

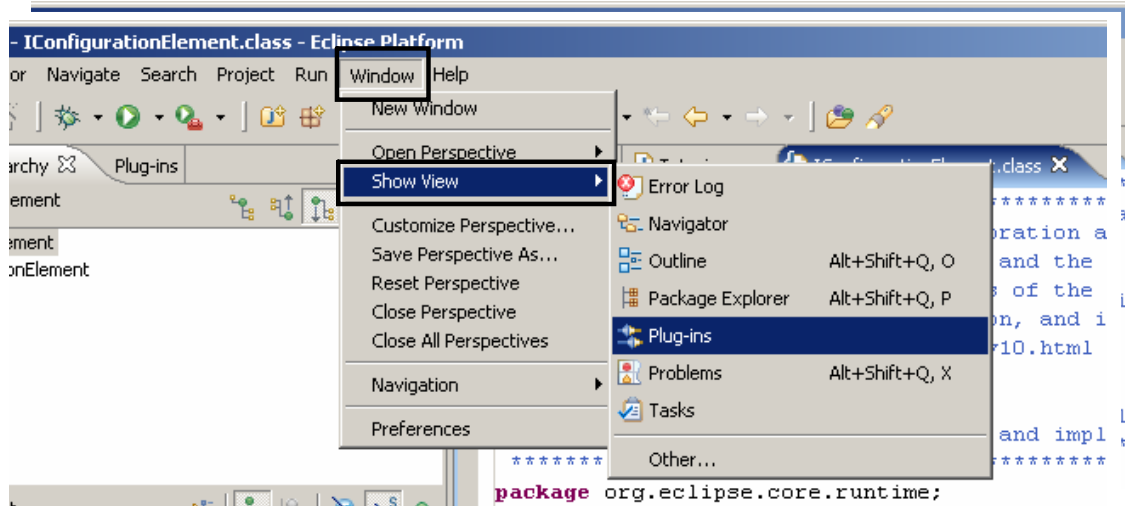
Les views

LES VIEWS.....	1
CONTENU DE LA SECTION	2
LA CLASSIFICATION DES VUES	3
<i>les categories.....</i>	3
LA DECLARATION D'UNE VIEW (1)	4
<i>la marche à suivre</i>	4
L'IMPLEMENTATION D'UNE VIEW (1)	5
<i>l'interface à implémenter : l'interface IViewPart.....</i>	5
<i>quelques méthodes à définir.....</i>	5
<i>un exemple minimal.....</i>	5
L'IMPLEMENTATION D'UNE VIEW (2)	6
<i>la synchronisation des Views</i>	6
<i>la synchronisation avec des vues particulières.....</i>	6
L'IMPLEMENTATION D'UNE VIEW (3)	7
<i>un exemple simplissime avec synchronisation.....</i>	7
<i>la gestion des événements.....</i>	7
L'IMPLEMENTATION D'UNE VIEW (4)	8
<i>Le plugin et la communication des événements.....</i>	8
L'IMPLEMENTATION D'UNE VIEW (5)	9
<i>les Views.....</i>	9
COMPLEMENTS SUR LES VIEWS	10
<i>quelques Views prédéfinis.....</i>	10
<i>ajout d'une View dans la FastView</i>	10

La classification des vues

les categories

- classification des views en category accessible à partir de **Window > Show View**
- possibilité de définir une nouvelle category
- le menu ne contient qu'un sous ensemble des views (item Other)

