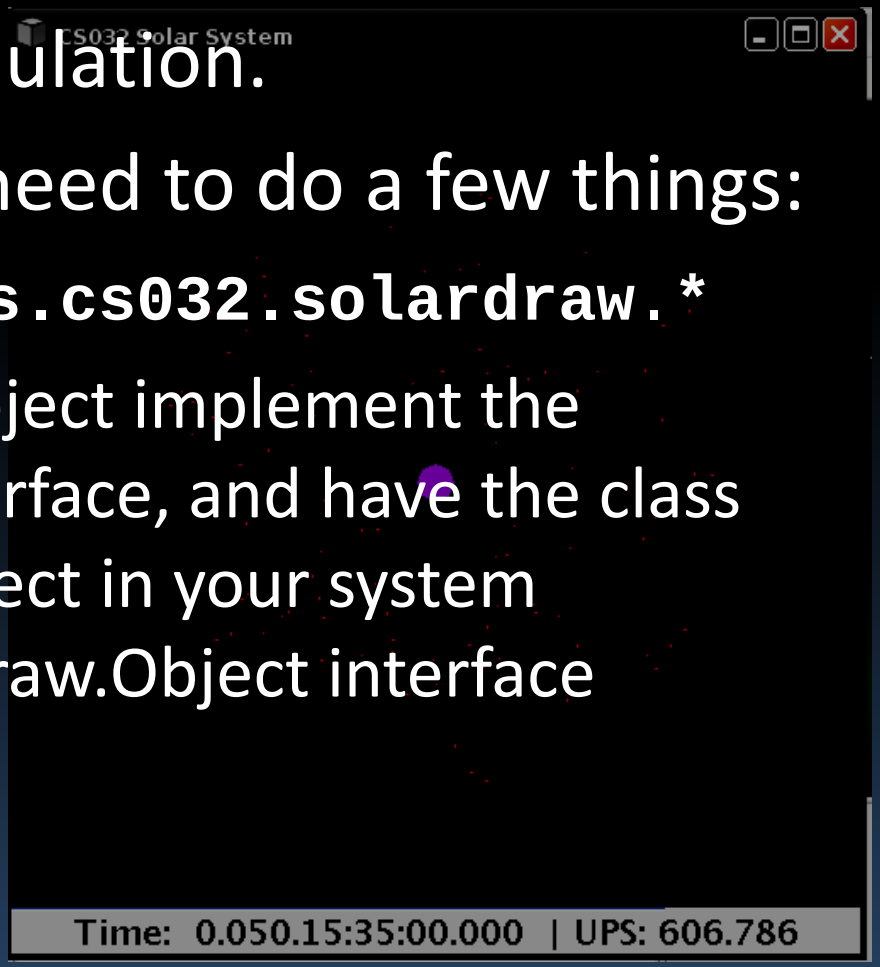
- Newton's Second Law: $F = ma$
- Change in velocity: $v = v_0 + at$
- Change in position: $p = p_0 + v_0t + \frac{1}{2}at^2$

