openInstaller IDE User Guide

v1.0

Introduction

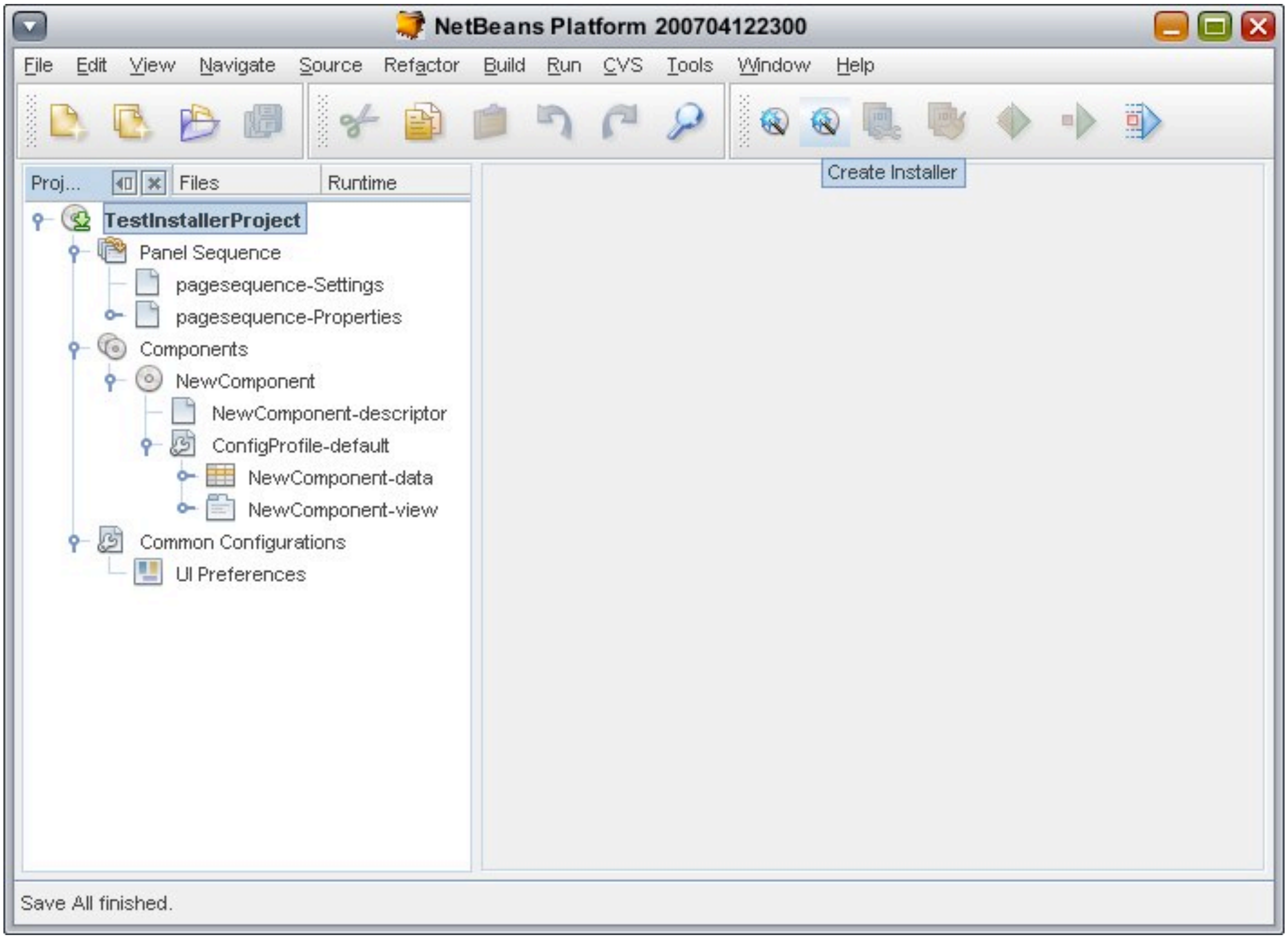
This document explains how to use openInstaller Developer Tools (aka IDE) to create Installers based on openInstaller Installer Framework

Getting the IDE

openInstaller Developer Tools can be built locally after checking out the sources from openInstaller.dev.java.net. The top source folder is source/trunk/DeveloperTools/InstallerModuleSuite. To build, use NetBeans 5.5.x or NetBeans 6.0 IDE running on top of J2SE 5.0. The module code is independent of any NetBeans 6.0 specifics so it works without any problems on NetBeans 5.5.x.

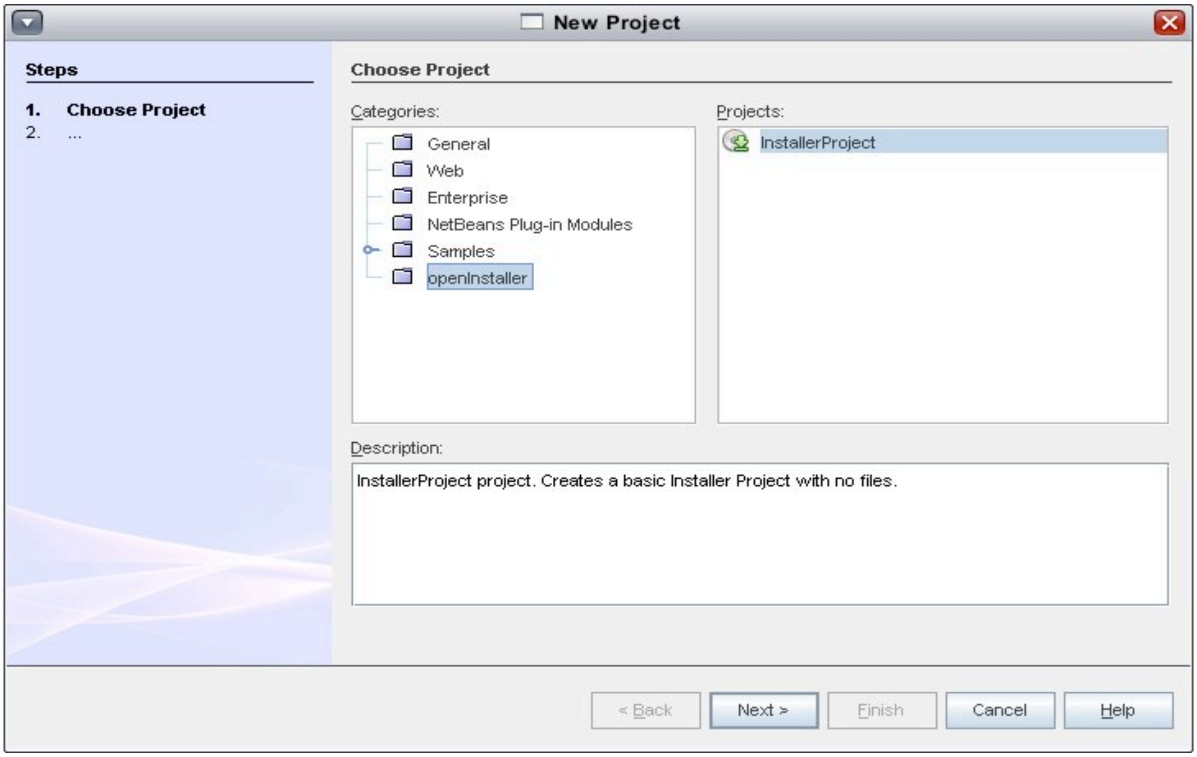
Running the IDE

Once you have the build, hit F6 in the NetBeans IDE and this will bring up the openInstaller Developer Tools installed in another session of NetBeans. You can also generate NBMs and host it on an NetBeans Update center for easy access. The following screenshot shows the openInstaller IDE main window.



Creating New Installer using openInstaller IDE

Creating a new installer starts with creating a new installer project in NetBeans. Select New Project from file menu and choose the Installer Project from the New Project wizard as shown below.



New Project

Steps

1. Choose Project
2. ...

Choose Project

Categories:

- General
- Web
- Enterprise
- NetBeans Plug-in Modules
- Samples
- openInstaller

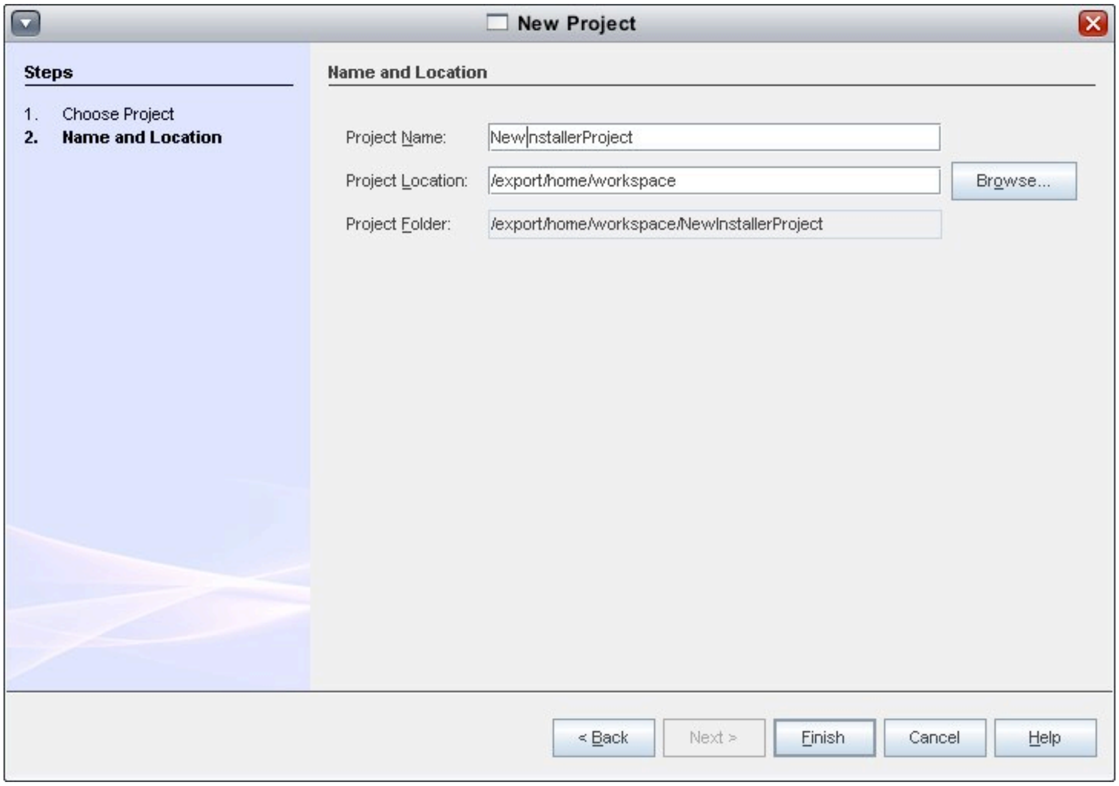
Projects:

- InstallerProject

Description:

InstallerProject project. Creates a basic Installer Project with no files.

