Project Presentation

Astronomy

Simulum:

Simulation of star movements

Kazaki Argyro
Department of Management Science and Technology
Athens University of Economics and Business

Friday, 09/06/06

A brief summary..

Previously on “Simulus”, we have talked about certain inconsistencies between user needs and application functions. We discussed about different simulation profiles, export and printing options.

Now, I think it’s the right time to get into the code and talk java…

digitalblasphemy.com

How things work:

Application, uses 8 main attributes to perform the simulation:

- gamma
- radius
- stars
- sensitivity
- snapshot
- movement
- deltat
- zoom

These are stored in a .properties file named milumis.properties

How things work2 :

Everytime a change of those variables takes place, simulation stops, milumus.properties is resaved (or refreshed, you can call it either way), the new attributes are loaded and a new simulation starts. Actually, these properties, are not refreshed. The .properties file is been reproduced form the beginning even if the smallest change occurs, and overrides the old one. And this happens also, when the application is loaded for the first time (with the default values of course).

The revenge of the Properties!!

The first thing I changed was during the shutdown process when I request that the default values should be stored instead of the current ones.

Note: Default values are placed hardcoded in the class the manages the milumus.properties file. So if things change, Meyling will have to recompile the project in order to change the values. If I am well informed, isn't this the reason why Properties and external .properties files have been introduced to java? To store variables that might change without affecting the source code? So, shouldn't there be a separate file, storing the default properties? Ambiguous, isn't it???

I'm joking of course..

Anyway..

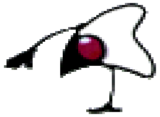
So, the astronomer observes the stars, makes changes and when he decides that he wants to save his simulation, presses the “Save” button. A new .properties file, named after the current date and time is being created, with all the selected attributed stored in it.

Cool.

And when he decides that he wants to study the specific simulation again, he just presses the “Open” button. A JFileChooser appears on screen, and asks for the path of the property file. Once the user provides the application with the path, the attributes are loaded and the simulation is on again.

More cool!

Exporting…



Remember that piece of paper we talked about the previous time? Yes, the notepad with attributes written on? Yeap. It's gone. While simulating, the user can choose to export the specific data. This time though, I didn't use a JFileChooser. I decided a default export path from the beginning, and all exports are stored in it. Exports come in a .txt file, with a standard name. The .txt contains the current date as a heading and one row for each property.

So, now, the astronomer can feel free to lose any of these papers. Simulus will simply produce another one!

And something funny know: I just said something about a default export path. I didn't placed it hardcoded. Can you imagine where is it located?? (for hints, take a look at slide 5)

Flaws…

Being into developing for only 3-4 months, I have already come to a rather interesting conclusion:

Client requirements and debugging, NEVER end…

So, while dealing with the stuff mentioned before, I have found a lot of flaws that would worth dealing with:

When saving profiles, I do not let the user choose the name he likes for the property file; instead, I use current date.toString() to produce it. The user, may later change it, of course, and it would be fine by me, but since we've been talking about "Client-centered" applications, such simple functions must be provided.

Flaws 2 …

Previously, I have been talking about JButtons to handle the actions described. The truth is that so far, most of these, are implemented using JOptionPanes. Not, so good, I know. I am not at all fully aware of how GUI and graphical interfaces in general work and I had a few troubles with positioning and layout. At that time I was more interested in implementing the function, rather than presenting it, so I dropped the effort. This must be fixed.

Finally, I am willing to consider the possibility of allowing the user to decide his own export path. And the name?? Xmm..I must put some thought in it..

The end…

So, my fellow devs, we have reached the end of the project, the end of the semester, and its time we make our ways towards the beautiful Greek islands…

Hope you liked Simulus as much as I did..

Hope you liked my ideas..

My best wishes!

System.out.println(“Over and Out”);