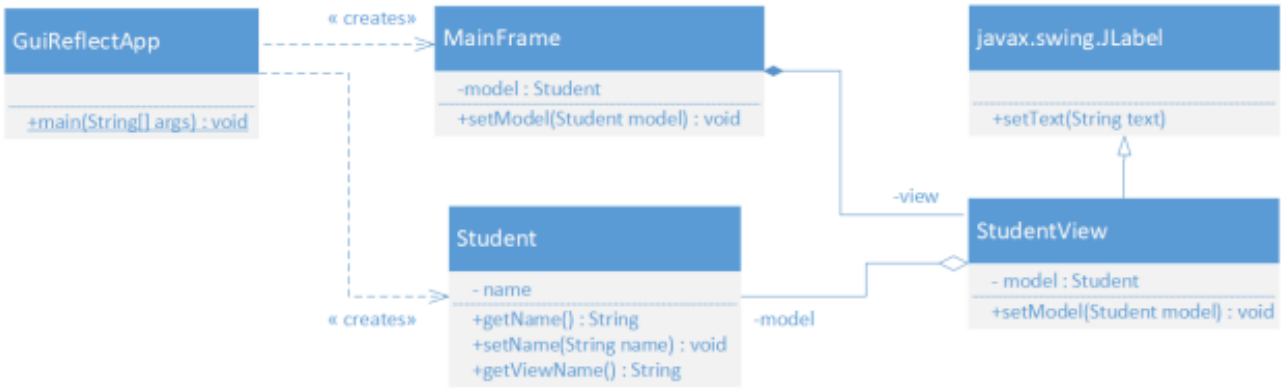
Background:

Consider the following UML design,



This design is implemented in the NetBeans project given in this Topic's online Worldclass notes.

When you run this NetBeans application, it displays a single window frame, which contains a view displaying the name of the associated student model. In this application, the creation of the StudentStudent view and its relation to the Student model is hard-coded within the application,.

Functional Requirements Part I:

1. Change the design of this GuiReflectApp application to remove the hard coding of the StudentView and its knowledge of a StudentModel.

   Instead, use reflection to ask the passed Model object for its associated view using the model's getViewName() method, which should return the fully qualified class name of the view that is used to display this object (in this example, it will be the StudentView)

   Hence, you are not changing the overall logic or design of this program, but simply using the reflection to create the StudentView class that is added to the MainFrame.

2. Use reflection to replace the "getName()" and setName(…) messages sent to the Student model. (See the main() method and updateDisplay() method in StudentView).

3. Appropriately handle all exceptions

   Essentially, you are replacing the hard-coded "new StudentView()" and "model.getName()" methods in the existing project.