The next screen asks you the location of the Installer project to be created on the disk.



New Project

Steps

1. Choose Project
2. Name and Location

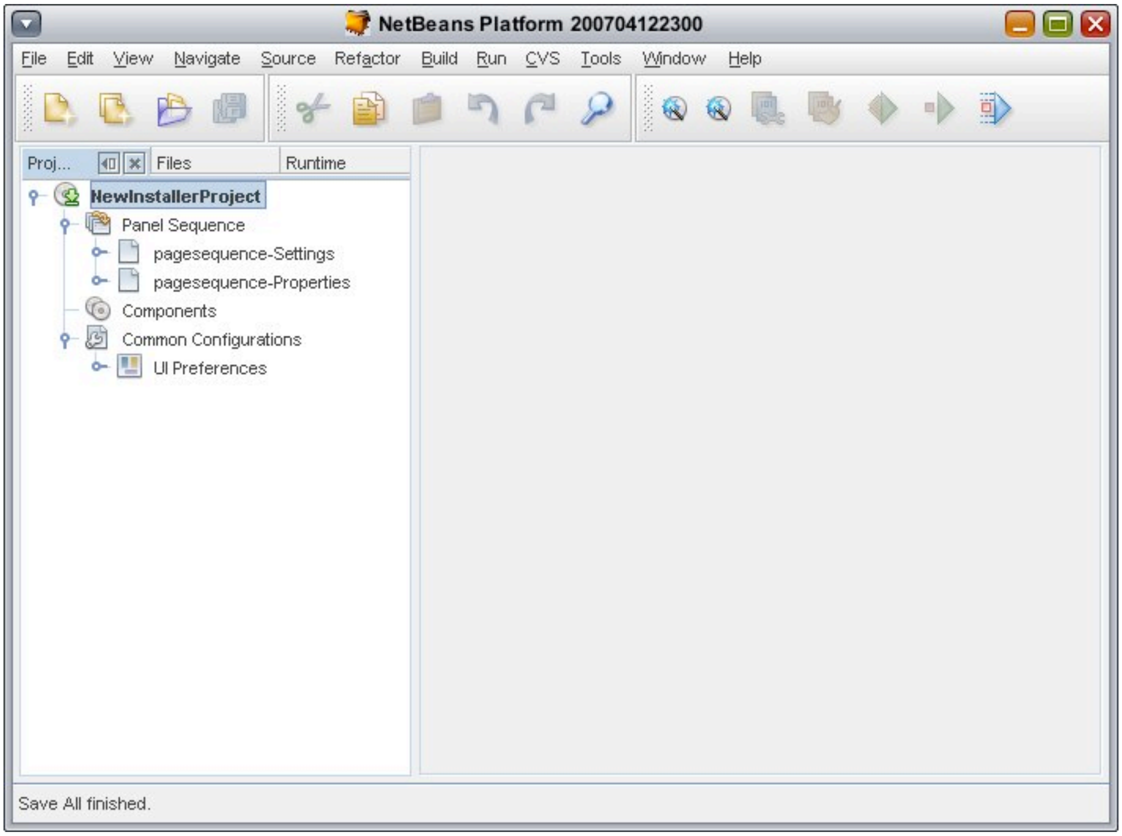
Name and Location

Project Name: NewInstallerProject

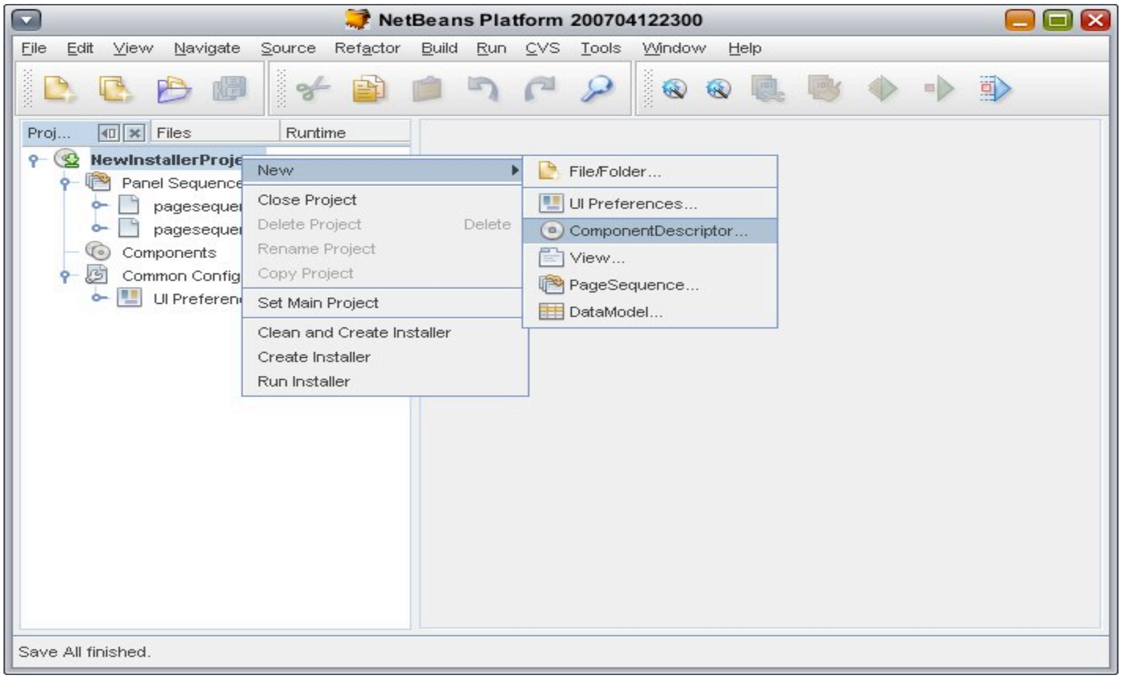
Project Location: /export/home/workspace

Project Folder: /export/home/workspace/NewInstallerProject

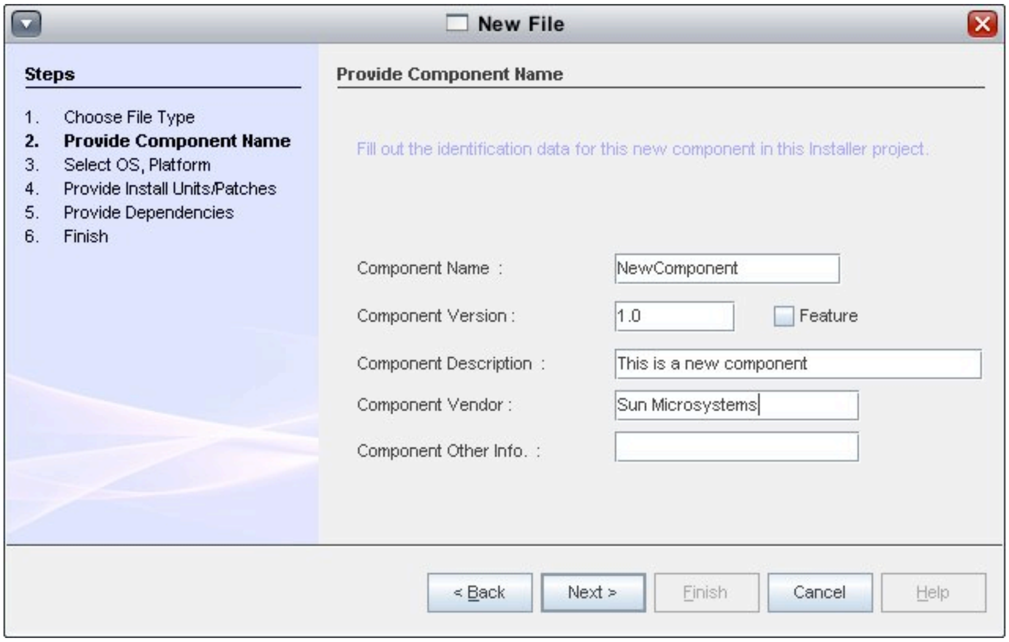
When you provide the location and click 'Finish' a new installer project is created and it will be opened in the project window in the NetBeans IDE as shown below.



The newly created Installer project will be empty and will not have any components/products in it. To create a new component or to add a new product into the installer select 'File/New' or right click on the project node and select 'New/ComponentDescriptor' from the popup.



