

La gestion des préférences et les Wizards

LA GESTION DES PREFERENCES ET LES WIZARDS	1
CONTENU DE LA SECTION	2
LA GESTION DES PREFERENCES ET LES WIZARDS	3
<i>le principe général</i>	3
LA DECLARATION DE PAGES DE PREFERENCES	4
<i>la marche à suivre</i>	4
L'IMPLEMENTATION DE PAGES DE PREFERENCES (1).....	5
<i>l'interfaces à implémenter : IWorkbenchPreferencePage</i>	5
L'IMPLEMENTATION DE PAGES DE PREFERENCES (2).....	6
<i>la construction d'une page avec la classe FieldEditorPreferencePage</i>	6
<i>un exemple simplissime</i>	6
LA GESTION DU PREFERENCESTORE.....	7
<i>la portée des préférences</i>	7
<i>le PreferenceStore</i>	7
<i>divers</i>	7
LA DECLARATION D'UN WIZARD (1).....	8
<i>la marche à suivre</i>	8

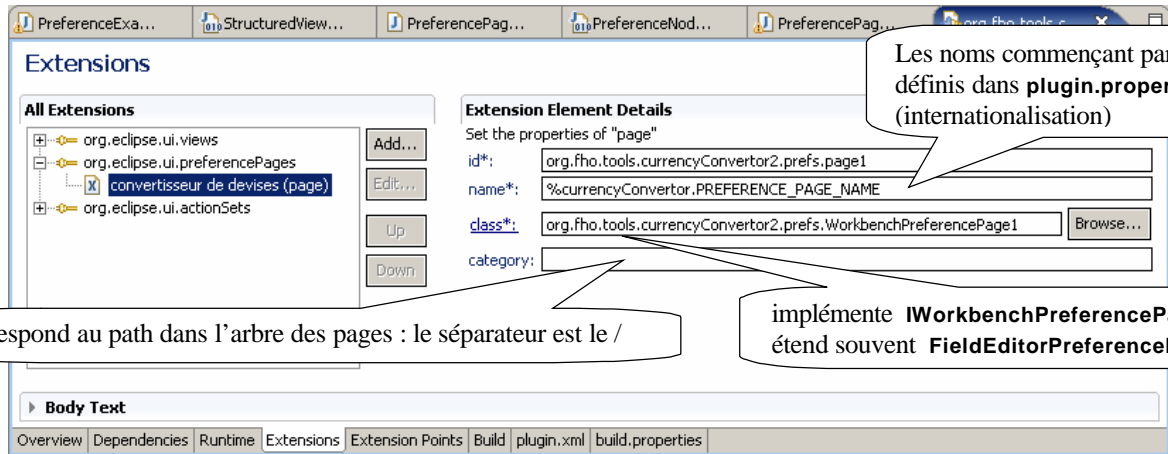
le principe général

- la gestion des préférences d'Eclipse repose sur la gestion préférences JFace
 - PreferenceStore, PreferenceManager intégrés
 - gestion des PreferenceNode masquée
 - il suffit de construire les PreferencePage
- les Wizards d'Eclipse sont des Wizards JFace
 - Wizard (le moteur d'enchaînement) masqué
 - il suffit de construire les WizardPage

La déclaration de pages de Preferences

la marche à suivre

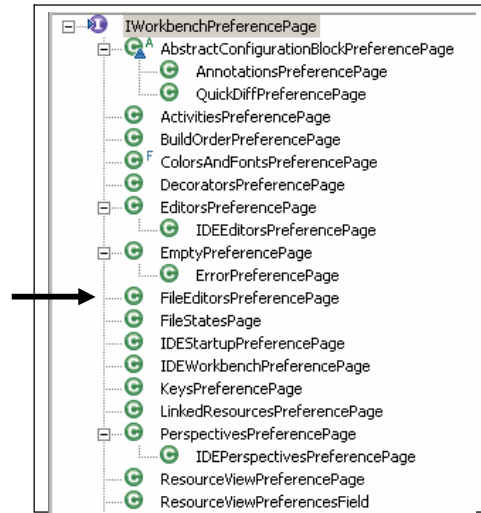
- Sélectionner **Extensions**
- Cliquer sur **Add**
- Sélectionner **org.eclipse.ui.preferencePages**
- Sélectionner **New > page** et remplir le formulaire associé



L'implémentation de pages de Preferences (1)

l'interface à implémenter : IWorkbenchPreferencePage

- Souvent : **extends PreferencePage implements IWorkbenchPreferencePage**
- principes méthodes (à définir) :
void init(IWorkbench wbench)
Control createControl(Composite parent)
boolean isValid(), boolean okToLeave()
void performOK(), void performCancel(), void performHelp()
void setTitle()



L'implémentation de pages de Preferences (2)

la construction d'une page avec la classe FieldEditorPreferencePage

- page spécialisée prévue pour être construite par simple ajout de **FieldEditor**
BooleanFieldEditor, **IntegerFieldEditor**, **FloatFieldEditor**,
ColorFieldEditor,

un exemple simplissime ...

```
public class RootPrefPage extends FieldEditorPreferencePage implements IWorkbenchPreferencePage {
    private IntegerFieldEditor v1FE;
    private IntegerFieldEditor v2FE;
    private IntegerFieldEditor v3FE;

    public RootPrefPage() { super("Preferences du modele", GRID); }

    protected void createFieldEditors() {
        v1FE = new IntegerFieldEditor("org.fho.tools.ex.first", "premiere valeur : ", getFieldEditorParent());
        v1FE.setStringValue("15");
        addField(v1FE);
        v2FE = new IntegerFieldEditor("org.fho.tools.ex.first ", "seconde valeur : ", getFieldEditorParent());
        v1FE.setStringValue("20");
        addField(v2FE);
    }
    .....
}
```

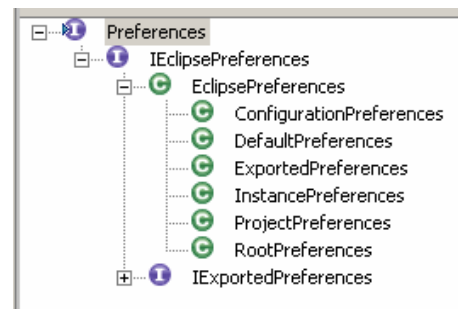
La gestion du PreferenceStore

la portée des préférences

- Instance (associée à un Workspace ou une instance de la plateforme)
- Configuration (commune aux différents Workspaces)
- Default

le PreferenceStore

- Attention : la manipulation explicite du PreferenceStore est déprécié
- obtention du PreferenceService à partir de la classe Platform
IPreferenceService srv = Platform.getPreferencesService()
boolean srv.get<TYPE>("pluginID", "preferenceKey", defaultValue, null);
- obtention à partir du Plugin :
 obtention des Preferences : **XXXPlugin.getDefault().getPreferences()**



divers

- observation des modifications des préférences :
EclipsePreferences prefs =
prefs.addPreferenceChangeListener(this); ← objet qui implémente interface **IPreferenceChangeListener**

La déclaration d'un Wizard (1)

la marche à suivre

- définition de nouveaux Wizards pour la création, l'import, l'export de nouveaux types de ressources
- Sélectionner **Extensions**
- Cliquer sur **Add**
- Sélectionner **org.eclipse.ui.newWizards** ou **org.eclipse.ui.importWizards** ou **org.eclipse.ui.exportWizards**
- Sélectionner **New > wizard** et remplir le formulaire associé

