Partie I : Programmation graphique en Swing

Programmation objet 2 (2012-2013)

Chapitre 5 MVC

- Design pattern
- Observable-Observer
- MVC et Observable-Observer
- Exemples

Design pattern

- Qu'est-ce qu'un design pattern?
  - Un design pattern est un modèle (patron, motif) de conception qui décrit une solution standard, utilisable dans la conception de différents logiciels

Design pattern

- Historique
  - Les design patterns tirent leur origine des travaux de l'architecte en bâtiments Christopher Alexander dans les années 70, dont son livre « A Pattern Language » définissant un ensemble de modèles d'architecture
  - Les design patterns sont formalisés dans le livre de Erich Gamma, Richard Helm, Ralph Johnson et John Vlissides, intitulé « Design Patterns – Elements of Reusable Object-Oriented Software » 1995

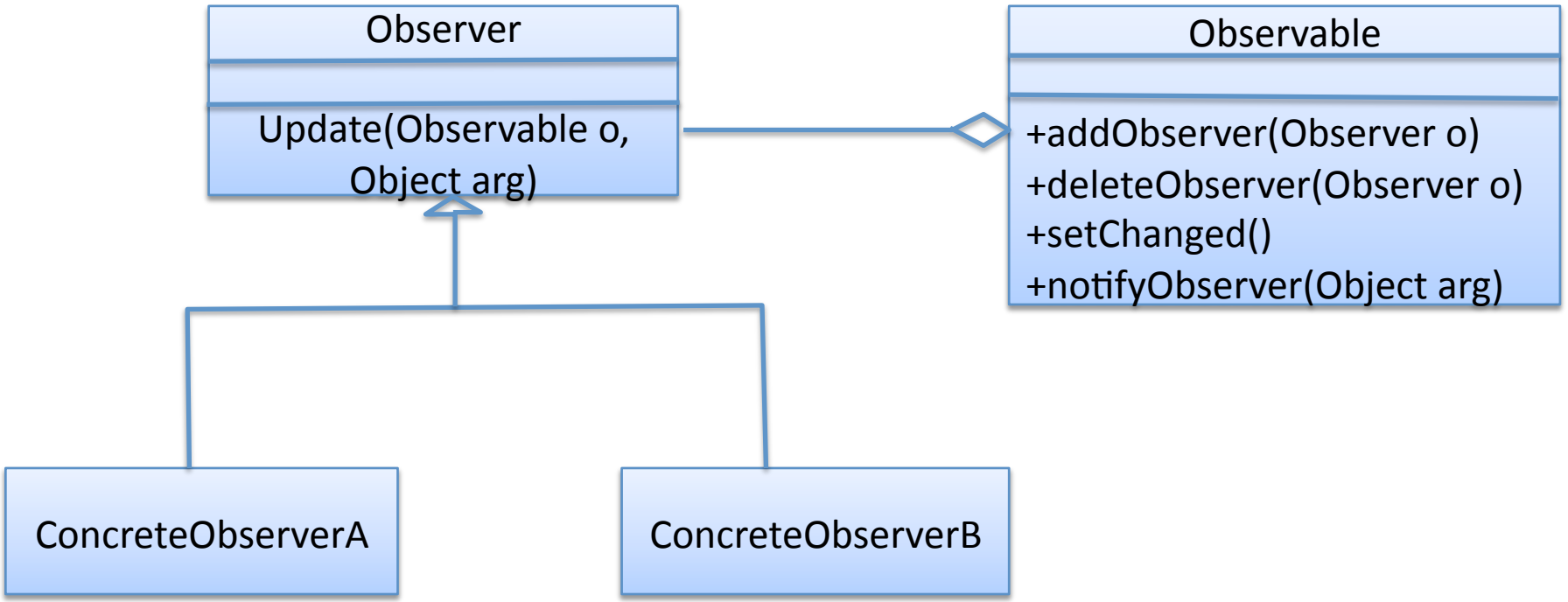
Design pattern

- Description d'un design pattern
  - Nom
  - Description du problème à résoudre
  - Description de la solution
    - Les éléments avec leurs relations de la solution, représentée avec un diagramme UML

Observable-Observer

- Problème
  - On souhaite gérer des événements
- Solution
  - L'objet observé (Observable)
    - la source d'un événement qui contient une liste d'observateurs, ainsi à l'aide d'une méthode de notification l'ensemble des observateurs est prévenu
  - L'objet observateur (Observer)
    - chargé de traiter des événements