This will bring up a wizard with will ask the component name, its version, the Operating System and platform support that you would like to include and finally packages for each platform/OS combinations. This wizard optionally also allows to enter the dependencies information if you know it at hand. The following screen shots shows the New component wizard in action.



New File

Steps

1. Choose File Type
2. **Provide Component Name**
3. Select OS, Platform
4. Provide Install Units/Patches
5. Provide Dependencies
6. Finish

Provide Component Name

Fill out the identification data for this new component in this Installer project.

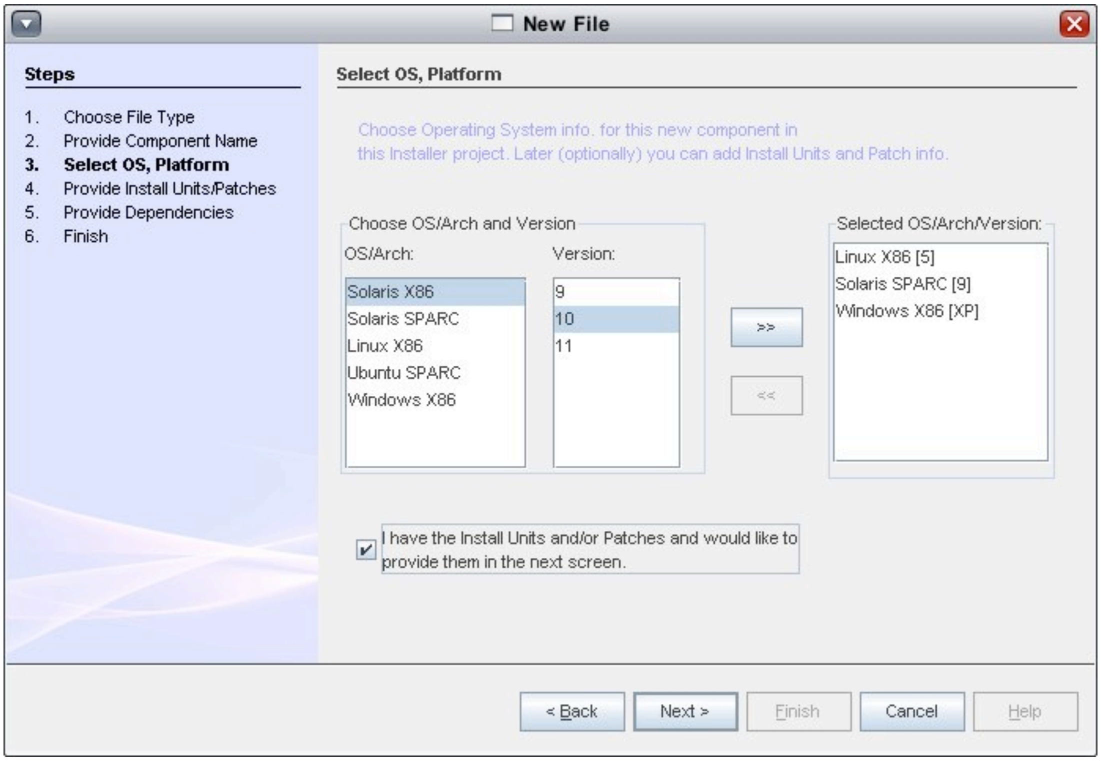
Component Name : NewComponent

Component Version : 1.0 □ Feature

Component Description : This is a new component

Component Vendor : Sun Microsystems

Component Other Info. :



New File

Steps

1. Choose File Type
2. Provide Component Name
3. **Select OS, Platform**
4. Provide Install Units/Patches
5. Provide Dependencies
6. Finish

Select OS, Platform

Choose Operating System info. for this new component in this Installer project. Later (optionally) you can add Install Units and Patch info.

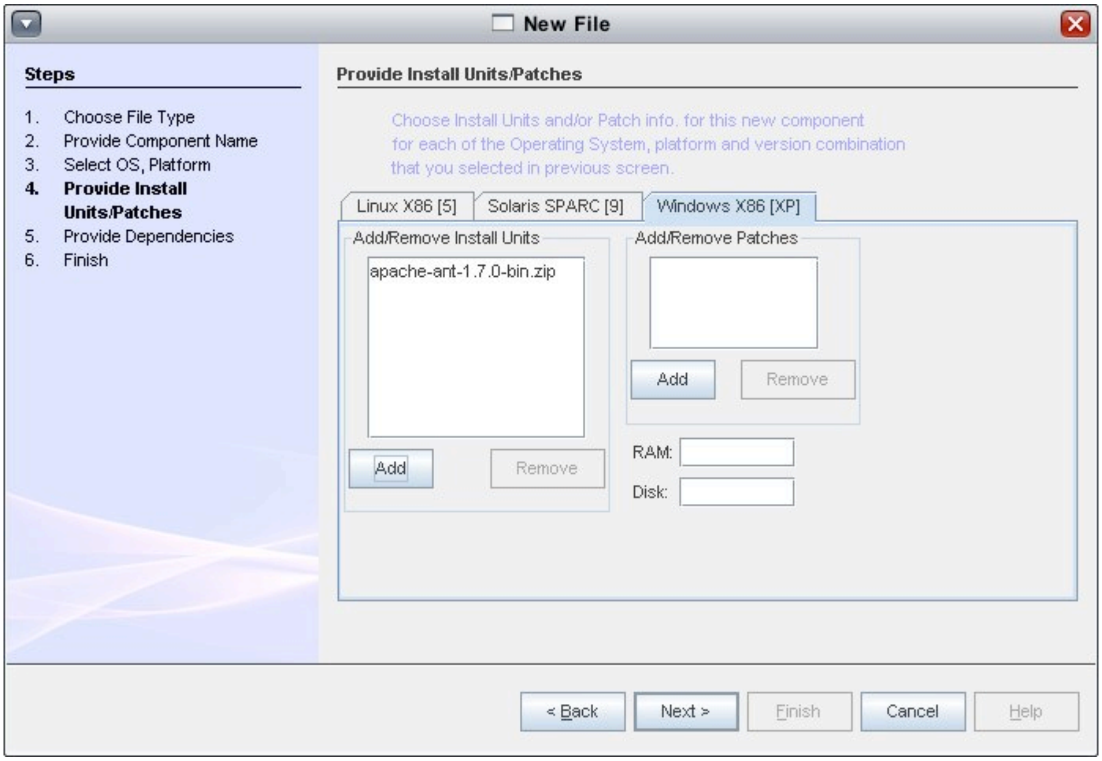
Choose OS/Arch and Version

OS/Arch: Solaris X86, Solaris SPARC, Linux X86, Ubuntu SPARC, Windows X86

Version: 9, 10, 11

Selected OS/Arch/Version: Linux X86 [5], Solaris SPARC [9], Windows X86 [XP]

☑ I have the Install Units and/or Patches and would like to provide them in the next screen.



New File

Steps

1. Choose File Type
2. Provide Component Name
3. Select OS, Platform
4. **Provide Install Units/Patches**
5. Provide Dependencies
6. Finish

Provide Install Units/Patches

Choose Install Units and/or Patch info. for this new component for each of the Operating System, platform and version combination that you selected in previous screen.

Linux X86 [5] | Solaris SPARC [9] | Windows X86 [XP]

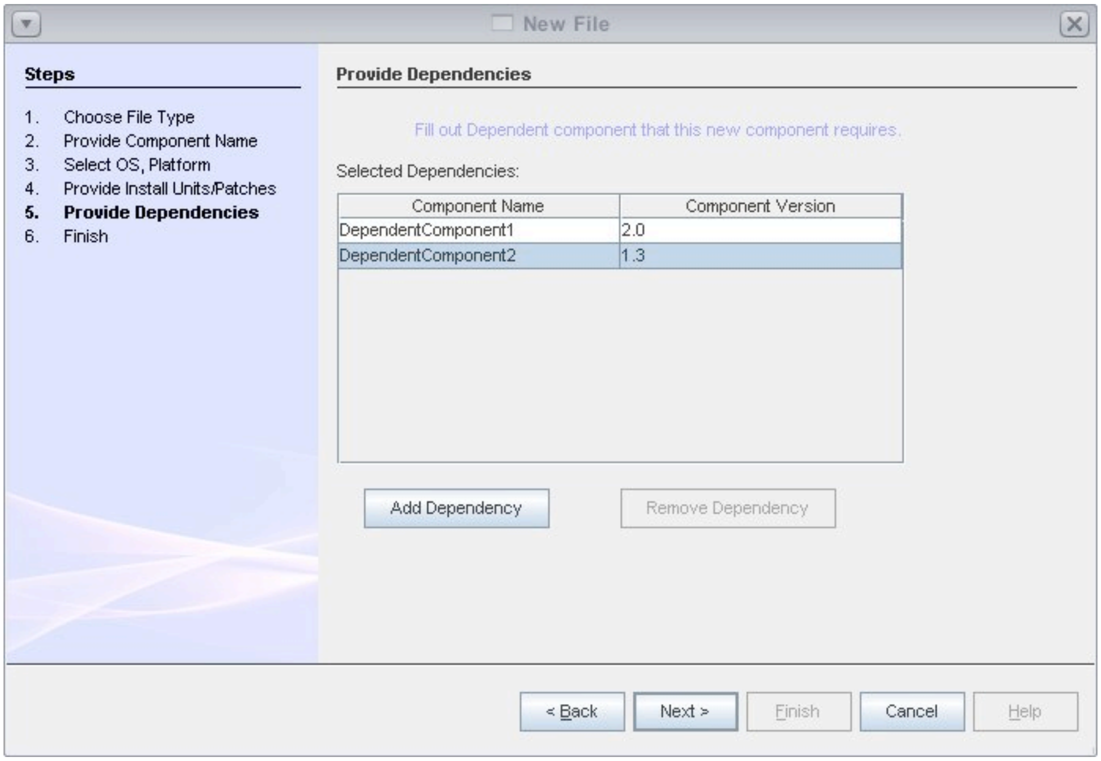
Add/Remove Install Units

apache-ant-1.7.0-bin.zip

Add/Remove Patches

RAM:

