Tecnología de la Programación - Academic Year 2018/2019

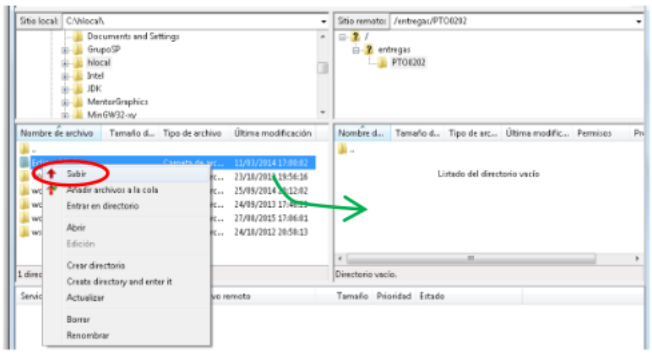
May's Exam (28/05/2019) - Duration: 3 hours.
Maximum grade: 10 points

Grados en Ingeniería Informática, del Software y de Computadores (Grupos B, C, F)
y Doble Grado en Informática y Matemáticas (Grupo A)

Instrucciones

- In this exam you have to start from the last version of assignment 5 that you have uploaded to the Campus Virtual recently. Shortly you will be provided with instructions on how to download it.
- **Create a text file `changes.txt` in the root of your project (inside `src`)**. In this file you will have to include the names of all files (classes, etc.) that you have modified or added. In addition, you can include other comments regards your solution that will be taken into account during the marking process.
- The submitted code ***MUST COMPILE***, otherwise you fail the exam.
- ***Breaking encapsulation*** (accessing private and protected fields from external classes, use of public fields, etc.) implies failing the exam.
- When marking the exam, we will evaluate functionality, clarity of the code, the use of object oriented principles (inheritance, polymorphism and dynamic binding) and comments.
- **In order to get the extra material** (slides, Java API doc, and an icon **exam.png** to be used in P3): click on **Publicación docente de ficheros** that is on the Desktop; choose **ALUMNO recogida docente**; and look for the directory **Todos/TP2019**.

Submission Instructions



To submit the exam, create a file called `YourName.zip` that includes your modified code and a file `student.txt` with your name. Double click on the icon **"EXAMENES en LABs entregas..."** that appears on the desktop, this will open a new window. Inside the window click on **"ALUMNOS entrega de practicas y examenes"**. A new window will pop up, in which you have to select the `zip` file that you have created and drag it to the right panel (or use the right button of the mouse and then select the option **Subir**). See the figure. Before leaving the lab, you have to pass by the professor table to verify that your code has been submitted correctly and sign the submission form.

Questions

1. **[3.5pt]** Add a new kind of body called *trapped* to the simulator. A *trapped* body behaves like a normal body, but it never moves to a position that is at distance more than *dist* from its initial position. This means that if a move takes it to a position at distance more than *dist* from its initial position, then it stays at the same place and its velocity stays the same as it was before the move. The following is an example of a JSON defining a body of this kind (dist is of type Double).

```
{
  "type" : "trapped",
  "data" : {
             "id"   : "b4",
             "pos"  : [0.0e00, 4.5e10],
             "vel"  : [1.0e04, 0.0e00],
             "mass" : 1.5e30,
             "dist" : 2e10,
           }
}
```

2. **[3pt]** Modify the *batch* mode of the simulator such that executed the option "-axe i" in the command-line, at the end of the simulation it prints for each body the number of times that its ith coordinate changed sign (i.e., from negative to nonnegative and vice versa, considering zero as positive). Assume that the value of i is correct, i.e., it refers to a valid coordinate. **You are not allowed to modify any class except the Main, but you can create new classes.**

   Note. Test your implementation using the gravity law ftcg, where change of sign occurs often (for example using ex3.4body.txt). If you want you can first use a fixed value for the coordinate i, and once it works add the support for the command-line parameter "-axe i". In order to detect changes in states store the last state (postive/negative) of bodies.

3. **[3.5pt]** A bang is a phenomena that reduces the mass of each body in the simulator by $X\%$, where X is a number between 1 and 100 called the power. This means that if the mass of a body is m before the bang, it will be $m * \frac{100.0-X}{100.0}$ after the bang. In this question you are required to modify the simulator to allow simulating a bang:

   - Modify the model (and controller) to have a method "void bang(int x)" that performs a bang, and then sends a corresponding notification to observers (you should create a **new kind of notification** for this event). It should throw an `IllegalArgumentException` if x has an invalid value. If your class Body does not have a method setMass, you can add one.
   - Add a button to the control panel that allows performing a bang, and a JSpinner with values from 1 to 100 that represents the power of the bang. Use the icon exam.png that we provide with the exam. The new button should be disabled during the simulation.