tutorial : eclipse rcp e4 with 3.x views like project explorer, properties, etc.

Posted on **August 1, 2012**

This tutorial shows step-by-step how to add classical Eclipse 3.x views like Project Explorer and the Properties View to an Eclipse e4 Rich Client Platform (RCP) Application.

Eclipse 4.2 was released a few weeks ago with the new e4 platform that basically uses an EMF model to describe how your application looks and feels. While this is neat, the caveat is that e4 currently (August 2012) does not provide views for a number of core features of the old Eclipse 3.x platform including things like the Project Explorer, a Properties View and the like.

Eclipse 4.2 comes with a “Compatibility Layer” that makes all the old features that were not ported yet to e4 available. However, I could not find a site that tells me how to use it. There are a few quite instructive tutorials out there on how RCP applications with e4 work, for instance http://www.vogella.com/articles/EclipseRCP/article.html. But none showed me how to add a Project Explorer and all the IDE views that I needed for my projects (http://service-technology.org/seda and http://service-technology.org/greta in case you want to know). So I went for the usual approach in the Eclipse-verse: trial and error. Here is what I figured out and works for me.

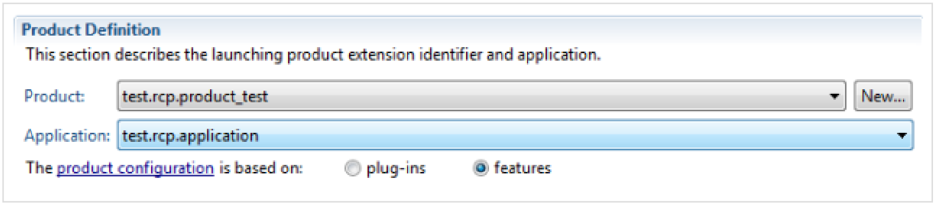
For the rest of this tutorial I assume you are familiar with building Eclipse 3.X RCP applications. Please consult a more complete tutorial such as http://www.vogella.com/articles/EclipseRCP/article.html to get into the topic or refresh some details. To get a code-base to try out the things I talk about next, create a new RCP project using *File > New Project > Plug-In Project*, give it a name (let’s call it `test.rcp` here), and pick the *Hello RCP* Template to create the project.

1. **To use the “Compatibility Layer” of Eclipse 4, do not run an e4-based application, but run a classical Eclipse application.** An Eclipse RCP consists of two things: an application, and a product. The product defines the look and feel of the RCP, the application provides the “framework” on which this look and feel can be pinned. e4 does not allow to pin the “Compatibility Layer”. Instead, we use a classical application (backed by Eclipse 4.X plugins) and add an “**e4 aware product**” on top of it. This solution has the drawback that some parts of your RCP application have to be defined in code. Yet some parts can be defined by the e4 model and become a lot easier.
2. **Add an e4 application model file to the RCP project**: *File > New > Eclipse 4 > Model > New Application Modeltest.rcp as container, give it the name* `Application.e4xmi`, *and check Include Default Addons. In this file we will later put all the fancy model definitions of our RCP application. But first we have to prepare a few other things.*
3. **Register a new product definition in the RCP project**: edit `plugin.xml` and insert the following product definition

```
<extension id="product_test" point="org.eclipse.core.runtime.products">
  <product application="test.rcp.application" name="My Product">
    <property name="appName" value="My App"> </property>
    <property name="applicationXMI" value="test.rcp/Application.e4xmi"> </property>
  </product>
</extension>
```

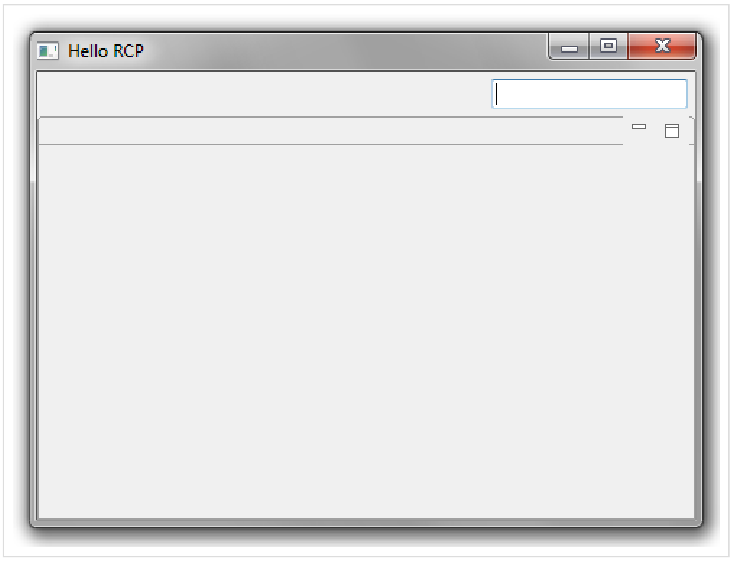
   where `test.rcp.application` refers to the Eclipse 3.X application defined in the RCP plugin at the `org.eclipse.core.runtime.applications` extension point, and `test.rcp/Application.e4xmi` refers to the application model file we just created.

4. **Create a new product configuration file** using the *File > New > Plug-In Development > Product Configuration* wizard. Put the file in your RCP plugin (`test.rcp`), give it a name (`test.rcp.product`), and *Initialize the file content* using the *existing product* we just defined: `test.rcp.product_test` (where `product_test` stems from the `id` given in the extension point. You can configure various things like names and ids etc. The most important settings are the following.
5. **Link the product to Eclipse 3.X application** by setting in `test.rcp.product` on the Overview Tab the *Application* to `test.rcp.Application` (as in the `plugin.xml`).