Disk:



New File

Steps

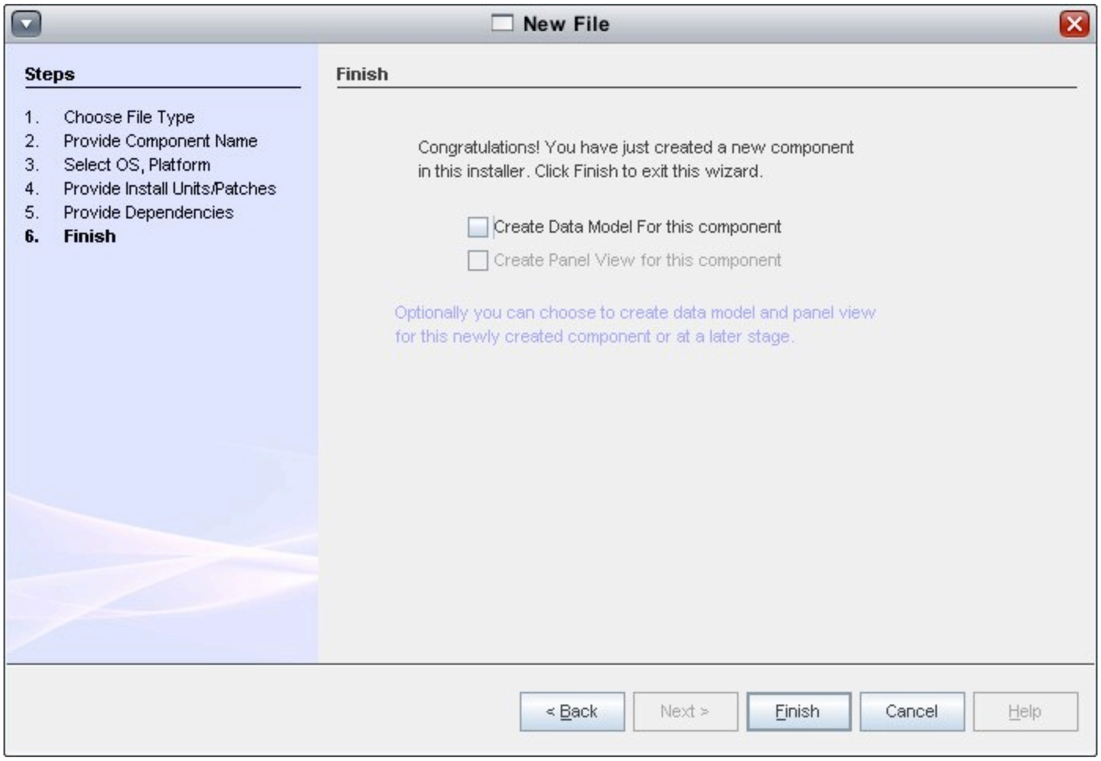
1. Choose File Type
2. Provide Component Name
3. Select OS, Platform
4. Provide Install Units/Patches
5. **Provide Dependencies**
6. Finish

Provide Dependencies

Fill out Dependent component that this new component requires.

Selected Dependencies:

Component Name	Component Version
DependentComponent1	2.0
DependentComponent2	1.3



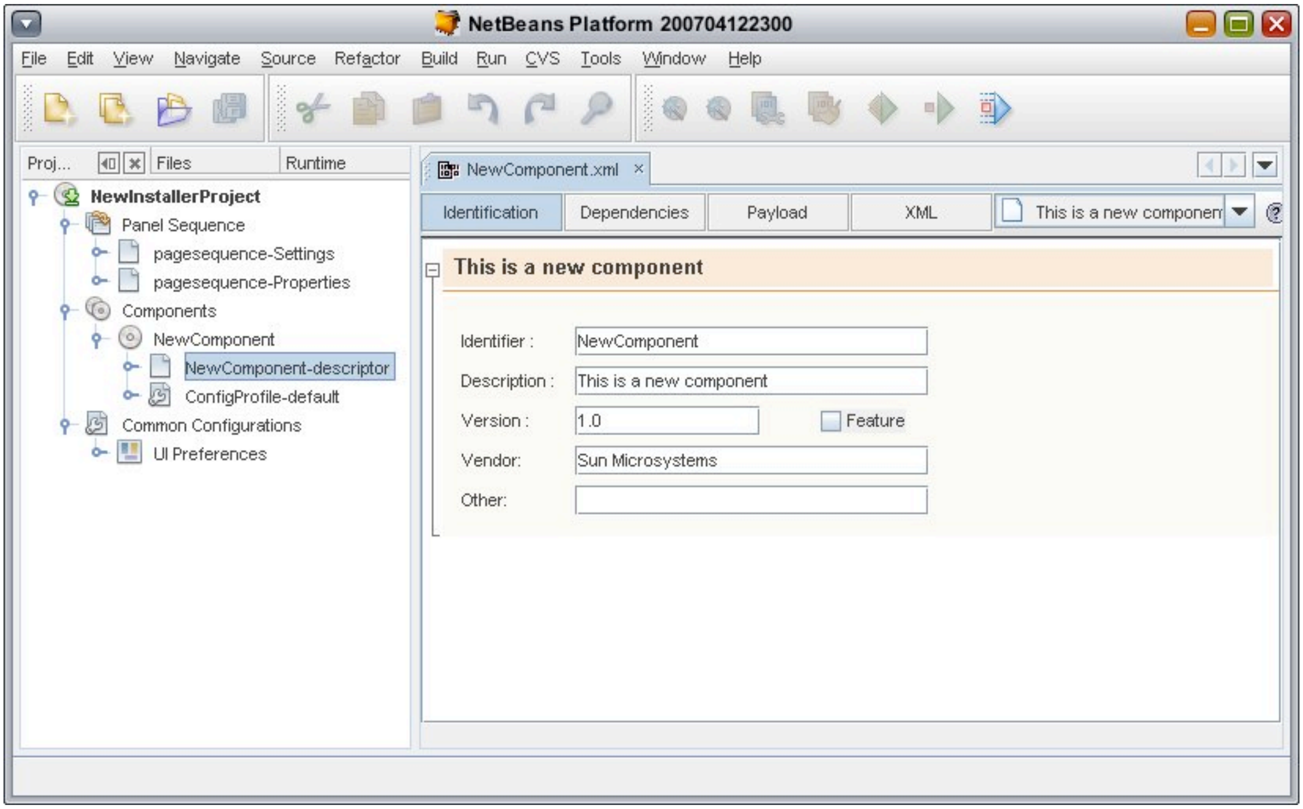
Congratulations! You have just created a new component in this installer. Click Finish to exit this wizard.

□ Create Data Model For this component

□ Create Panel View for this component

Optionally you can choose to create data model and panel view for this newly created component or at a later stage.

Once a new component is created, it will be opened for editing as shown below. The visual editor is implemented using XML multiview API of NetBeans.



This is a new component

Identifier : NewComponent

Description : This is a new component

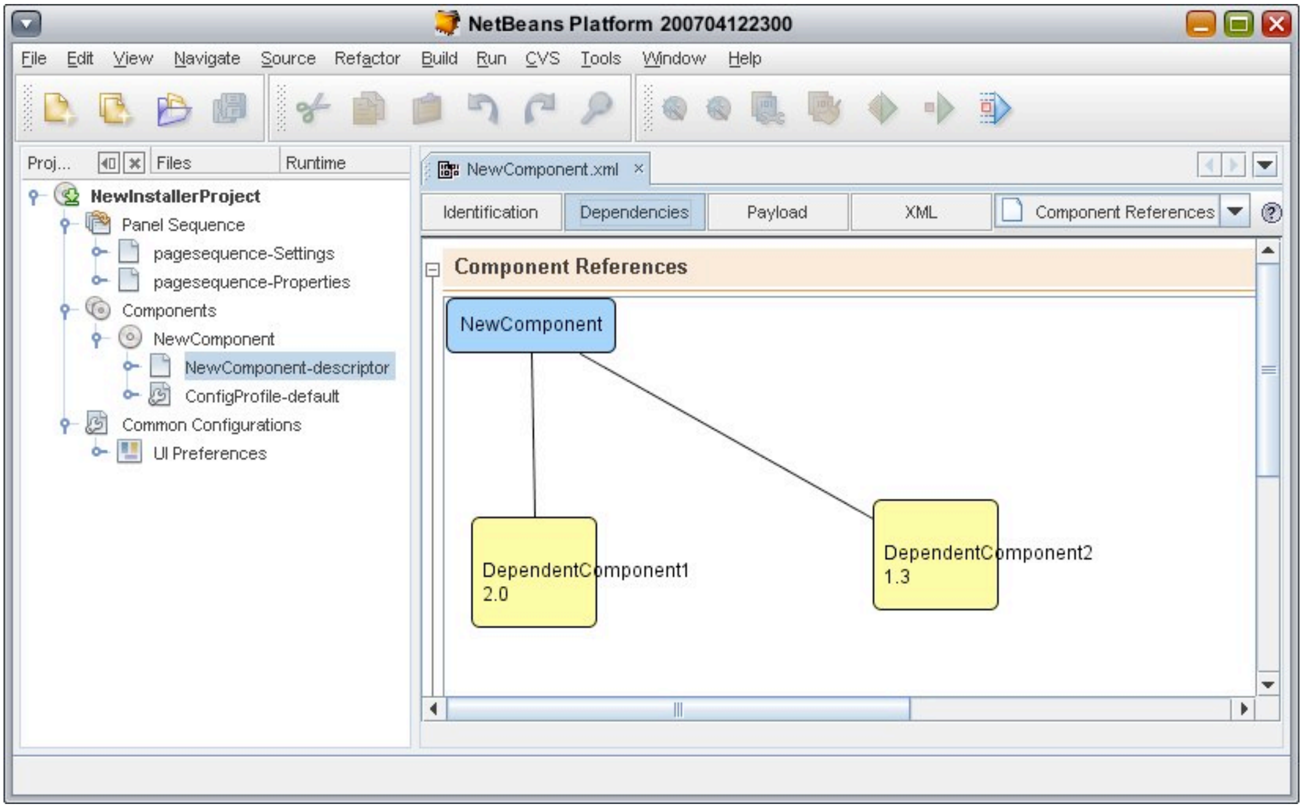
Version : 1.0 □ Feature

Vendor: Sun Microsystems

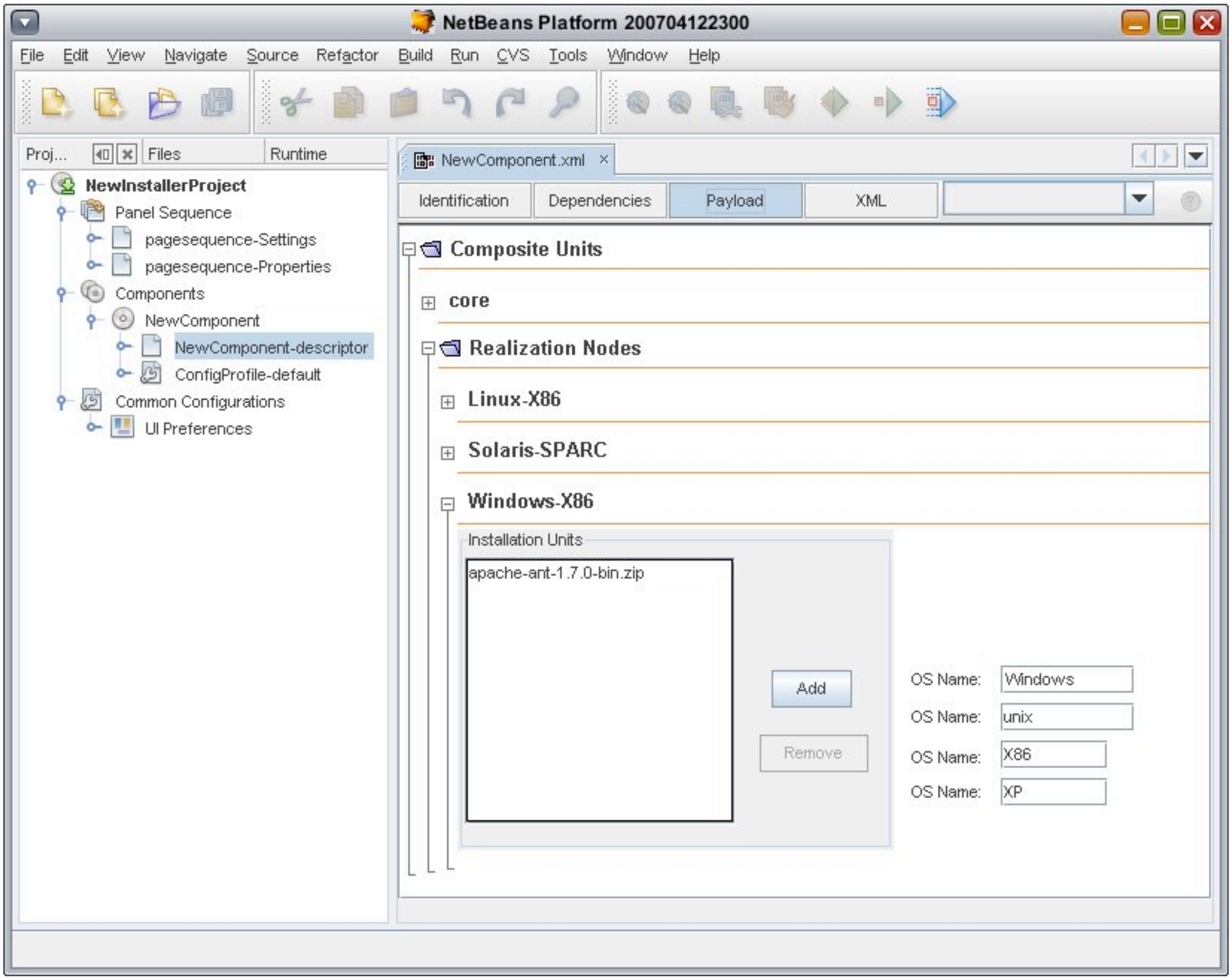
Other:

The first tab shows the component basic details like name and version. What ever the fields gets edited

here will get automatically synched to the XML file.
The following screen shot shows the dependency view, where in the graphical editor shows the dependent components for the current component in this installer project. You can also add new dependencies or delete existing dependencies from this view.



The following screen shot shows the Payload view wherein, for each OS/Platform combinations of this current component, there exists an editor section which allows to view and edit individual packages for this component for that particular OS/Platform combination.



apache-ant-1.7.0-bin.zip

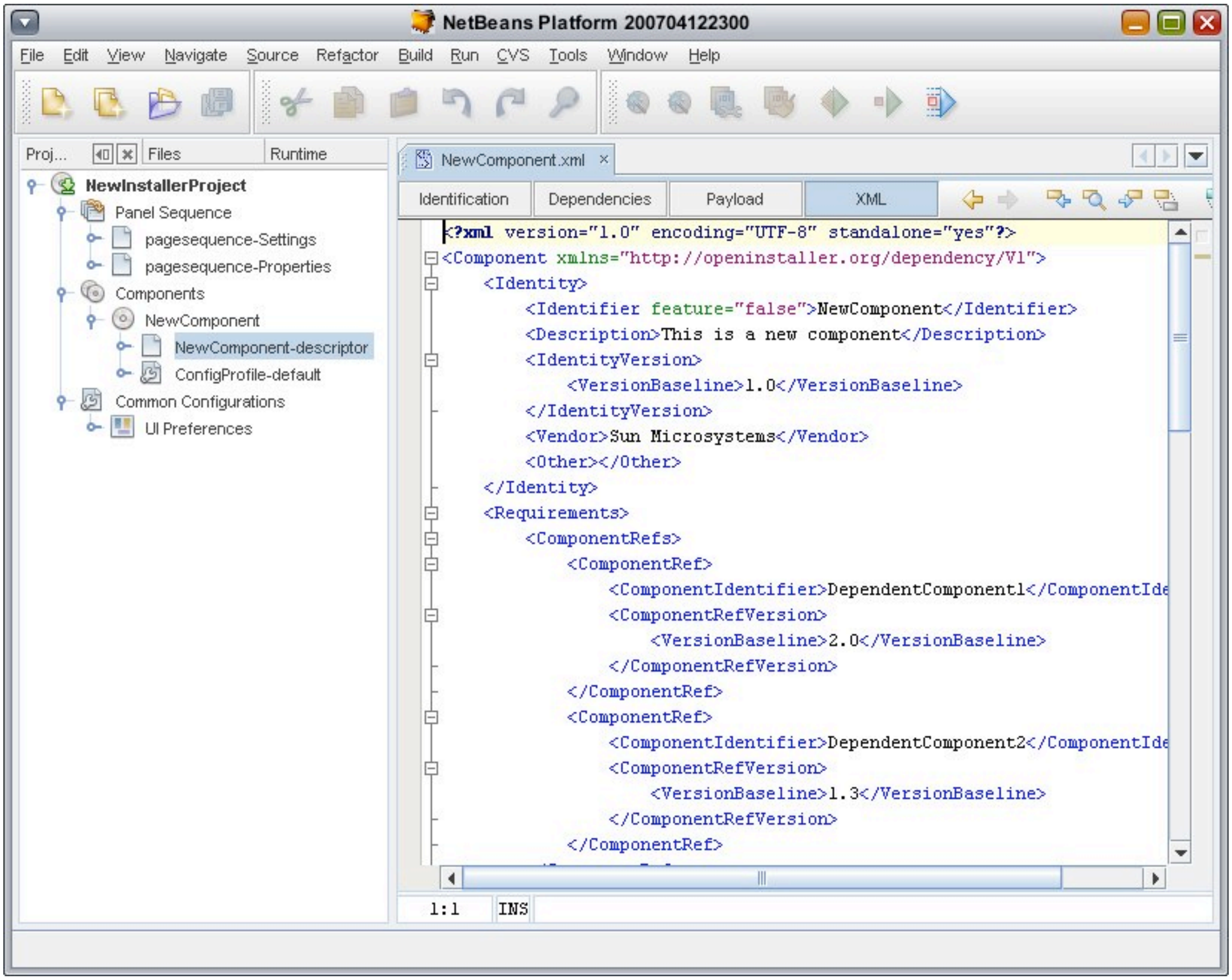
OS Name: Windows

OS Name: unix

OS Name: X86

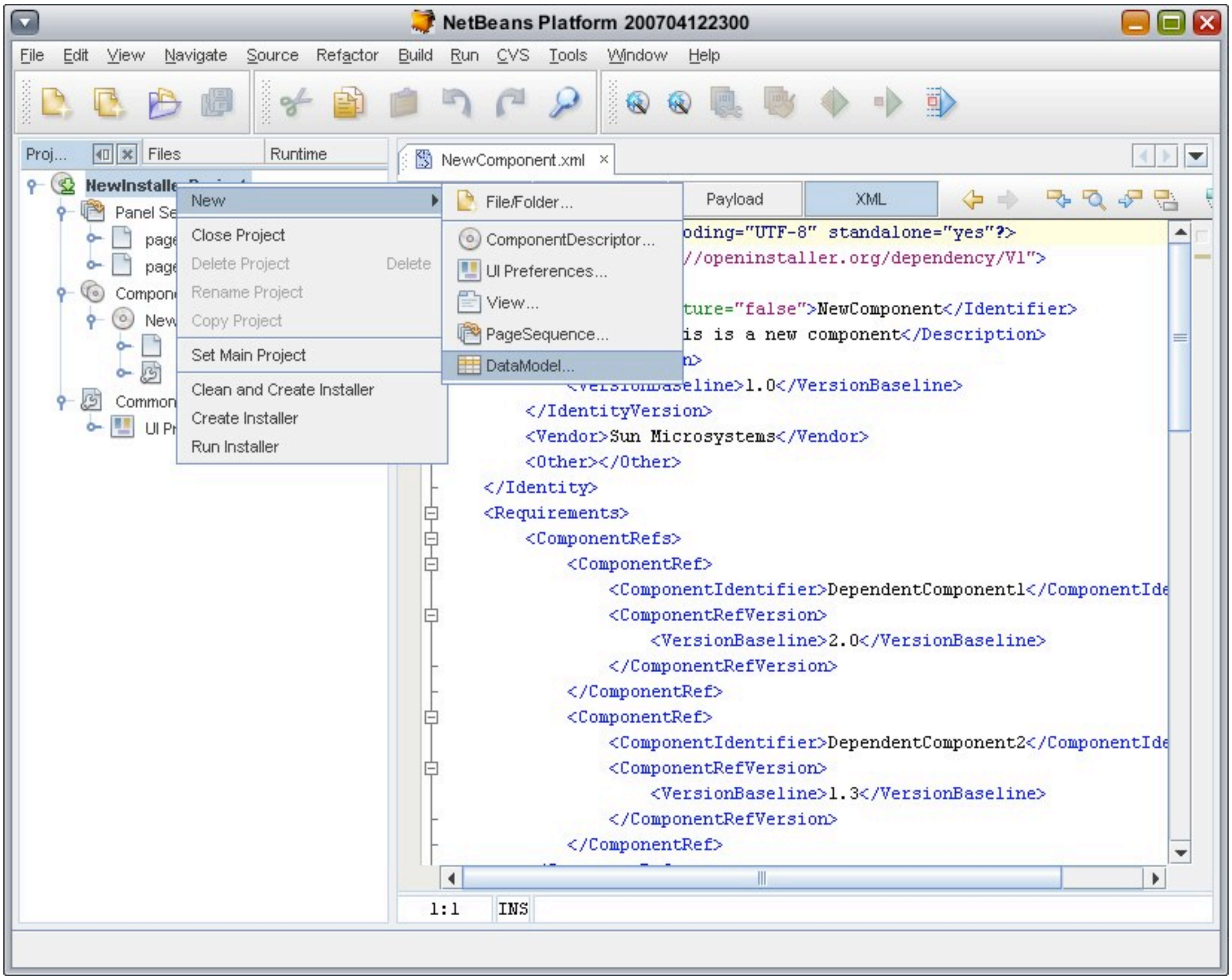
OS Name: XP

The following screen shot shows the Raw XML file view of the component. Here one can edit/update the generated XML manually and it will be synched with the rest of the visual editor.



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Component xmlns="http://openinstaller.org/dependency/V1">
    <Identity>
        <Identifier feature="false">NewComponent</Identifier>
        <Description>This is a new component</Description>
        <IdentityVersion>
            <VersionBaseline>1.0</VersionBaseline>
        </IdentityVersion>
        <Vendor>Sun Microsystems</Vendor>
        <Other></Other>
    </Identity>
    <Requirements>
        <ComponentRefs>
            <ComponentRef>
                <ComponentIdentifier>DependentComponent1</ComponentIde
                <ComponentRefVersion>
                    <VersionBaseline>2.0</VersionBaseline>
                </ComponentRefVersion>
            </ComponentRef>
            <ComponentRef>
                <ComponentIdentifier>DependentComponent2</ComponentIde
                <ComponentRefVersion>
                    <VersionBaseline>1.3</VersionBaseline>
                </ComponentRefVersion>
            </ComponentRef>
```

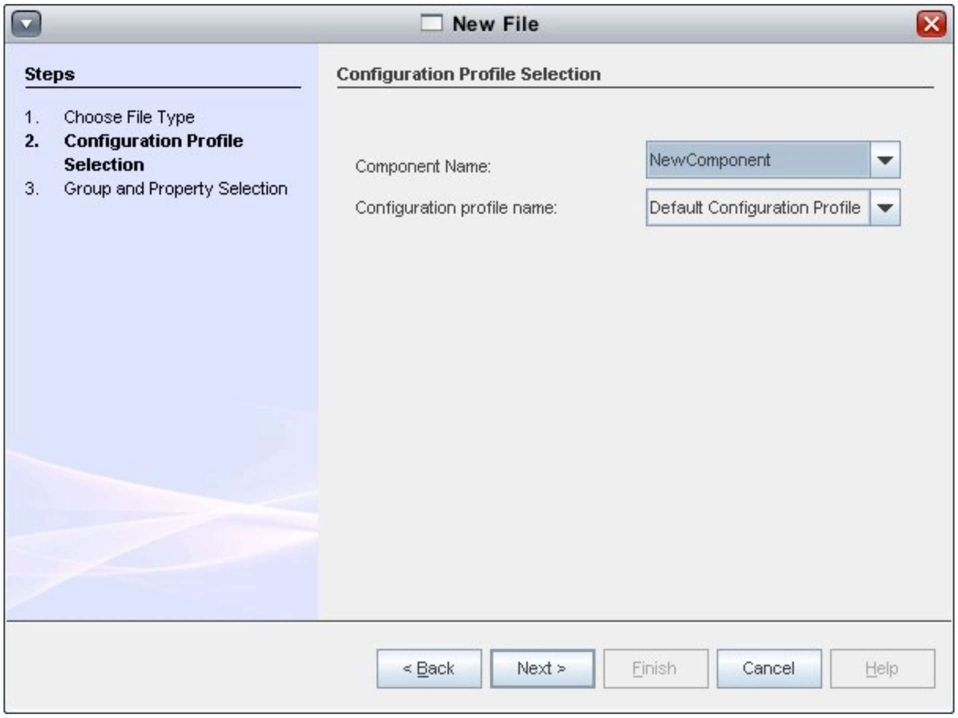
We will continue the process of creating new installer by creating a new Data Model. This data model in openInstaller, will hold the configuration parameters thats are required for the product during installation configuration time. Right click on the project node and select 'New/Data Model' to start.



```
                        oding="UTF-8" standalone="yes"?>
                        //openinstaller.org/dependency/V1">

                        ture="false">NewComponent</Identifier>
                        is is a new component</Description>
                        n>
            <VersionBaseline>1.0</VersionBaseline>
        </IdentityVersion>
        <Vendor>Sun Microsystems</Vendor>
        <Other></Other>
    </Identity>
    <Requirements>
        <ComponentRefs>
            <ComponentRef>
                <ComponentIdentifier>DependentComponent1</ComponentIde
                <ComponentRefVersion>
                    <VersionBaseline>2.0</VersionBaseline>
                </ComponentRefVersion>
            </ComponentRef>
            <ComponentRef>
                <ComponentIdentifier>DependentComponent2</ComponentIde
                <ComponentRefVersion>
                    <VersionBaseline>1.3</VersionBaseline>
                </ComponentRefVersion>
            </ComponentRef>
```

The wizard below shows the first panel. In this, the wizard asks to select a config profile which this new data model should belong to and allows to select the component for which it defines the config. Store.



New File

Steps

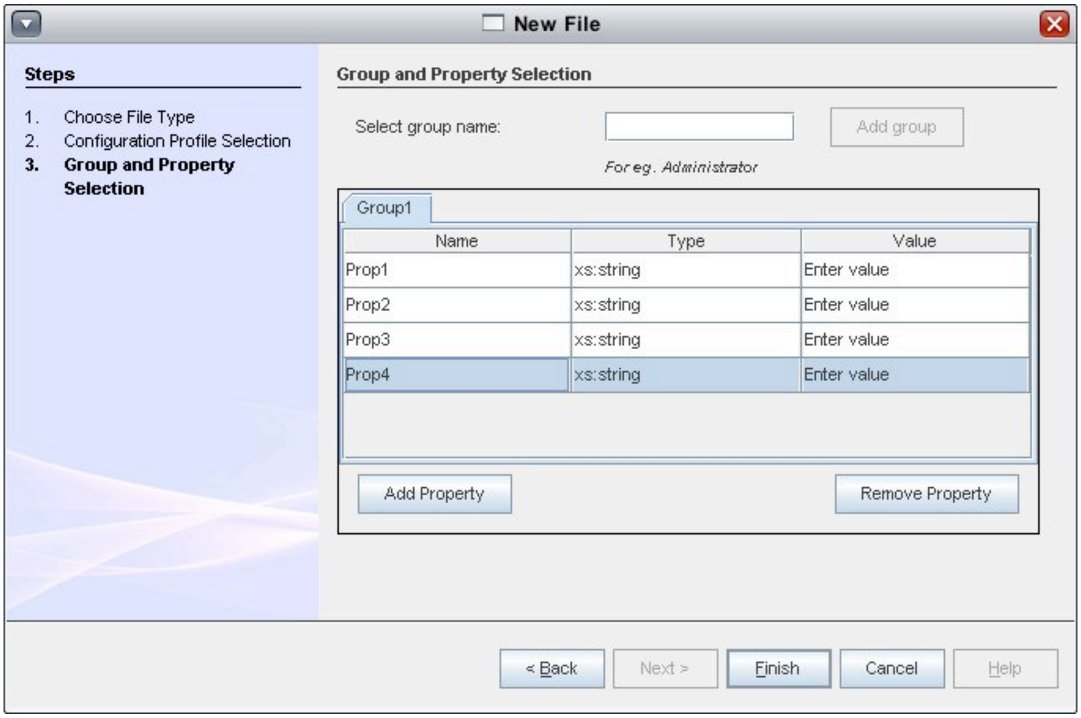
1. Choose File Type
2. **Configuration Profile Selection**
3. Group and Property Selection

Configuration Profile Selection

Component Name: NewComponent

Configuration profile name: Default Configuration Profile

On the next screen (following screen shot) you can add and update the groups and properties in this config store.



New File

Steps

1. Choose File Type
2. Configuration Profile Selection
3. **Group and Property Selection**

Group and Property Selection

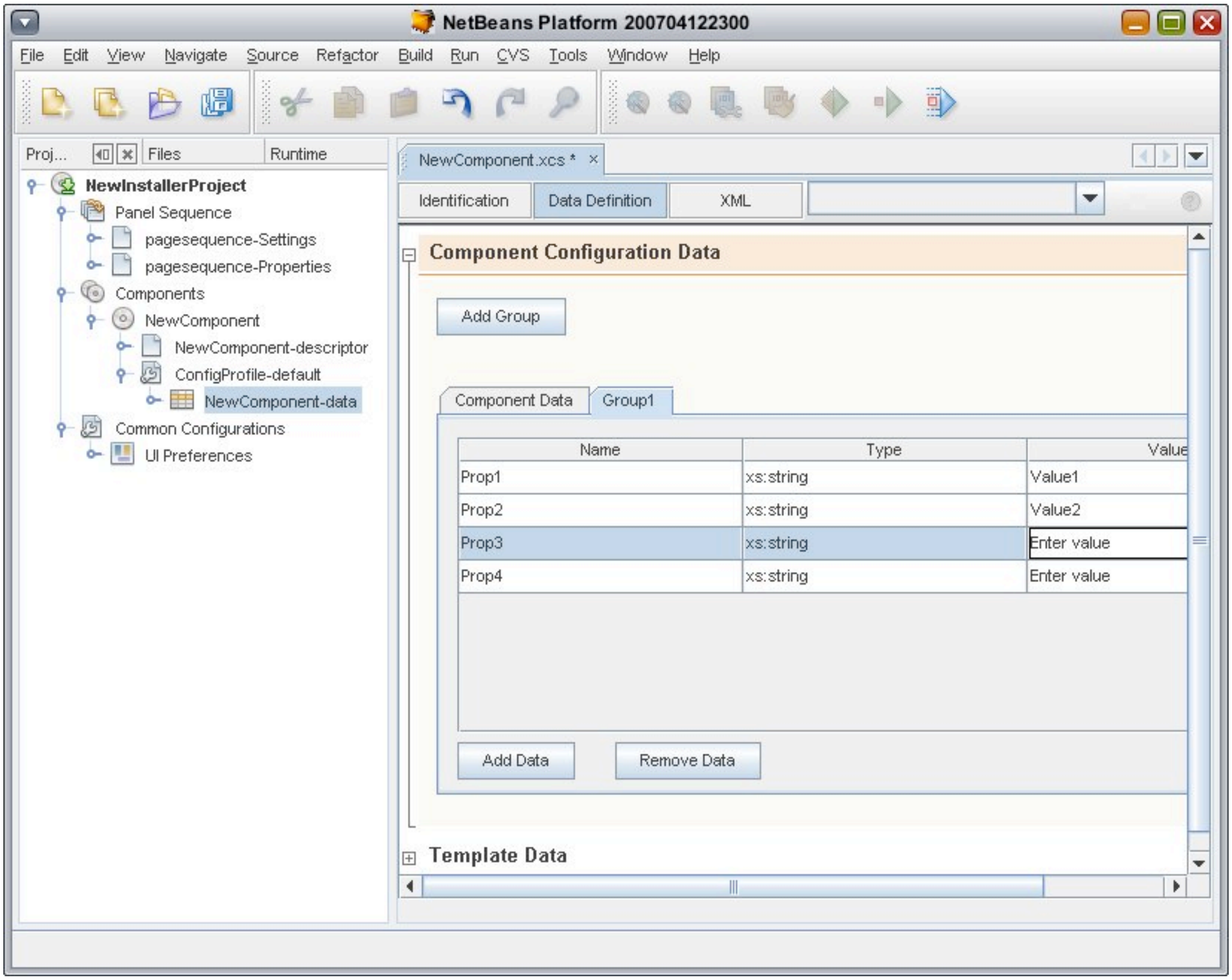
Select group name:

For eg. Administrator

Group1

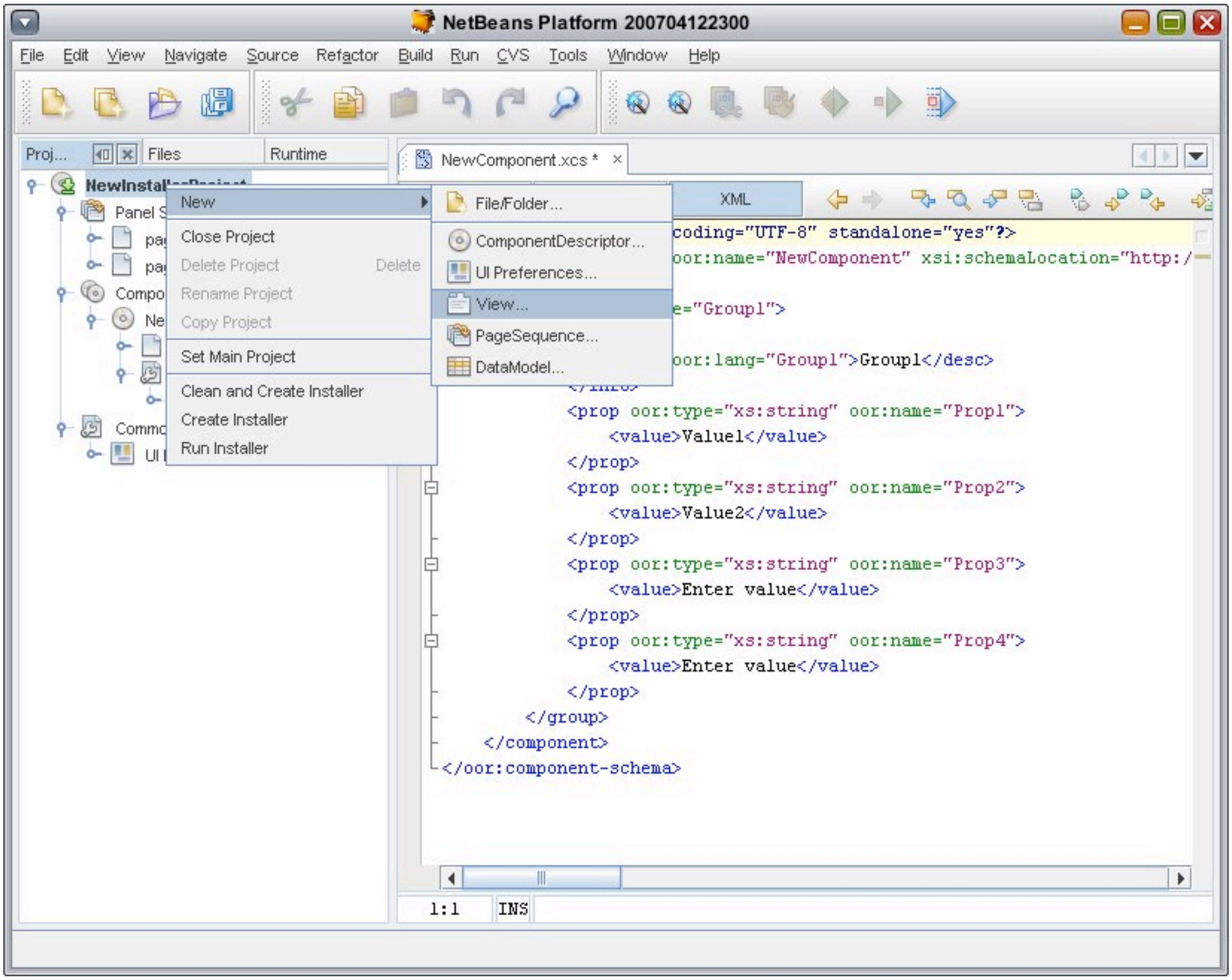
Name	Type	Value
Prop1	xs:string	Enter value
Prop2	xs:string	Enter value
Prop3	xs:string	Enter value
Prop4	xs:string	Enter value

The screen shot below shows the newly created data model in the editor.



Name	Type	Value
Prop1	xs:string	Value1
Prop2	xs:string	Value2
Prop3	xs:string	Enter value
Prop4	xs:string	Enter value

Next we will create a new view and use this newly created data model as the backing store. Again right click on the project node and select 'New/View'. This will open the New View panel wizard.

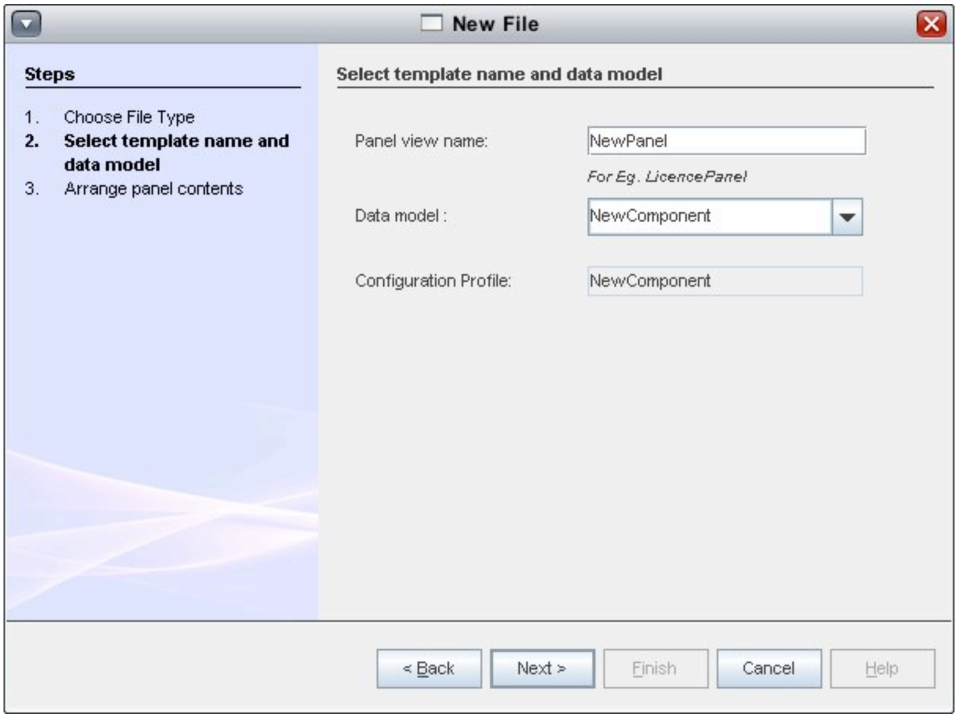


