

SDT Projects 2011-2012

1. *Document editor*

Implement a text and image editor. For the text the user can choose color, font and size. The strings are organized in paragraphs, each paragraph has an alignment (left, right, justified, center) which can be chosen at runtime. It can be possible to introduce new formats. For example the user can select to save font=Times New Roman, size=12, color=red under the name “heading1” and reuse this format for other texts.

Images can be inserted and also formatted (choose the margin between the text and the image).

The editor should support undo/redo for the last 10 operations.

In the second edition we want to introduce additional features such as character counters or dictionaries.

2. *Feedback GUI Builder*

A builder for graphical user interface feedback application. It allows building flexible interfaces for customer interviews, varying the number of windows, presentation, implementation (awt/swing), and also customizing each window with different controls (buttons, check boxes). The frames can be enriched at runtime, for example selecting a thicker border or adding a watermark.

3. *LAN Communication Framework*

Provide a framework for the abstraction of network communication. It should contain classes for peer to peer transmission of text files, binaries and serialized object over different protocols (TCP, UDP, RMI). Use your framework by building a distributed application in which users send different files one to another.

4. *House surveillance solutions*

Provide a framework for deploying different sensors (temperature, humidity, smoke, break-in, composite) in different topologies. The framework provides a library of topologies (star, mesh, linear).

Sensors also act as wireless repeaters, forwarding events they receive from other sensors to the next sensor until a main server is reached which deals with the events. Event handling can be customized ranging from notifications (email) to sending commands to actuators present in the house (for example

shutting down the gas). The event handling can be enriched with other handlers. Use your framework to build a graphical user interface in which the user can add sensors and simulate the system by generating different events.

5. *Snake game*

Implement the snake game ([http://en.wikipedia.org/wiki/Snake_\(game\)](http://en.wikipedia.org/wiki/Snake_(game))). The user can choose new game or options. In options menu he can choose between multiple scoring systems, how many lives the snake has and also graphical features such as if the snake has or not a shadow.

6. *File Processor*

Create a file processor able to scan large records of xml files organized in folders. The files should be retrieved lazily. We should be able to retrieve only files accomplishing one or more criteria, specified at runtime. The files have fields with different authorization levels such as `<secret>...</secret>` or `<top-secret>..</top-secret>` which should be retrieved only if the user has the required privileges.

The folders and files can be copied, moved or deleted; operations which must support undo command. Assume also other routines will navigate through the file system.

7. *RPG Game*

Implement a text-based Dungeon and Dragons RPG game. As your hero moves through different zones enemies and items should spawn at running time. Enemies and items are created starting from some base classes and attributes which can be combined in different ways.

It is possible to introduce new base classes and new attributes. The player can play on different levels of difficulty which change the strength and number of the enemies. The game can be saved and resumed.

Observation: You will be graded according to your judicious choice, usage and documentation of design patterns in your project as well as to the overall quality of your produce software. You should use at least five non-trivial (Interface, Delegation) design patterns. Be advised that the design patterns are taken into consideration only if they are implemented. For example for Iterator it is not enough to use the Iterator supplied in Java API.