Product Definition
This section describes the launching product extension identifier and application.
Product: test.rcp.product_test
Application: test.rcp.application
The product configuration is based on: ( ) plug-ins (●) features

6. **Add required Eclipse 3.X and Eclipse e4 features to your product.** To run the product, you will need a bunch of plugins and features from Eclipse which you can configure in the product configuration. Make it based on *features* (there are other solutions, but I prefer this one), open the *Dependencies* tab, add `org.eclipse.e4.rcp` and click the *Add Required* button which will add `org.eclipse.emf.core` and `org.eclipse.emf.common` to the product configuration.
7. **Define a Main Application Window**. In the old eclipse world you would now be good to go and run the product. With e4, you aren’t. We have to define the look and feel of the application in the `Application.e4xmi`. Open the file and at *Application > Windows* add a new *TrimmedWindow*. You can configure various things here, but we will leave it for now.

8. **Run the application** to test if things are good so far: right click on `test.rcp.product` and select *Run As > Eclipse Application*. The launch will probably fail, edit the launch configuration in *Run > Run Configurations…*, open the *Plugins* tab, select *Launch with 'plug-ins selected below only*, check your RCP plugin `test.rcp` and click on *Add Required Plug-ins* to get all Eclipse plugins that are needed to run your application. When you click on *Run* an application windows looking like this should show up.



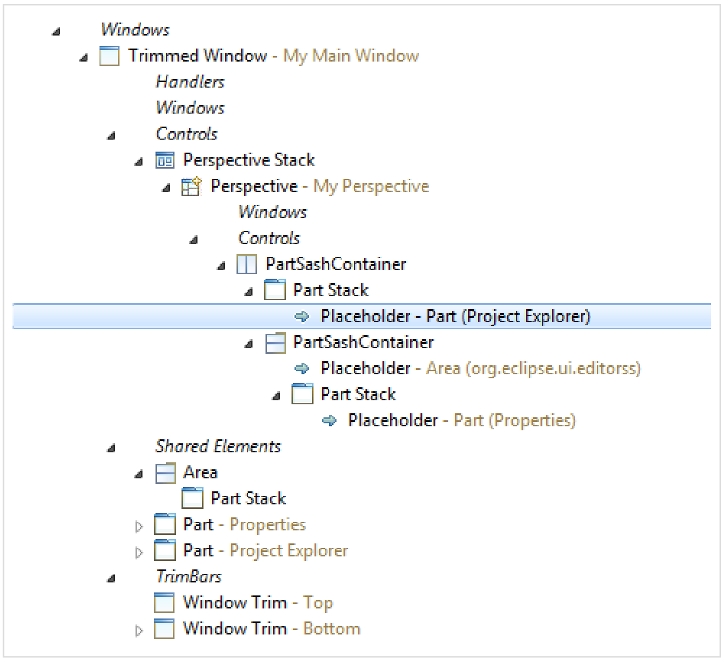
In case your product does not launch and raises further errors, check whether there are *VM arguments* of your launch configuration that conflict with the available Java runtime.

9. **Define 'shared elements' that access Eclipse 3.x compatibility layer**. Now comes the first step to actually enable Eclipse 3.x in your application. As the current editor for the `Application.e4xmi` has some limitations, we have to do some xml editing. Take a short-cut and replace the contents of your `Application.e4.xmi` with this code:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<application:Application xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="
  <children xsi:type="basic:TrimmedWindow" xmi:id="_Vnmah9SzEeGBn6dQ9VPexA" label="My Mai
    <children xsi:type="advanced:PerspectiveStack" xmi:id="_Vnmaj9SzEeGBn6dQ9VPexA" selec
      <children xsi:type="advanced:Perspective" xmi:id="_VnmakNSzEeGBn6dQ9VPexA" elementI
        <tags>Perspective</tags>
        <tags>categoryTag:General</tags>
        <children xsi:type="basic:PartSashContainer" xmi:id="_VnnBgNSzEeGBn6dQ9VPexA" hor
          <children xsi:type="basic:PartStack" xmi:id="_VnnBgdSzEeGBn6dQ9VPexA" container
            <tags>newtablook</tags>
            <tags>org.eclipse.e4.primaryNavigationStack</tags>
            <children xsi:type="advanced:Placeholder" xmi:id="_QR29cNS2EeGBn6dQ9VPexA" el
          </children>
          <children xsi:type="basic:PartSashContainer" xmi:id="_bTQ1YNS2EeGBn6dQ9VPexA" e
            <children xsi:type="advanced:Placeholder" xmi:id="_hP-EYNS2EeGBn6dQ9VPexA" el
            <children xsi:type="basic:PartStack" xmi:id="_qAHrYNS2EeGBn6dQ9VPexA" element
              <tags>newtablook</tags>
              <tags>org.eclipse.e4.secondaryDataStack</tags>
              <children xsi:type="advanced:Placeholder" xmi:id="_soNH8NS2EeGBn6dQ9VPexA"
            </children>
          </children>
        </children>
      </children>
    </children>
    <sharedElements xsi:type="advanced:Area" xmi:id="_Hmg4kNSjEeGwXp2p959l3w" elementId="
      <children xsi:type="basic:PartStack" xmi:id="_HmhfoNSjEeGwXp2p959l3w" elementId="or
        <tags>newtablook</tags>
        <tags>org.eclipse.e4.primaryDataStack</tags>
        <tags>EditorStack</tags>
      </children>
    </sharedElements>
    <sharedElements xsi:type="basic:Part" xmi:id="_WqRGcNS1EeGBn6dQ9VPexA" elementId="org
      <tags>View</tags>
      <tags>categoryTag:General</tags>
      <menus xmi:id="_HngXENSjEeGwXp2p959l3w" elementId="org.eclipse.ui.views.PropertyShe
        <tags>ViewMenu</tags>
        <tags>menuContribution:menu</tags>
      </menus>
      <toolbar xmi:id="_Hng-INSjEeGwXp2p959l3w" elementId="org.eclipse.ui.views.PropertyS
    </sharedElements>
    <sharedElements xsi:type="basic:Part" xmi:id="_sS7Q8NS1EeGBn6dQ9VPexA" elementId="org
      <tags>View</tags>
      <tags>categoryTag:General</tags>
      <menus xmi:id="_4UXAUNSmEeG-a6zBTyJ-wg" elementId="org.eclipse.ui.navigator.Project
        <tags>ViewMenu</tags>
        <tags>menuContribution:menu</tags>
      </menus>
      <menus xsi:type="menu:PopupMenu" xmi:id="_4VpZwNSmEeG-a6zBTyJ-wg" elementId="org.ec
        <tags>menuContribution:popup</tags>
        <tags>popup:org.eclipse.ui.navigator.ProjectExplorer#PopupMenu</tags>
      </menus>
      <toolbar xmi:id="_4UXAUdSmEeG-a6zBTyJ-wg" elementId="org.eclipse.ui.navigator.Proje
    </sharedElements>
    <trimBars xmi:id="_VOV6ANTAEeGTY-uOtVc6Mg" elementId="org.eclipse.ui.main.toolbar"/>
    <trimBars xmi:id="_VP1HwNTAEeGTY-uOtVc6Mg" elementId="org.eclipse.ui.trim.status" sid
      <children xsi:type="menu:ToolControl" xmi:id="_VP1u0NTAEeGTY-uOtVc6Mg" elementId="o
        <tags>stretch</tags>
      </children>
```

```
   <children xsi:type="menu:ToolControl" xmi:id="_VQKe8NTAEeGTY-uOtVc6Mg" elementId="o
   <children xsi:type="menu:ToolControl" xmi:id="_VQRMoNTAEeGTY-uOtVc6Mg" elementId="o
  </trimBars>
 </children>
 <bindingTables xmi:id="_VnmafNSzEeGBn6dQ9VPexA" bindingContext="_VnmacdSzEeGBn6dQ9VPexA
 <rootContext xmi:id="_VnmacdSzEeGBn6dQ9VPexA" elementId="org.eclipse.ui.contexts.dialog
  <children xmi:id="_VnmactSzEeGBn6dQ9VPexA" elementId="org.eclipse.ui.contexts.window"
  <children xmi:id="_Vnmac9SzEeGBn6dQ9VPexA" elementId="org.eclipse.ui.contexts.dialog"
 </rootContext>
 <descriptors xmi:id="_9E9jYNS4EeGBn6dQ9VPexA" elementId="org.eclipse.ui.navigator.Proje
  <tags>View</tags>
  <tags>categoryTag:General</tags>
 </descriptors>
 <descriptors xmi:id="_W6yT4NSjEeGgKvWHVYtRxQ" elementId="org.eclipse.ui.console.Console
  <tags>View</tags>
  <tags>categoryTag:General</tags>
 </descriptors>
 <descriptors xmi:id="_W6ziANSjEeGgKvWHVYtRxQ" elementId="org.eclipse.ui.views.ProgressV
  <tags>View</tags>
  <tags>categoryTag:General</tags>
 </descriptors>
 <descriptors xmi:id="_W63zcdSjEeGgKvWHVYtRxQ" elementId="org.eclipse.ui.views.PropertyS
  <tags>View</tags>
  <tags>categoryTag:General</tags>
 </descriptors>
 <addons xmi:id="_VnmadNSzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.core.commands.servic
 <addons xmi:id="_VnmaddSzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.ui.contexts.service"
 <addons xmi:id="_VnmadtSzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.ui.bindings.service"
 <addons xmi:id="_Vnmad9SzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.ui.workbench.command
 <addons xmi:id="_VnmaeNSzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.ui.workbench.context
 <addons xmi:id="_VnmaedSzEeGBn6dQ9VPexA" elementId="org.eclipse.e4.ui.workbench.binding
</application:Application>
```

10. **Understand a basic Application Window layout** This definition does a lot of small things. Technically, a few additional namespaces are made available to the `application:Application` element to use some of the more advanced features. Besides some key binding handlers, the main changes are: **shared elements** for the main window which define some kind of placeholders for UI contributions. These placeholders are used in the definition of the main window.



```
Windows
  Trimmed Window - My Main Window
    Handlers
    Windows
    Controls
      Perspective Stack
        Perspective - My Perspective
          Windows
          Controls
            PartSashContainer
              Part Stack
                Placeholder - Part (Project Explorer)
              PartSashContainer
                Placeholder - Area (org.eclipse.ui.editorss)
                Part Stack
                  Placeholder - Part (Properties)
    Shared Elements
      Area
        Part Stack
      Part - Properties
      Part - Project Explorer
    TrimBars
      Window Trim - Top
      Window Trim - Bottom
```

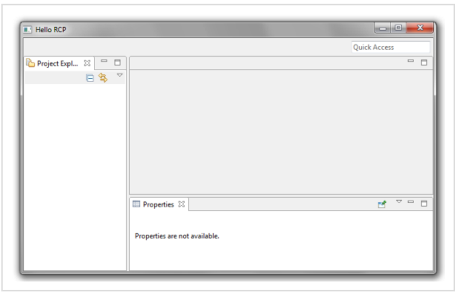
The Main Window defines *Controls*which are all the parts a user can interact with.

A. It's main children is a *Perspective* (also set as the main perspective of that window) that contains two *PartSashContainers.*
B. A *PartSashContainer* splits the window into several areas (one area for each of its child elements) that have an adjustable width or height, depending on the orientation of the *PartSashContainer*.
C. A *PartStack* is an area where several views can be stacked on top of each other, the different views on such a stack that can all be accessed using tabs at the top of the *PartStack* as we know it from the Eclipse IDE.
D. The *Placeholders* refer to the *Shared Elements* that we defined as well.
E. In addition we defined TrimBars, one for the main menu and one for the status bar the bottom for which we also define some default contents such as a progress indicator.
F. The *Shared Elements* themselves are where the magic lies. For instance, the *Project Explorer* is defined by a shared element that defines a *Part* with the `elementId="org.eclipse.ui.navigator.ProjectExplorer"` that is made available by the Compatiblity Layer, defined at the URI `contributionURI="bundleclass://org.eclipse.ui.workbench/org.eclipse.ui.internal.e4.compatibility.Compatibi`

11. To run the new application, we need a few more plugins that provide the code for the Project Explorer etc. Open the `plugin.xml` of your RCP plugin (`test.rcp`) and add the following plugins to its dependencies

A. `org.eclipse.ui.views`
B. `org.eclipse.ui.navigator`
C. `org.eclipse.ui.navigator.resources`

When you run the application, you should now see the following window.



Properties are not available.

Notes:

A. You may have to add more *Required Plugins* to your launch configuration.
B. You may have to delete the file `${runtime-workspace}/.metadata/.plugins/org.eclipse.e4.workbench/workbench.xmi` in the runtime-workspace used by your application. This file is created during startup from the `Application.e4xmi` and after that is used to remember the layout of the window etc. Changes in `Application.e4xmi` are not copied to the `workbench.xmi` file.

12. **Last Step: Enable the workspace resources.** In the current application, you will not be able to see and use workspace resources such as projects and files. To actually use them, we need to initialize the IDE workspace of Eclipse. In the `ApplicationWorkbenchAdvisor.java` (generated by the *Hello RCP* in package `test.rcp`), override the `initialize(...)` method (taken from stackoverflow.com)

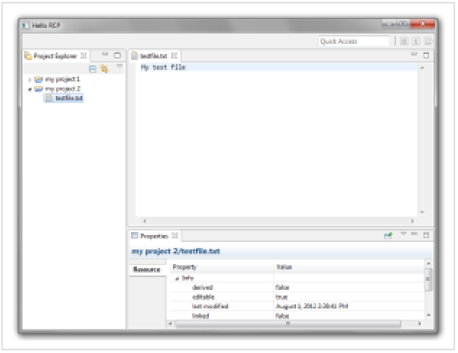
```
...
import org.eclipse.core.runtime.Platform;
import org.eclipse.jface.resource.ImageDescriptor;
import org.eclipse.ui.ide.IDE;
import org.eclipse.ui.internal.ide.IDEWorkbenchPlugin;
import org.osgi.framework.Bundle;
...

@Override
public void initialize(IWorkbenchConfigurer configurer) {
    super.initialize(configurer);
    ...
    // inserted: register workbench adapters
    IDE.registerAdapters();

    // inserted: register images for rendering explorer view
    final String ICONS_PATH = "icons/full/";
    final String PATH_OBJECT = ICONS_PATH + "obj16/";
    Bundle ideBundle = Platform.getBundle(IDEWorkbenchPlugin.IDE_WORKBENCH);
    declareWorkbenchImage(configurer, ideBundle,
        IDE.SharedImages.IMG_OBJ_PROJECT, PATH_OBJECT + "prj_obj.gif", true);
    declareWorkbenchImage(configurer, ideBundle,
        IDE.SharedImages.IMG_OBJ_PROJECT_CLOSED, PATH_OBJECT + "cprj_obj.gif", true);
}

private void declareWorkbenchImage(IWorkbenchConfigurer configurer_p,
        Bundle ideBundle, String symbolicName, String path, boolean shared) {
    URL url = ideBundle.getEntry(path);
    ImageDescriptor desc = ImageDescriptor.createFromURL(url);
    configurer_p.declareImage(symbolicName, desc, shared);
}
```

If you do not have any `ApplicationWorkbenchAdvisor.java`, your Eclipse RCP is probably not built on the Eclipse 3.X framework. Use the *Hello RCP* template to get a basic structure of your application, including this class, or consult an Eclipse 3.X RCP tutorial. Finally, add plugin `org.eclipse.ui.ide` to the dependencies of your RCP plugin project in the `plugin.xml`. The application should now look like this:



My test file

my project 2/testfile.txt

Property	Value
Info	
derived	false
editable	true
last modified	August 1, 2012 2:28:41 PM
linked	false

Done. You can now extend your application through code as well as through the e4 application model and have access to the Eclipse 3.x views, including other views like the Problems View or the Error Log.

Update 2015-08-24: if in the final application the ProjectExplorer content is not visible by its own (but only after you forced an updated, e.g., via opening the context menu with a right-click), try this: override the `getDefaultPageInput()` method of the `ApplicationWorkbenchAdvisor` and add the following line:

```
  return ResourcesPlugin.getWorkspace().getRoot();
```

(thanks Alexander for the hint)

SHARE THIS:

Email Facebook Twitter Print Google

Like

One blogger likes this.

RELATED

gmf knowledge: positioning of external labels
In "GMF"

go with the flow
In "PhD"

activity report
In "Activity Report"

This entry was posted in **Tutorials** and tagged **Compatibility Layer**, **eclipse**, **Eclipse e4**, **Project Explorer**, **RCP**, **Rich Client Platform**, **tutorial** by **dirkfahland**. Bookmark the **permalink [https://dirksmetric.wordpress.com/2012/08/01/tutorial-eclipse-rcp-e4-with-3-x-views-like-project-explorer-properties-etc/]** .

64 THOUGHTS ON "TUTORIAL : ECLIPSE RCP E4 WITH 3.X VIEWS LIKE PROJECT EXPLORER, PROPERTIES, ETC."

Angel Manica
on **August 10, 2012 at 7:12 am** said:

I really do hope that you'll be able to read this soon, I believe your Instructions on step 11 is incomplete. I may have to… what, exactly? Thank you! :)

Dirk
on **August 10, 2012 at 11:41 am** said:

Oh, yes, you're right. "delete" was the missing word. Thank you!

Angel Manica
on **August 10, 2012 at 7:15 am** said:

Never mind :) lol. got it.

Gem Chen
on **November 5, 2012 at 10:20 am** said:

Thanks a lot! It works!!
This feature puzzled me for a long time.

Lars Vogel
on **November 13, 2012 at 7:59 am** said:

Great article, thank you for writing about Eclipse 4. I have a question to your statement about the Eclipse 4 application model editor (As the current editor for the Application.e4xmi has some limitations), which limitation do you see, we can try to fix them.

Dirk
on **November 15, 2012 at 4:53 pm** said:

Hi Lars, thank you. The limitation I refer to is that (as the time of writing of the article), I could not create an “Area” element as child of the “Shared Elements” in a “Trimmed Window”. The only options I get there are “Part”, “PartSashContainer” and “InputPart”. I don’t know whether this is on purpose and there are better ways of solving the problem (i.e., whether my solution currently exploits an undocumented feature of Eclipse 4).

Lars Vogel
on **November 15, 2012 at 7:21 pm** said:

Could you please open a bug for this feature? Link:
https://bugs.eclipse.org/bugs/enter_bug.cgi?product=e4 Component would be “UI”

Dirk
on **November 16, 2012 at 10:06 am** said:

thanks, done: https://bugs.eclipse.org/bugs/show_bug.cgi?id=394446

Lars Vogel
on **November 13, 2012 at 8:04 am** said:

Another comment, to delete the deltas of your application model, you can use the -deletePersistedState flag in your product configuration file. See here:
http://www.vogella.com/articles/EclipseRCP/article.html#tutorial_clearpesistence

Ravi Somayaji (@r_somayaji)
on **December 24, 2012 at 9:14 pm** said:

Super Thanks! The sample Application.e4xmi helped a lot.

Vítor Moscon
on **January 21, 2013 at 2:20 pm** said:

Hi Ravi,
It is a very interesting article!!
After finishing it, I could add the Console, too.
However, I don't know how to use the Project Navigator and Console, or even how to add a simple customized View.
I tried adding simple view as Lars teaches in Eclipse 3 and 4 tutorials, but it is not working over the project I created following your tutorial: "eclipse rcp e4 with 3.x views …"
Please, could you give just more few steps on how to add a simple view or just how to do a simple thing over the Project Navigator you added?

Thanks,
Vítor Moscon

Vítor Moscon
on **January 21, 2013 at 2:57 pm** said:

Correcting: the greeting and compliments are't to Ravi, but to Dirk, who wrote the tutorial!

Hammad
on **March 1, 2014 at 3:54 pm** said:

Whats explained in thsi tutorial works well and all 3.x editors are also working just fine as were expected from compatibility layer. But the problem is, when I add PartStack and Parts according to E4 directly from Application.e4xml file, it doesnt show up. Any thoughts?

Hammad
on **March 1, 2014 at 7:57 pm** said:

solution:… You got to remove workbench.xmi everytime you do the layout change in e4xml file…

```
#cd runtime-x.y.product/.metadata/.plugins/org.eclipse.e4.workbench
#rm -f workbench.xmi
```

and the run your application. New workbench.xmi will be generated and all your changes would be there

teronius
on **April 3, 2013 at 10:19 am** said:

A thousand thanks to you for this. Finally someone not doing copy paste bull… and coming up with a proper solution ;)

Numbers
on **April 5, 2013 at 1:33 pm** said:

Whoa, now that I'd basically given up hope (and had accomodated myself with the full platform, meh), I stumble across this.
Great article.

Frank Benoit
on **April 13, 2013 at 9:37 am** said:

I tried this in my app and encountered problems with the menu bar. It dissappeared after the second start, related to the option -clearPersistedState.
The conclusion on my discussion in the E4 forum and the bug report then was, as I understood it, that mixed mode is not supported.
https://bugs.eclipse.org/bugs/show_bug.cgi?id=398847#c8

kable
on **April 17, 2013 at 7:34 pm** said:

Thanks for the great information. I am able to now get the eclipse3 Properties VIew to show up in my E4 application (well, now e3 w/ e4 model application).

However, I am not able to get the Properties View to see my selected items that implement IPropertySource. So nothing shows up in the Properties View. I suspect this is because the selection model has changed between E3 and E4.

In E3 I would have done something like:
.getSite().setSelectionProvider(viewer);
to get selection events to start firing and the Properties View would have seen these.

In E4 you add yourself to the viewer as a selection change listener and then in the handler you tell the E4 ESelectionService that something has changed selection.