Observable-Observer



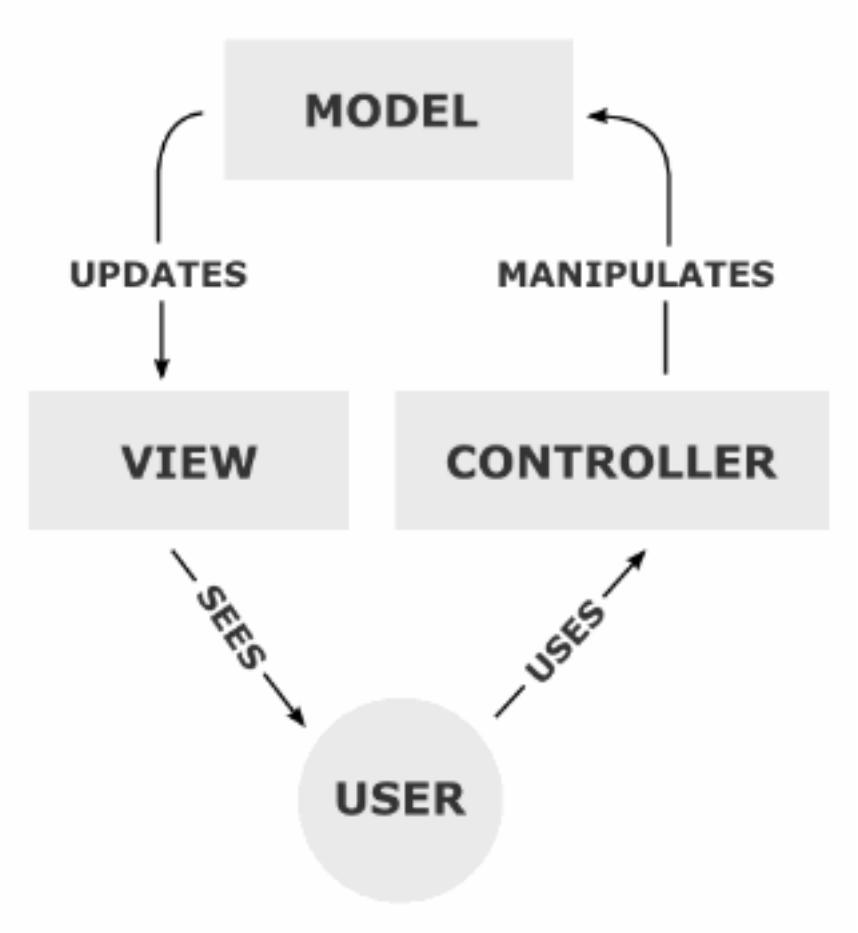
MVC (Model/View/Controller)

- Afin de séparer les responsabilités des différents composants d'une GUI, il est nécessaire de disposer d'un moyen de synchroniser les données avec leur affichage, et gérer les actions de l'utilisateur pour agir sur ces données

MVC

- Problème
  - On souhaite réaliser une GUI souple qui peut avoir plusieurs vues, ou plusieurs contrôleurs
- Solution
  - Le Modèle contient les données et les traitements des données
  - La Vue affiche les résultats des traitements effectués par le modèle
  - Le Contrôleur analyse la requête de l'utilisateur et demande au modèle d'effectuer les traitements

MVC

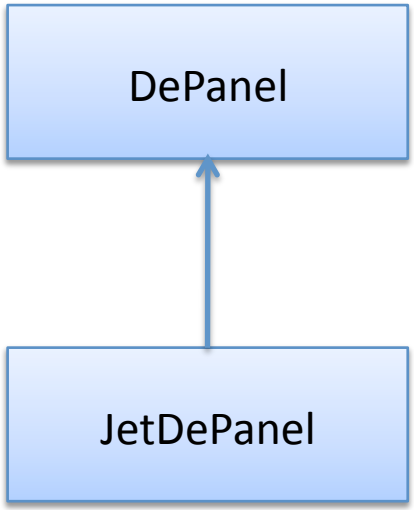


MVC et Observable-Observer

- Le MVC peut être réalisé par le pattern Observable-Observer
  - Modèle = Observable
  - Vue = Observer

Exemple : sans MVC

Pour les exemples JetDeEvent1, JetDeEvent2
- Le modèle et l'affichage ne sont pas séparés



