**Student**

Bernardo Atalaia – 59962

Design Patterns:

1. Observer

This Design pattern is usually used to observe a specific class, that is notified once a modification has happened in this class. Other classes that are waiting for a specific state or event, can use their observer to observe it and notify once an important action is made.

In the example below, GPUndoListener is an Interface of an observer, any observer class that implements this is going to observe any actions related to “undo”.

In the second example, there is a class (“UndoManagerImpl”) notifying it’s observers (“listeners”) of an action that happened.

Path: main\java\net\sourceforge\ganttproject\undo\GPUndoListener.java

main\java\net\sourceforge\ganttproject\undo\UndoManagerImpl.java

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

1. Facade

This Pattern is really useful to simplify code and it’s understanding, aswell as giving a simplified interface to a complex system. In the example below, the interface TaskContainmentHierarchyFacade does just that.

Path: main\java\net\sourceforge\ganttproject\task\TaskContainmentHierarchyFacade.java

Uma imagem com texto

Descrição gerada automaticamente

1. **Memento Pattern**

This pattern is basically a backup of a class, or something stored by the class itself of a previous state/version that can be brought back if for some reason an undo is needed.

As the class UndoableEditImpl has in the example below, the old document saved in case an undo is needed along side with the method undo that brings the old document back.

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente