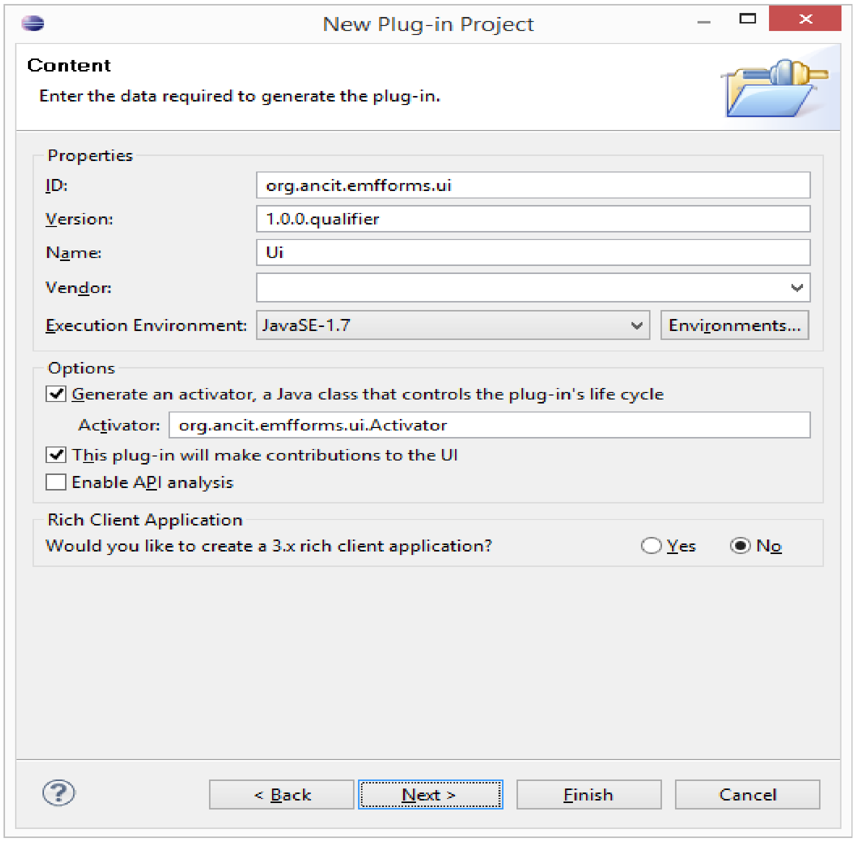
Adding a FormPage to EMF Form Editor

In the previous tutorial we learnt how to customize the look of the editor rendered by ECP Forms using View Model. In this tutorial we have documented the steps for adding a new custom Tab to the EMF Form Editor.

CREATE A Eclipse Plugin for UI Development:

1. We will create a new Plugin Project with Contribution to UI enabled.



ID: org.ancit.emfforms.ui

Version: 1.0.0.qualifier

Name: Ui

Vendor:

Execution Environment: JavaSE-1.7

☑ Generate an activator, a Java class that controls the plug-in's life cycle

Activator: org.ancit.emfforms.ui.Activator

☑ This plug-in will make contributions to the UI

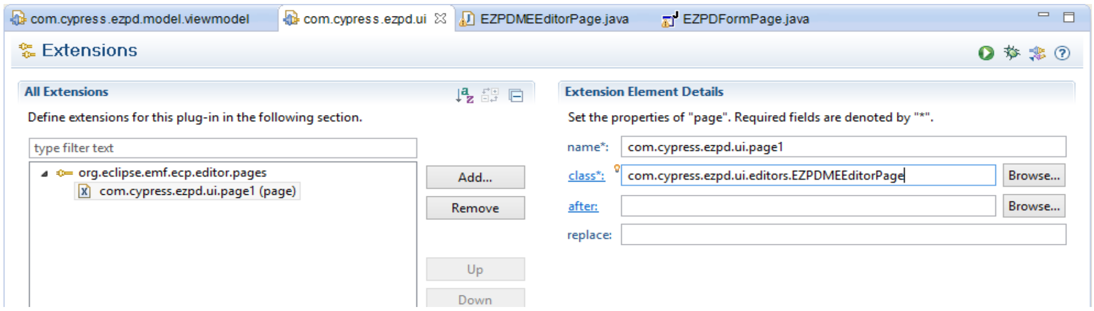
☐ Enable API analysis

Would you like to create a 3.x rich client application? ○ Yes ● No

2. We are interested in customizing the EMF Form Editor. Therefore you need to include **org.eclipse.emf.ecp.editor.e3** plugin in your dependancies.

3. EMF Form Editor is an extension of Form Editors. Therefore to add a new tab, we actually need to create a FormPage. Therefore we also need to add **org.eclipse.ui.forms** along with **org.eclipse.emf.ecore**.

4. In the Extensions Tab, we need to add an extension point org.eclipse.emf.ecp.editor.pages. Fill in the Name and the Class. Class should extend AbstractMEEditorPage.



Define extensions for this plug-in in the following section.

- org.eclipse.emf.ecp.editor.pages
  - com.cypress.ezpd.ui.page1 (page)

Set the properties of "page". Required fields are denoted by "*".

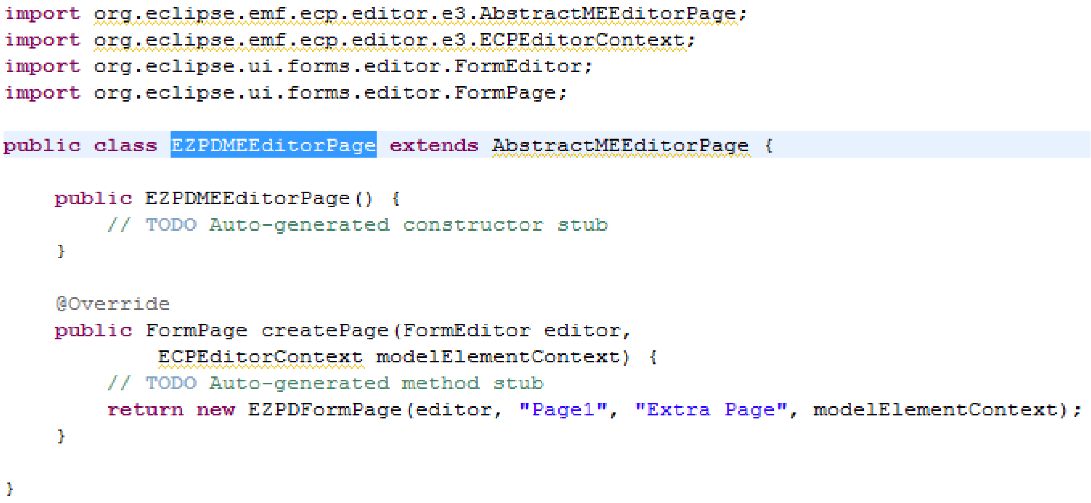
name*: com.cypress.ezpd.ui.page1

class*: com.cypress.ezpd.ui.editors.EZPDMEEditorPage

after:

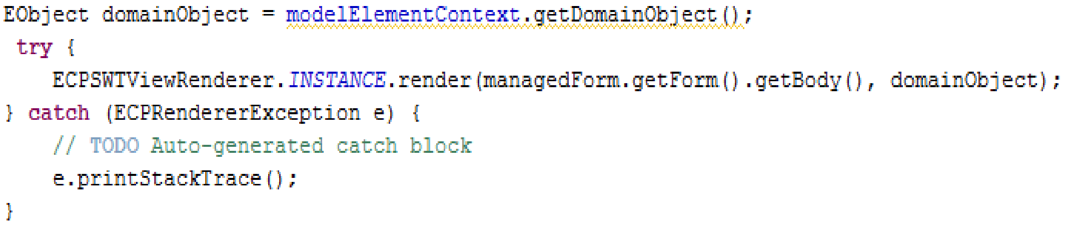
replace:

5. In our class, you will have a method named createPage(). Create a new FormPage and return the same as shown below



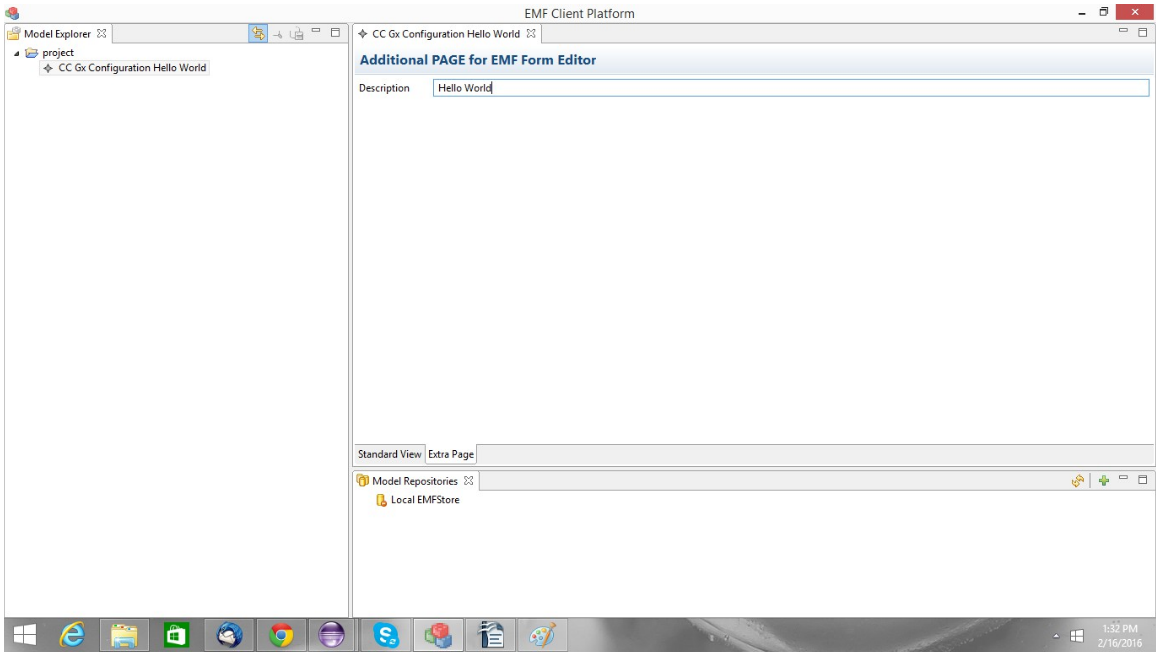
```
import org.eclipse.emf.ecp.editor.e3.AbstractMEEditorPage;
import org.eclipse.emf.ecp.editor.e3.ECPEditorContext;
import org.eclipse.ui.forms.editor.FormEditor;
import org.eclipse.ui.forms.editor.FormPage;

public class EZPDMEEditorPage extends AbstractMEEditorPage {

    public EZPDMEEditorPage() {
        // TODO Auto-generated constructor stub
    }

    @Override
    public FormPage createPage(FormEditor editor,
            ECPEditorContext modelElementContext) {
        // TODO Auto-generated method stub
        return new EZPDFormPage(editor, "Page1", "Extra Page", modelElementContext);
    }

}
```

6. YourFormPage class will extend from FormPage and in the createFormContent we use a ECPSWTViewRenderer to render the desired UI as shown below



```
EObject domainObject = modelElementContext.getDomainObject();
 try {
    ECPSWTViewRenderer.INSTANCE.render(managedForm.getForm().getBody(), domainObject);
} catch (ECPRendererException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

7. Now run your ECP Application and you will find a Custom Page contributed to your EMF Form Editor.



Additional PAGE for EMF Form Editor

Description: Hello World

REFERENCES:

http://eclipse.org/ecp/
http://eclipsesource.com/blogs/tutorials/getting-started-with-the-emf-client-platform/
http://eclipsesource.com/blogs/tutorials/emf-forms-renderer/

ABOUT ANCIT:

ANCIT Consulting is an Eclipse Consulting Firm located in the "Silicon Valley of Outsourcing", Bangalore. Offers professional Eclipse Support and Training for various Eclipse based Frameworks including RCP, EMF, GEF, GMF. Contact us on annamalai@ancitconsulting.com to learn more about our services.

Author: Annamalai
February 16, 2016