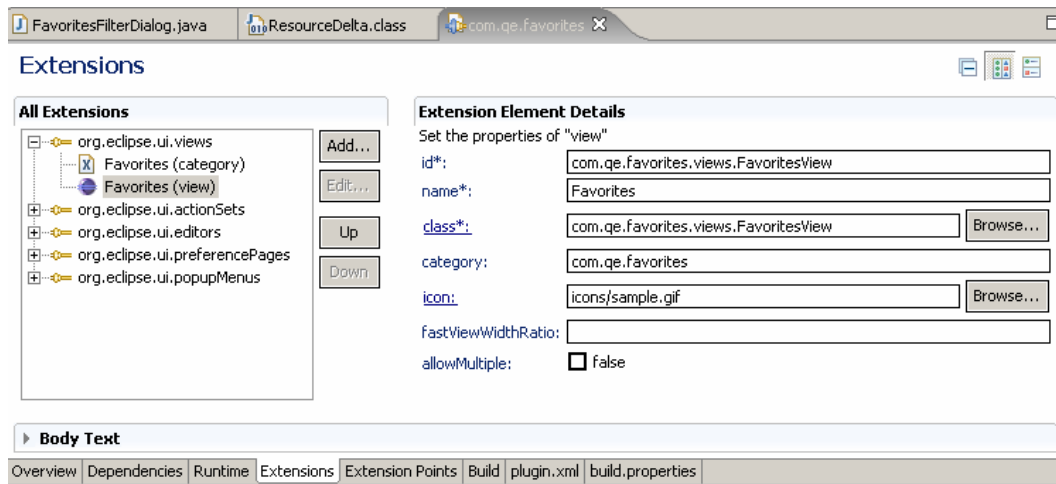


La déclaration d'une View (1)

la marche à suivre

- Sélectionner **Extensions**
- Cliquer sur **Add**
- Sélectionner **org.eclipse.ui.views**
- Sélectionner **New > category** [éventuellement] et remplir le formulaire associé
- Sélectionner **New > view** et remplir le formulaire associé

si on désire créer une nouvelle catégorie de views (qui apparaît dans Show View)



L'implémentation d'une View (1)

l'interface à implémenter : l'interface IViewPart

- une View implémente **IViewPart** (qui étend **IWorkbenchPart**)
- une View étend souvent directement **ViewPart** (ou des classes plus spécifiques)

quelques méthodes à définir

```
void init(IViewSite site)
void createPartControl(Composite parent)
void setFocus()
```

gère la partie graphique et installe généralement les listeners (voir suite)

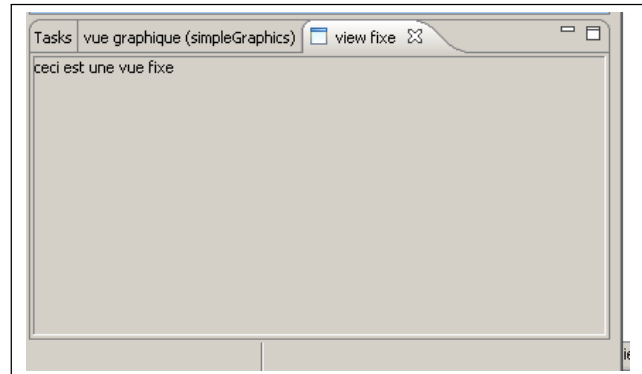
un exemple minimal

```
import org.eclipse.swt.SWT;
.....

public class TP2F extends ViewPart {

    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        Label lab1 = new Label(parent, SWT.BORDER);
        lab1.setText("ceci est une vue fixe");
    }

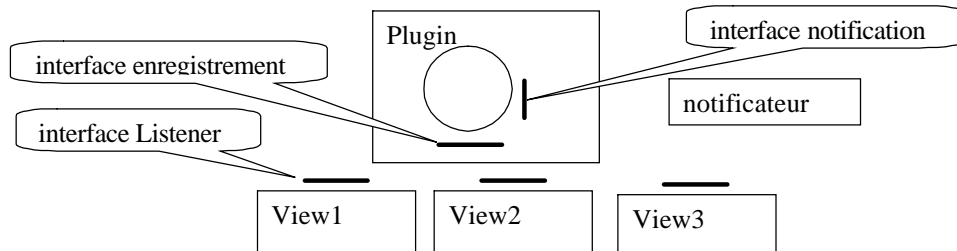
    public void setFocus() {}
}
```



L'implémentation d'une View (2)

la synchronisation des Views

- par le biais du DP Observer : les Views sont des Listeners d'un modèle de données commun
- le service de notification est souvent attaché au plugin



la synchronisation avec des vues particulières

- la synchronisation avec le Navigator via l'observateur **SelectionService** et l'interface **ISelectionListener**

```
getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(this);
```

- la synchronisation avec la View : "Properties" via l'interface **IInputSelectionProvider**