The GUI



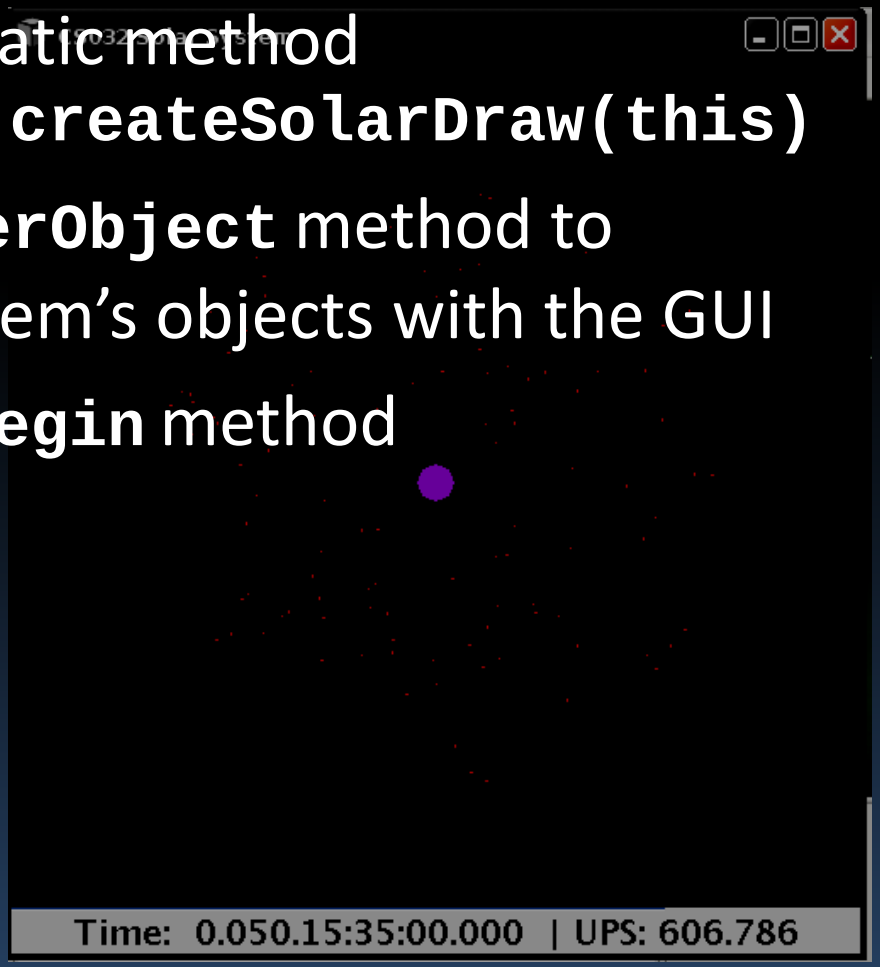
The GUI

- The SolarDraw GUI lets you visualize what's happening in your simulation.
- To make it work, you need to do a few things:
  - Import **`edu.brown.cs.cs032.solardraw.*`**
  - Have your top-level object implement the SolarDraw.Control interface, and have the class that represents an object in your system implement the SolarDraw.Object interface



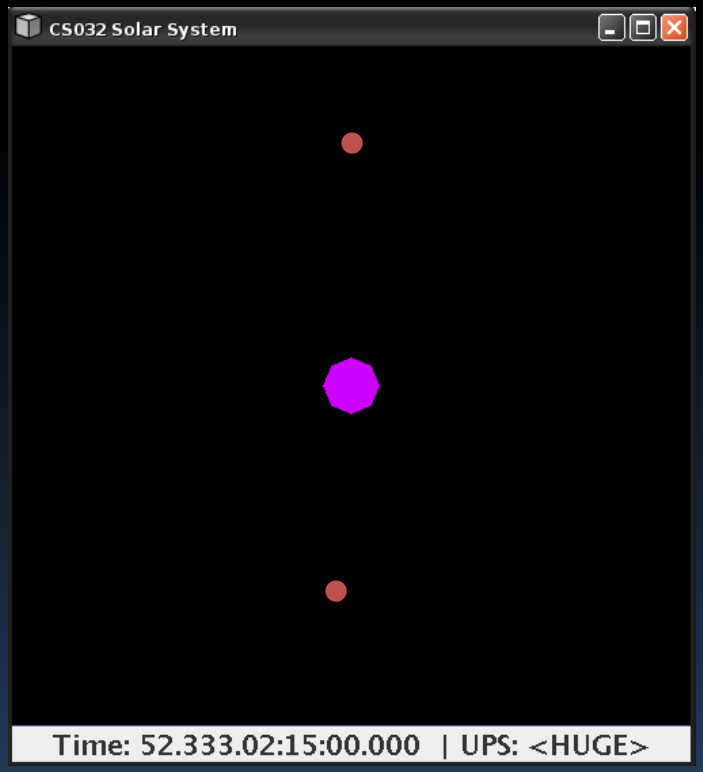
The GUI (continued)

- Create an instance of the GUI in your top-level object by calling the static method **`SolarDraw.Factory.createSolarDraw(this)`**
- Use the GUI's **`registerObject`** method to register your solar system's objects with the GUI
- Finally, call the GUI's **`begin`** method



Testing

- Testing your simulation by running it with 100 objects is probably a Bad Idea
  - A few ideas for tests:
    - A star and an object with zero velocity
    - One object orbiting a star
    - Two objects orbiting a star in opposite directions



Questions?