```
viewer.addSelectionChangedListener(new ISelectionChangedListener() {
public void selectionChanged(SelectionChangedEvent event) {
IStructuredSelection selection = (IStructuredSelection) event.getSelection();
// set the selection to the service
selectionService.setSelection(
selection.size() == 1 ? selection.getFirstElement() : selection.toArray());
}
});
```

My guess is that the E3 Properties View doesn't know anything about this ESelectionService and I am not sure if there is something to 'tweak' in the CompatibilityLayer that is used in this demo. Or is there some way to still get the old ViewPart and use the E3 selection service.

Thanks,
kable

Andrzej
on **April 29, 2013 at 12:11 pm** said:

I just want to say that the last 3 – obviously spam – comments (from April 18-23) are absolutely hilarious :).

Also, thanks Dirk for very useful tutorial.

dirkfahland
on **July 30, 2013 at 1:09 pm** said:

Yep, just cleaned up a bit. Glad I could help.

Tuan Anh Nguyen
on **June 26, 2013 at 11:32 am** said:

Thank you very much. I had been searching for 3 days about using 3x views for e4 before found this article. It helps me a lot!!!!

Steve Jones
on **July 23, 2013 at 12:29 pm** said:

Thanks for sharing your experience. I managed to get it to build & run under eclipse 43 (Kepler)

I've uploaded my effort to gitGUB https://github.com/stevej2608/test.rcp

I have a problem with the project explorer needing a r-click before it will display the tree. I'll try and resolve this.

Thanks again, Steve

dirkfahland
on **July 30, 2013 at 1:08 pm** said:

Thank you Steve!

Arthur
on **May 16, 2014 at 3:09 pm** said:

Hi,

does anybody have a solution for this yet?

Pim van Nierop
on **August 14, 2013 at 1:39 pm** said:

I am new to eclipse RCP development and I get stuck in the first section of the walkthrough:

"where test.rcp.application refers to the Eclipse 3.X application defined in the RCP plugin at the org.eclipse.core.runtime.applications extension point, and test.rcp/Application.e4xmi refers to the application model file we just created."

I have created a plugin-project, but I am usure where you are referrign to when you mention the "Eclipse 3.X application". As far as I follow it, we just created an e4 application (via it's model file).

Could anyone explain where my thinking is wrong? Any help is appriciated greatly.

Pim van Nierop
VU University Amsterdam

Pim van Nierop
on **August 14, 2013 at 2:39 pm** said:

To give more insight into what I am doing:

ad init. I created the pugin-project
ad 2. I Created a new e4 Application Model config file
ad 3.1. Added the xml fragment as shown in the plugin.xml
ad 3.2. I created the following extension to org.eclipse.core.runtime.applications in the plugin.xml.

This is not indicated in the text, but I guess this creates a 3.x style application definition. I fact, addign this xml fragment correcly allows me to perform the step 5.
ad 4. Created a product definition.
ad 5. Added the 3.x style application to the product.
ad 6. Added application window.

After fixing the plugin dependencies I get an error that, as far as I could deduce, relates to the 3.x style application:
"!ENTRY org.eclipse.osgi 4 0 2013-08-14 14:29:07.274
!MESSAGE Application error
!STACK 1
org.eclipse.core.runtime.CoreException: Executable extension definition for "run" not found."

So, is the example in this tutorial not self contained? Does it assue I have a running 3.x application somewhere?

Pim van Nierop
on **August 14, 2013 at 2:50 pm** said:

It turned out I needed to add a 'run' item to the 3.x application in the plugin.xml. My application extension now looks like (I hope to get the xml structure displayed, I am unfamiliar with wprdpress markup):

|extension
id="test.rcp.application"
point="org.eclipse.core.runtime.applications"|
|application
cardinality="singleton-global"
thread="main"
visible="true"|
|run
class="test.rcp.Application"|
|/run|
|/application|
|/extension|

I have to create a new Application Java class file that implements IApplication. However, when running the product I get no windows and the console message:

gogo: InterruptedException: sleep interrupted
java.lang.InterruptedException: sleep interrupted
at java.lang.Thread.sleep(Native Method)
at org.apache.felix.gogo.shell.Activator.run(Activator.java:72)
at java.lang.Thread.run(Thread.java:722)

Pim van Nierop

on **August 15, 2013 at 3:27 pm** said:

I have sorted out what was giving me problems. For reference I will state the solution here. Please feel free to clear up my messages if desired.

The cause of my problem is that in the eclipse 4.3 sdk I did not create the initial plugin-project with the option “Would you like to create a 3.x rich client application?” as “Yes”. When you do this the “Hello RCP” template becomes available.

Pim van Nierop
on **August 14, 2013 at 2:40 pm** said:

I am sorry but the xml fragment did not make it into the final port. Don’t know why.

bava502
on **August 18, 2013 at 12:56 pm** said:

Hi Dirk,
Its a great article. Thanks for that.
I have created one e4 application model project which consists of a trimmed window (say window name as “analysis”) with some parts on it. When i launch this project everything is fine as expected.
Now i want to modify this project in such a way that, i need to create one more window(with a part which has a button, say “Open analysis window”) , which should become my first window when i run this project and clicking on this button should launch the trimmed window which i created earlier using e4 application model.

Is it possible to do so?…Please help

Wernke
on **August 23, 2013 at 10:34 am** said:

Great article, which helped me a lot. I am currently trying to get an Xtext generated editor (based on IEditorPart) to display in the placeholder “org.eclipse.ui.editorss”. Is this possible? I am struggling to bring the two worlds (e3 and e4) to speak to each other at this point.

Any hint would be appreciated.

Pim van Nierop
on **August 23, 2013 at 10:44 am** said:

Yes, I am struggeling with sort of the same problem. How can a e4 MInputPart be started on a file/resource that is double clicked in the e3 Project Explorer.

Wernke
on **August 23, 2013 at 3:15 pm** said:

I think your problem is different from mine.

I got my editor to show up by simply opening it from the IWorkbenchPage injected into my OpenEditorHandler:

public class OpenEditorHandler {
@Execute
public void openItem(@Named(IServiceConstants.ACTIVE_SELECTION) MyModelItem item, IWorkbenchPage page) {
IEditorInput input = … ; // something based on the item's nature and location
page.openEditor(input, EDITOR_ID);
}

where EDITOR_ID is the id of your editor contributed to org.eclipse.ui.editors in your xtext.ui plugin.

Pim van Nierop
on **August 23, 2013 at 3:37 pm** said:

Wernke, would you allow me to ask you for additional info about this via email? I am a novice in this and I would like to get working what you suggest here.

Pim van Nierop
on **August 25, 2013 at 11:23 am** said:

How did you couple the "open file" event that is tiggered from double clicking a file in the 3.x Project Explorer to a 4.x Handler class that uses dependency injection? I understand your OpenEditorHandler, but I do not know how to couple it to the Project Explorer. Thnx.

Wernke
on **August 26, 2013 at 9:28 am** said:

Hi Pim, I am not using the project explorer. Your case should be straightforward though. If you register your editor to the extension point org.eclipse.ui.editors it should open on its own when you double click a file with the registered extension. No own handler needed.

Mirko
on **September 27, 2013 at 2:27 pm** said:

Hi,
thank you very much for this tutorial. It saved me a lot of time.
But I still have one anoying problem. Implementing the Project Explorer or Package Explorer (doesn't matter) worked just fine, but a few key bindings are not working one them.
In my case DEL, COPY, PASTE don't work, but bindings like RENAME do work.
Do you maybe have any idea why that is?

Mirko
on **November 5, 2013 at 1:06 pm** said:

I got it working, but just with an ugly workaround. Since the DEL key is working in the editor, i assumed that it is at least working in that context. And because rightclick->DEL in the project explorer is also working, I figured it would be best to built a handler that checks for the different contexts and than calls the appropriate command. So here it is…maybe someone knows a better way. Because this way might not work once you have another part with a different context.

public class DeleteKeyHandler{

```
@Execute
public void execute(@Named(IServiceConstants.ACTIVE_SHELL) Shell shell, EHandlerService
hService, ECommandService cService, EContextService context) {

for (String tmp : context.getActiveContextIds()) {
if (tmp.equals("org.eclipse.ui.textEditorScope")) {
Command command = cService.getCommand("org.eclipse.ui.edit.delete"); //Context:
[org.eclipse.ui.contexts.dialogAndWindow, org.eclipse.ui.contexts.window,
org.eclipse.ui.textEditorScope, org.eclipse.xtext.ui.XtextEditorScope]
if (command.isDefined()) {
ParameterizedCommand cmd = cService.createCommand("org.eclipse.ui.edit.delete", null);
if (hService.canExecute(cmd)) {
hService.executeHandler(cmd);
}
}
return;
}
}
Command command =
cService.getCommand("org.eclipse.ltk.ui.refactoring.commands.deleteResources"); //Context:
[org.eclipse.ui.contexts.dialogAndWindow, org.eclipse.ui.contexts.window]
if (command.isDefined()) {
ParameterizedCommand cmd =
cService.createCommand("org.eclipse.ltk.ui.refactoring.commands.deleteResources", null);
if (hService.canExecute(cmd)) {
hService.executeHandler(cmd);
}
}
}
}
```

Jonghun Park
on **January 27, 2014 at 10:29 am** said:

Thank you for your Help!!!

When I tried this, I faced some critical problem…..

In Application.e4xmi file, I tried to make command and handler, then I make 'HandledMenuItem' in Main Menu…. But when I run my application, that menu does not work. (@Execute method in Handler, I make some logs…but does not printed…..T_T)…

Maybe,….. This is Start with e3 Application LifeCycle… (related to Application, WorkbenchAdvisor…etc…) But Handler (and Injection..) must be start with E4.Application LifeCycle..(right?) So, my Handler does not triggered…I think…

Does any Solution for this problem?

(I am a Korean.. So My English is terrible…Sorry..T_T)

Hammad
on **March 2, 2014 at 5:55 am** said:

As I mentioned in a reply above a problem then a solution – I will list down a few more general problems I faced and their solution for others' eyes in case they also hit same thing as these issue wasted a few hours of my time…

Problem1: Any additional parts you add as per e4 practice in e4xml file – it wont appear when you run.
Solution: You got to remove workbench.xmi everytime you do the layout change in e4xml file…
#cd runtime-x.y.product/.metadata/.plugins/org.eclipse.e4.workbench
#rm -f workbench.xmi
and the run your application. New workbench.xmi will be generated and all your changes would be there.

Problem2: Newly added part class referenced in 'Class URI' field of part; wont invoke code in @postConstruct and @preDestroy dependency injection annotations
Solution: By default E3 plugin initially created is not picking the right version of javax.annotations. Go to plugin.xml and add 'imported packages' in the Dependency Tab; select 'javax.annotations 1.1.0'

Bruno
on **May 26, 2014 at 3:56 pm** said:

Excellent Tutorial!!!
I tried to do something similar, but starting from the mail RCP Template …
I tried to replace the class corresponding to the ProjectExplorer (removed from plugin.xml) with a part descriptors in the corresponding application.e4xmi … and I don't get it to work and get only a "Could not create the view: com.eclipse-tips.rcp.mail.navigationView" …
Any help is appreciated

Bruno
on **May 27, 2014 at 2:42 pm** said:

I did it!!! So … feel free to remove my question … I'll post my sample code soon.
Thanks your blog has been very helpfull

Lorenzo
on **December 8, 2014 at 6:22 am** said:

Can you post the sample code?

Bruno
on **May 28, 2014 at 4:21 pm** said:

I am trying to add an handler to the process explorer (implemented via the Navigationview class of the RCP mail plug-in template) but no success … of course I delete workbench file, but I get no error and nothing show up in console(my handler class does just that and nothing more right now).
In Application.e4xmi I have added category/command/handlers … but nothing shows up.
I tried a generic Part -Explorer -> Handler and a Part-Explorer->Menus->Popup Menu

and here's my handler:

```
package test.rcp;
import org.eclipse.e4.core.di.annotations.Execute;
public class MessagePopupAction {

@Execute
public void execute(){
System.out.println((this.getClass().getSimpleName() + " called"));