L'implémentation d'une View (3)

un exemple simplissime avec synchronisation

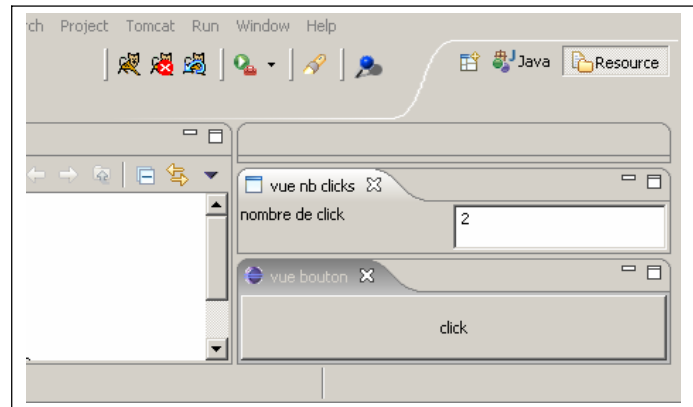
- vue avec un bouton
- vue avec un compteur de click

la gestion des événements

```
public interface Tp1ValuesListener {
    void valuesChanged(Tp1ValuesEvent evt);
}
```

```
public class Tp1ValuesEvent {
    public int[] values;

    public Tp1ValuesEvent(int[] aV) { values = aV; }
}
```



événement = simple conteneur typé d'information

L'implémentation d'une View (4)

Le plugin et la communication des événements

```
public class Tp1Plugin extends AbstractUIPlugin {
```

```
.....  
ArrayList listeners;
```

enregistrement d'un observateur

```
public void addTp1ValuesListener(Tp1ValuesListener aL) {  
    if (aL == null) return;  
    if (listeners == null) listeners = new ArrayList();  
    if (listeners.contains(aL)) return;  
    listeners.add(aL);  
}
```

désenregistrement d'un observateur

```
public void removeTp1ValuesListener(Tp1ValuesListener aL) {  
    if (aL == null) return;  
    if (listeners == null) return;  
    listeners.remove(aL);  
}
```

notification des observateurs

```
public void notifyValuesChange(Tp1ValuesEvent aEvt) {  
    if (listeners == null) return;  
    int size = listeners.size();  
    for (int i = 0; i < size; i++) { ((Tp1ValuesListener) listeners.get(i)).valuesChanged(aEvt);}  
}  
.....
```

L'implémentation d'une View (5)

les Views

```
public class Tp1View extends ViewPart {
    private int values[] = new int[1];
    Tp1Plugin plugin = Tp1Plugin.getDefault();

    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        Button but1 = new Button(parent, SWT.PUSH);
        but1.setText("click");
        but1.addSelectionListener(new SelectionListener() {
            public void widgetSelected(SelectionEvent e) {
                values[0]++;
                plugin.notifyValuesChange(new Tp1ValuesEvent(values));
            }
            public void widgetDefaultSelected(SelectionEvent e) {}
        });
    }

    public void setFocus() {}
}
```

```
public class Tp1NbClick extends ViewPart
    implements Tp1ValuesListener {
    private Text nbClick;

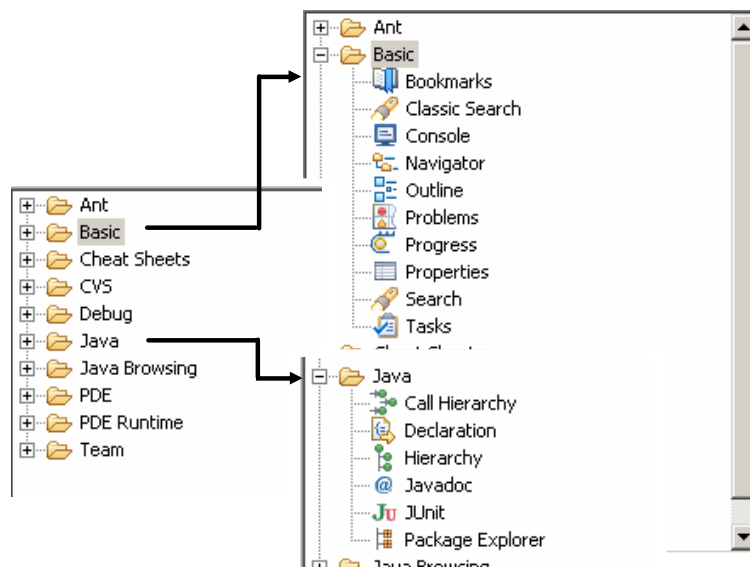
    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        Label lab1 = new Label(parent, SWT.NONE);
        lab1.setText("nombre de click");
        nbClick = new Text(parent, SWT.BORDER);
        nbClick.setText(" 0");
        Tp1Plugin.getDefault().addTp1ValuesListener(this);
    }

    public void valuesChanged(Tp1ValuesEvent evt) {
        nbClick.setText("" + evt.values[0]);
    }

    public void setFocus() {}
}
```

Compléments sur les Views

quelques Views prédéfinis



ajout d'une View dans la FastView

- via les perspectives