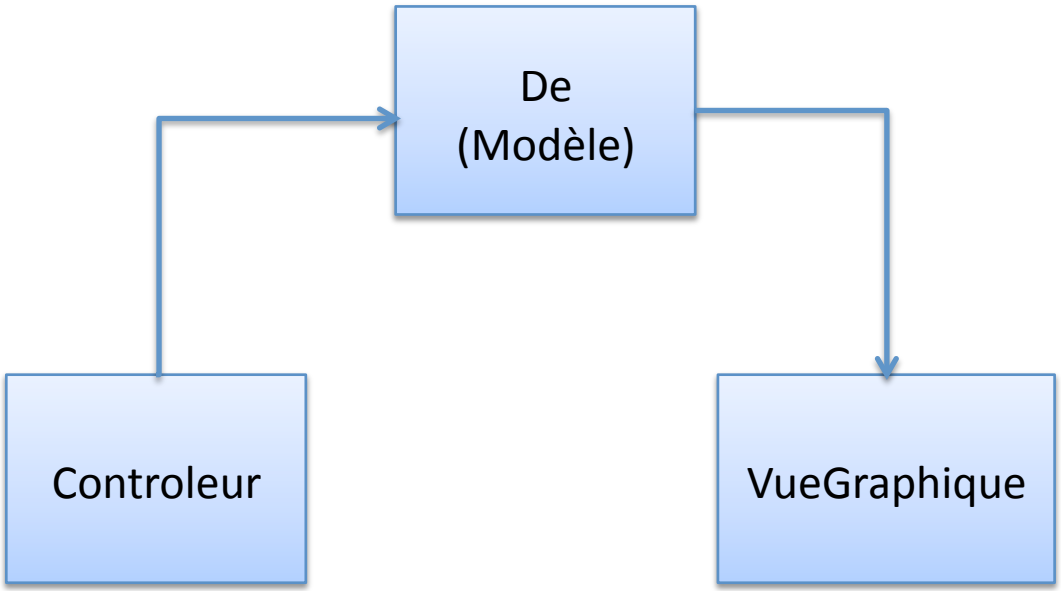
Exemple : sans MVC

```
public class DePanel extends JPanel {
    private static final int SPOT_DIAMETER = 10;
    private int val;
    public DePanel() {
        setBackground(Color.white);
        setPreferredSize(new Dimension(80,80));
        val = (int)(Math.random()*6)+1;
    }
    public int getVal() {
        return val;
    }
    public void setVal(int val) {
        this.val = val;
    }
    public void jet() {
        val = (int)(6*Math.random() + 1);
        repaint();
    }
    public void paintComponent(Graphics g) {
     ……
    }
    private void drawSpot(Graphics g, int x, int y) {
     ……
    }
}
```

Exemple : avec MVC

- MVCJetDe1

Le modèle et l'affichage sont séparés



Exemple : avec MVC

```
public class De extends Observable {
  private VueGraphique v;
  private int val=0;
  public De() {
    val = (int)(Math.random()*6)+1;
  }
  public De(int val) {
    this.val = val;
  }
  public De(De d) {
    val = d.val;
  }
  public int getVal() {
    return val;
  }
  public void jet(){
    val = (int)(Math.random()*6)+1;
    setChanged();
    notifyObservers(new Integer(val));
  }
  public boolean equals(Object obj) {
   ……
  }
}
```

Exemple : avec MVC

```
public class VueGraphique extends JPanel implements
Observer {
    private static final int SPOT_DIAMETER = 10;
    private int val;
    public VueGraphique() {
     setBackground(Color.white);
     setPreferredSize(new Dimension(100,100));
    }
    public void update(Observable o, Object arg) {
         Integer valObj = (Integer) arg;
         val = valObj.intValue();
         repaint();
    }

    public void paintComponent(Graphics g) {
     ……
    }
    private void drawSpot(Graphics g, int x, int y) {
     ……
    }
```

Exemple : avec MVC

```
public class JetDeGUI extends JFrame {
    public JetDeGUI() {
        this.setTitle("Jetter-Dés MVC");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        De m = new De();
        VueGraphique v1 = new VueGraphique();
        m.addObserver(v1);
        Controleur c = new Controleur();
        c.setModel(m);
        this.setLayout(new BorderLayout());
        this.add(c, BorderLayout.NORTH);
        this.add(v1, BorderLayout.CENTER);
        this.pack();
        this.setVisible(true);
    }
   public static void main(String[] args) throws Exception {
    ……
   }
}
```

Exemple : avec MVC

- MVCJetDe2

Ajouter une autre vue : VueTexte