};
```

Per Mildner
on **August 18, 2014 at 1:03 pm** said:

What about the standard IDE menus? I followed the example (in Eclipse 4.4) and I get a "Hello RCP" window, much like in the screen shot above. However, I do not get any of the standard menus (File, Help, …). I only get a "My Product" menu.

dirkfahland
on **August 19, 2014 at 9:25 am** said:

Hi Per, you will have to add these on your own by configuring your RCP application. See this tutorial: http://www.vogella.com/tutorials/EclipseCommands/article.html

Pingback: How to add project explorer,Outline and Property Table views in to eclipse rcp application | Computer science, programming and Eclipse RCP

Ángel Mora
on **January 20, 2015 at 1:05 pm** said:

Got it!, thank you very much. You saved my morning of work :)

cmalai
on **May 8, 2015 at 8:14 am** said:

Great Article. Definitely need of the hr for many ! Thanks Dirk

Anthony Maldonaldo
on **July 7, 2015 at 10:28 am** said:

Dirk,. Thank you kindly for this post.

I'm very new to Eclipse RCP and maybe I don't understand all of this enough. I am working with Eclipse 4.4 and e4 Application Model. Is there a way to do this strictly in e4 without using 3.x compatibility layer on any level?. Specifically I am looking to populate the table in the properties page with data from the selected object which is selected in the Project Manager. Any link to helpful tutorials on how to experiment with this would be most appreciated. Is there a simple way to do this, or will I simply have to run the getter methods on the selected object and then populate the table from that data?
– Anthony

dirkfahland
on **July 9, 2015 at 9:48 am** said:

Dear Anthony,

as far as I know, e4 does not provide a properties component that you could fill. You would have to rebuild that yourself in e4. If you want to use the existing properties component, then you have to see the e3 layer.

Dirk

Marianne Stecklina
on **September 29, 2015 at 6:01 pm** said:

Dirk,
thank you for the tutorial. I followed your instructions and now I'm wondering about the following issue:
When running the application I always get the errors
"Product iec61850ETCompat.product could not be found." and
"java.lang.RuntimeException: No application id has been found." if I don't check my own project in the run configurations Plug-in tab each time I want to run it.
Furthermore another product extension point is added to my plugin.xml though I already defined the one in step 3.
Could you point me out what I am doing wrong?

kirankumar
on **December 14, 2015 at 11:44 am** said:

Dirk,
We have our product well in shape with rcp 3.x and development IDE is eclipse indigo. My product is having many legacy views like Project Explorer and console.
Now we want to migrate it to rcp 4 and IDE will Mars.1

Do you see any issue to migrate to rcp4 ? Please suggest if there is any potential problems ?

Can you give suggestions for migration.

dirkfahland
on **December 14, 2015 at 2:56 pm** said:

I don't have much experience with migrating RCP 3.x to 4, just for one product based on which I wrote this tutorial. I think everything is there, i.e., you won't loose functionality, but finding the right hooks in RCP 4 to get them all in can be quite time consuming as you have to synchronize the way the legacy views get on the UI (via API code) with the e4 way (models).

kiran kumar
on **December 21, 2015 at 7:31 am** said:

Hello Dirk,
Thanks for your reply.

Yes migration and getting 3.x legacy components to RCP4 is quite time consuming process but somehow i managed and migrated my legacy 3.x components successfully.

Now i am facing issue with CSS.

CSS is working over RCP4 also with my existing 3.x CSS setup but few specific things are not working after migration.

CSS is not applied to CoolBar,Banner on RCP4 and extra space is added prior to ToolBar contribution.

Can i get any help ??

Snippet of CSS :-
CBanner,CoolBar,CTabFolder,ToolBar{

background-color: rgb(82,84,81);
}

CoolBar and Banner is not working on RCP4.

kiran kumar
on **December 22, 2015 at 1:47 pm** said:

Hello,
Above issue is fixed with using proper class names and identifiers respectively but extra spaces are added prior to ToolBar is not fixed.

Any Help?

Thanks,
Kiran.

dirkfahland
on **December 22, 2015 at 5:36 pm** said:

Hi Kiran, good to hear that you solved the CSS issue as I am not familiar with that part of e4. As for the extra spaces added prior to ToolBar: I experienced that myself as well and I think (but am not sure) that it is because both 3.X and e4 both contribute toolbars in their own way. The empty space most likely originates in an empty 3.X toolbar provided by the legacy layer/some view. That's all I recall from the time of writing the tutorial. I hope this helps you a bit further.

Dirk

Abhi
on **January 19, 2016 at 7:01 pm** said:

Hello Dirk,

Fantastic article! Thanks a lot!

A basic question. It is now 2016. If I want to write an e4 application, and I want to use the common navigator framework (CNF) and use extensions for using the team functionality (e.g. SVN/Git integration), do I still need to use the compatibility layer? Is there no "native" e4 support? If yes, any hints to any links or APIs? I have looked around, but I can see of no mention of being able to use CNF e.g. in an e4 application.

Thanks in advance!
Kind regards,
Abhi

dirkfahland
on **January 20, 2016 at 5:18 pm** said:

Dear Abhi,

thanks! There is a "bug" filed on eclipse.org under http://bugs.eclipse.org/bugs/show_bug.cgi?id=403745 that captures the request for more complete support of the legacy components in e4. Not much has happened since, but the bug has only few votes so far. I guess promoting the bug (voting it up) might move it more to the center of attention. (I'm not actively involved in eclipse development.)

Dirk

Anael
on **January 20, 2016 at 11:28 am** said:

Hello Dirk,
I followed your tutorial with my e3 application and I have an exception at the launch:
ERROR ui.ApplicationWorkbenchAdvisor – Unable to process “PartServiceImpl.engine”: no actual value was found for the argument “IPresentationEngine”.
There is another way to use e3 application with e4 ? I didn t find another concrete tutorial to migrate e3 to e4.

dirkfahland
on **January 20, 2016 at 5:20 pm** said:

Hi Anael, I haven’t seen that problem before. My closest guess would be some missed or wrong configuration somewhere. Try doing the tutorial again down to the last comma and point. Maybe that fixes it. Sorry, Dirk

Martin
on **August 12, 2016 at 6:29 pm** said:

Hi Dirk,
thanks for this very helpful tutorial.
I am using NEON (4.6) to implement your tutorial. All wokrs fine except the icons of the project explorer, I don’t mind to have no menu for the moment. The project icon is missing at all, its just a small red square. The folder and text file icons are fine but the text its too close to the icons.
I think the icons should be in org.eclipse.ui.ide and this plugin is included but it dies not help at all.

Do you have any idea how to have propper icons displayed?

nakag
on **August 20, 2016 at 7:22 am** said:

Hi Martin,
I had same problems.
Replace *.gif to *.png, It works fine for me.

Since Neon, gif images seem to be removed from org.eclipse.ui.ide .

thanks.

Martin
on **August 22, 2016 at 11:23 am** said:

Thanks nakag,
the *.png Images do the trick ;).

Martin