Serie 3

Content:
– Design Pattern : Observer

(14) Design Pattern : Observer

Given a simple application implementing and "analog" clock, made of the following classes/interface in the `clock` package :
– The main class `ClockApp.java`.
– The utility class (Singleton) `clock.util.PositionManager.java` generating the window positions on the screen.
– The `clock.timer.ClockTimer.java` class updating the time.
– The `clock.analog.AnalogClock.java` frame containing a panel with the analog clock.
– The `clock.analog.AnalogClockPanel.java` class drawing the hands and numbers of the analog clock.
The class diagram of Figure 1 show the structure and the relations of these classes.

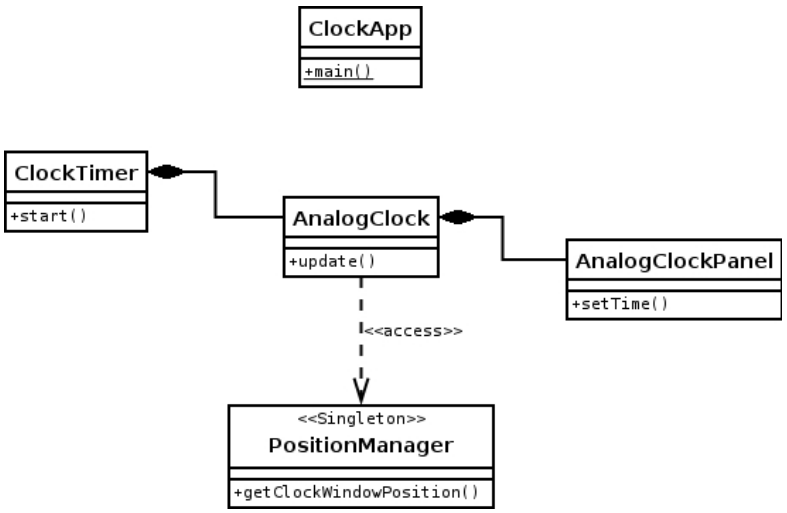

Figure 1 – Diagramme de classes de l'application de base Clock.

As you can see, the design is far away from an optimal way. It is difficult to extend, difficult to reuse, classes are highly coupled,...

1. Create a new project in your IDE with the classes available online [3].
2. Test the created project and get familiar with the source-code.
3. Improve the desing of the application :
   – Modify the structure of the classes in order to implement the Observer pattern [2].
   – Modify the main class in order to be able to add and remove observers dynamically (see demo during lecture).
   – Based on your new desgin, implement a digital clock.
   – Optional : create yet another cool clock. If you lack inspiration, have a look at the site of Tokioflash [1].
4. Document your architecture with the help of a class diagram and a sequence diagram.
5. Comment and jusitfy the choices you made (pull/push model,...)

Références

[1] Tokyoflash japan, 2012. http://www.tokyoflash.com/en/watches/1/ (accessed Apr 03, 2012).

[2] Erich GAMMA, Richard HELM, Ralph JOHNSON, and John VLISSIDES. *Design Patterns - Elements of reusable object-oriented software*. Addison-Wesley, 1995.

[3] Jacques Pasquier. Génie logiciel I, 2013. http://moodle2.unifr.ch/course/view.php?id=1252 (accessed Mar 25, 2013).