Small Group Exercise - Start a UML Class Diagram to model the GUI for your Units selection option

Lynn Robert Carter, PhD

2018-02-24

Activity Kind

Small Group Exercise

Purpose

The purpose of this exercise is to make the design choice more concrete by being specific about which classes and methods are needed to implement the GUI for the Units input selection.

Pre-requisite

Students are expected to have completed

- the Individual Research - How does the JavaFX Accordion Select List work?
- Small Group Exercise - Design choices text, combo box, accordion, or other

Tasking

The group must start by listing the library API elements required for the input option selected by this group. These elements include the list of classes and the methods within the classes required for this choice. The best way to do this is to search for tutorials or examples on the internet. For example, a search for these items: “JavaFX”, “combo box”, “Eclipse” returns a nice list of hits. The tutorials from “docs.oracle.com” are worth your study, as that site is a trusted information source as is information from “stackoverflow.com”.

Regardless of the input mechanism you choose, the code must transform the input action into an integer value in the range 0..*n* and it would be reasonable to encode an error as -1. Create a Unit class as follows:

- The constructor, “`new Unit()`”, creates an object representing a particular unit without specifying what that unit is. Internally a specifier of -1 is used, so attempts to actually use it without further specification of which unit would generate an error.
- The constructor “`new Unit(int unitSpecifier)`” creates a specific object, specified by the integer parameter.
- The getter “`int getUnitSpecifier()`” returns the internal integer value of the specifier.
- The setter “`void setUnitSpecifier(int unitSpecifier)`” sets the internal integer value to the input parameter’s value as long as it is in the range 0..*n*. If the parameter’s value is not in that range, it sets the internal value to -1, signaling an error.
- The method “`toString()`” in this class returns a String object appropriate for displaying the specific unit to the user as a string of characters (e.g. m/s², the text you should use for acceleration) as opposed to an integer in the range 0..*n*.
- The method “`boolean checkIfValidForAddition(Unit operand1Unit, Unit operand2Unit)`” is used to check whether or not the two specified units are valid for the operation of addition. There are similar methods for subtraction, multiplication, and division. The similar method for Square Root has only one parameter.
- The method “`Unit unitResultingFromAddition(Unit operand1Unit, Unit operand2Unit)`” takes the two input parameters and determines what the resulting Unit should be. If the inputs are not valid or the result is not valid, it returns an empty Units (e.g. “`new Unit()`”). There are similar methods for subtraction, multiplication, and division, as well as one for Square Root with only one input parameter.

What other methods are needed for this Unit class? Are there other classes needed? If so, what is it that that class represents and does?

The small group is expected to produce a set of documentation complete with the classes and methods that will be needed and any new classes and methods as well as new methods for any existing classes for a demonstration project leveraging the testbed you have been using. (Do **not** integrate this into your calculator now!) That documentation should look like the bullet items shown to you on the previous page.

The small group is then expected to produce a UML Class Diagram that explicitly lists each of the methods that you have documented as part of the classes documented in the diagram.

Produce as much as you can in the time allotted. You will be given time to complete this work, but you **must** show good progress during this exercise.

Deliverable

Each individual member of the group is expected to provide evidence of the documentation and the start of the resulting diagram in their ENB.

The students **must** also take notes during the exercise and record any concerns or doubts in their ENB. If there are no notes from this activity in an individual's engineering notebook, our only conclusion must be that you did not participate.

Submission

Students are expected to **complete** this part of their ENB prior to starting the next activity.