Módulo III
Padrões GOF: Observer

Professores
Eduardo Bezerra – edubezerra@gmail.com
Ismael H F Santos – ismael@tecgraf.puc-rio.br

Ementa

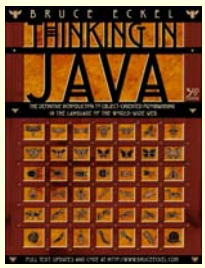
- Padrões GOF
  - Observer

Bibliografia

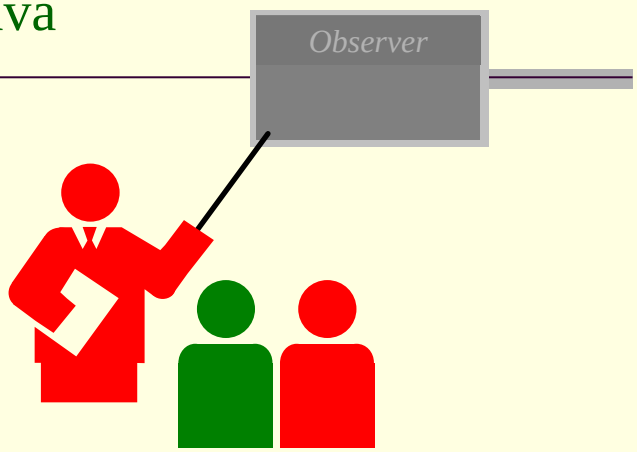
- *Eric Gamma, et ali, Padrões de Projeto, Ed Bookman*
- *Martin Fowler, Analysis Patterns - Reusable Object Models, Addison-Wesley,1997*
- *Martin Fowler, Refatoração - Aperfeiçoando o projeto de código existente, Ed Bookman*

Livros

- **Core Java 2**, Cay S. Horstmann, Gary Cornell
  - Volume 1 (Fundamentos)
  - Volume 2 (Características Avançadas)
- **Java: Como Programar**, Deitel & Deitel
- **Thinking in Patterns with JAVA**, Bruce Eckel
  - **Gratuito**. http://www.mindview.net/Books/TIJ/



POO-Java

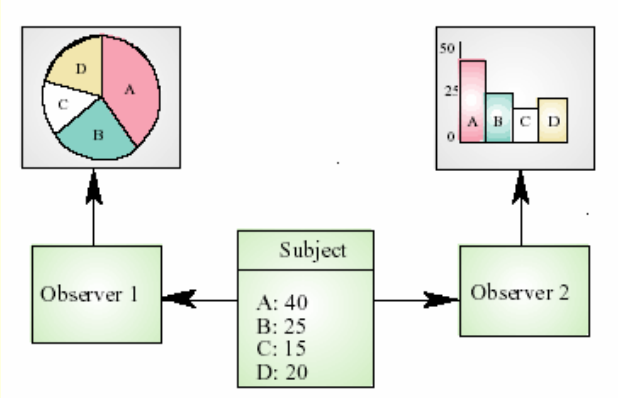


Observer

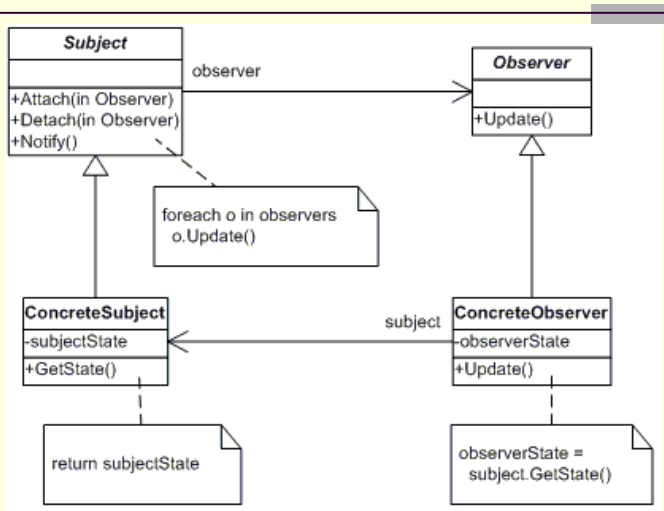
- Existem situações onde diversos objetos mudam seu estado de acordo com a mudança de estado de outro objeto.
  - e.g. as *views* e o *model* no framework MVC
- Define uma relação de dependência 1:N entre objetos, de tal forma que, quando um objeto (assunto) tem seu estado modificado, os seus objetos dependentes (observadores) são notificados.
  - Assunto → subject
  - Observadores (objetos dependentes) → observers

Observer (motivação)

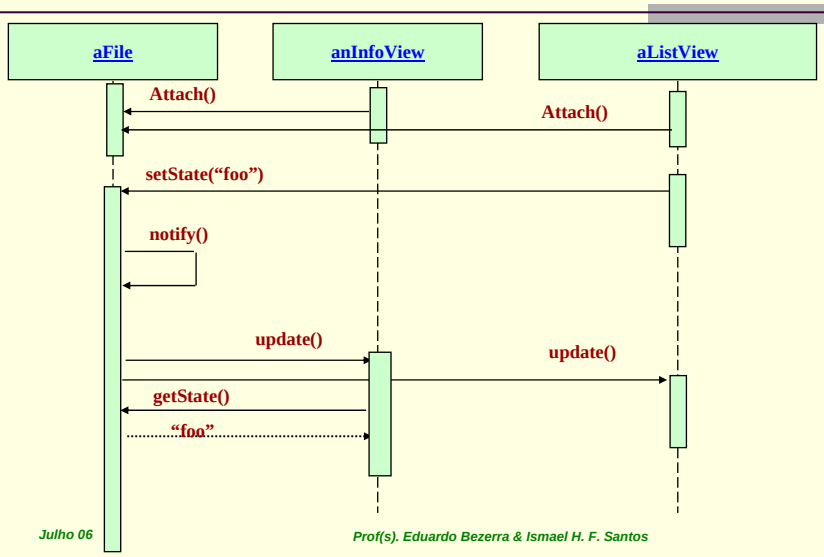
- Exemplo clássico



Observer (estrutura)



Observer (exemplo de interação)



Observer em Java

```
public class Observable extends Object {
   Collection<Observer> observers;
   public void addObserver(Observer o);
   public void deleteObserver(Observer o);
   public boolean hasChanged();
   public void notifyObservers();
   public void notifyObservers(Object arg);
   …
}

public abstract interface Observer {
   public abstract void update(Observable o, Object arg);
}

public class Subject extends Observable{
   public void setState(String filename);
  public string getState();
}
```

Observer (aplicabilidade)

- Quando uma abstração tem dois aspectos, um dependente do outro. Encapsulando-se esses aspectos em objetos separados fará com que se possa variá-los e reusá-los independentemente;
- Quando uma mudança em um objeto requer uma mudança em outros, e não se sabe como esses outros objetos efetivamente fazem suas mudanças;
- Quando um objeto deve poder notificar outros objetos sem assumir nada sobre eles. Dessa forma evita-se que os objetos envolvidos fiquem fortemente acoplados.

Observer (conseqüências)

- Possibilita baixo acoplamento entre os objetos dependentes (os observadores) e o assunto.
- Acoplamento Abstrato
- Suporte para *broadcast*
- Dificuldade em saber o que foi mudado?