```
coding="UTF-8" standalone="yes"?>
oor:name="NewComponent" xsi:schemaLocation="http:/

e="Group1">

oor:lang="Group1">Group1</desc>
        </info>
        <prop oor:type="xs:string" oor:name="Prop1">
            <value>Value1</value>
        </prop>
        <prop oor:type="xs:string" oor:name="Prop2">
            <value>Value2</value>
        </prop>
        <prop oor:type="xs:string" oor:name="Prop3">
            <value>Enter value</value>
        </prop>
        <prop oor:type="xs:string" oor:name="Prop4">
            <value>Enter value</value>
        </prop>
    </group>
  </component>
</oor:component-schema>
```

The first panel of the wizard will allow you to select the panel name. This panel name will get registered in the panel sequence for this installer.



New File

Steps

1. Choose File Type
2. **Select template name and data model**
3. Arrange panel contents

Select template name and data model

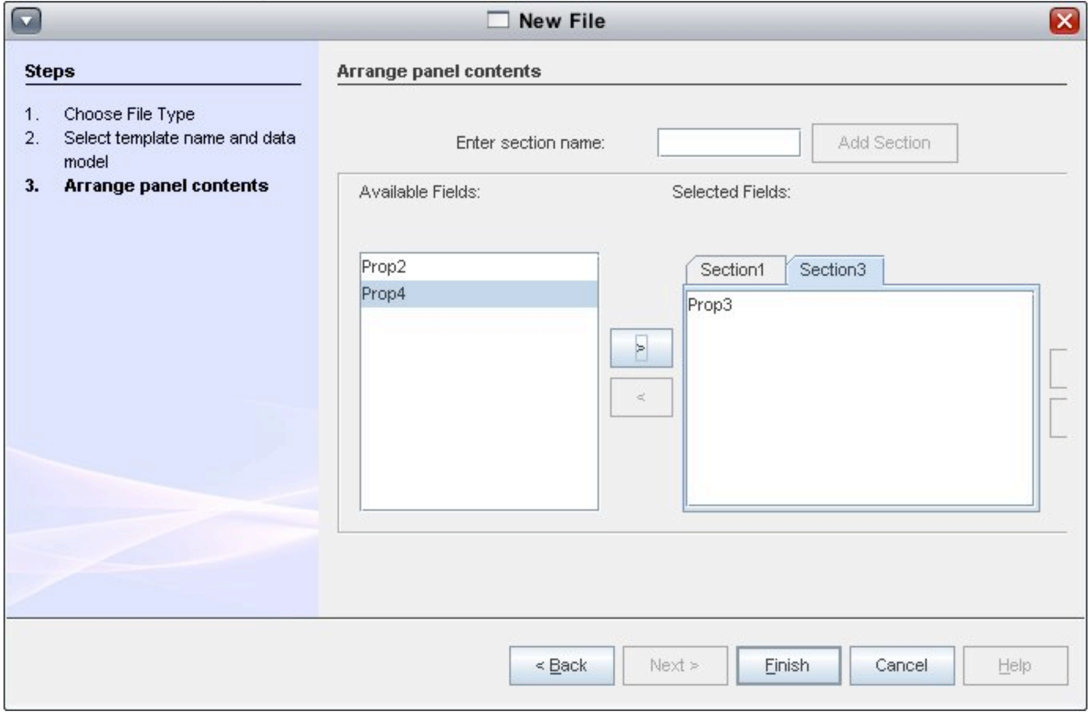
Panel view name: NewPanel

For Eg. LicencePanel

Data model : NewComponent

Configuration Profile: NewComponent

Based on the data model selection the second wizard screen (below) allows to select the individual properties from the underlying data model and include them in sections.



New File

Steps

1. Choose File Type
2. Select template name and data model
3. **Arrange panel contents**

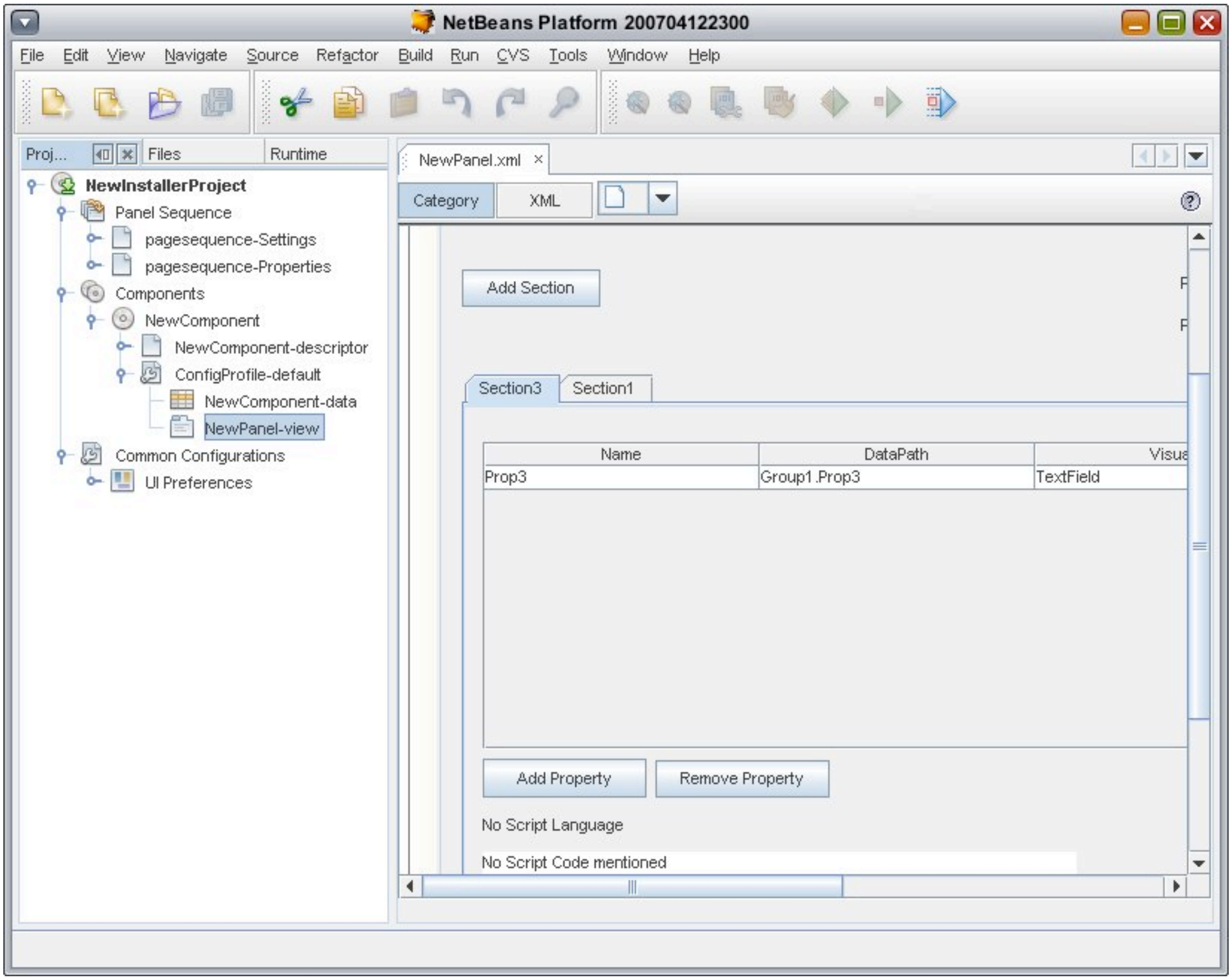
Arrange panel contents

Enter section name:

Available Fields: Prop2, Prop4

Selected Fields: Section1, Section3 — Prop3

The following screen shots shows the newly created view in editor.

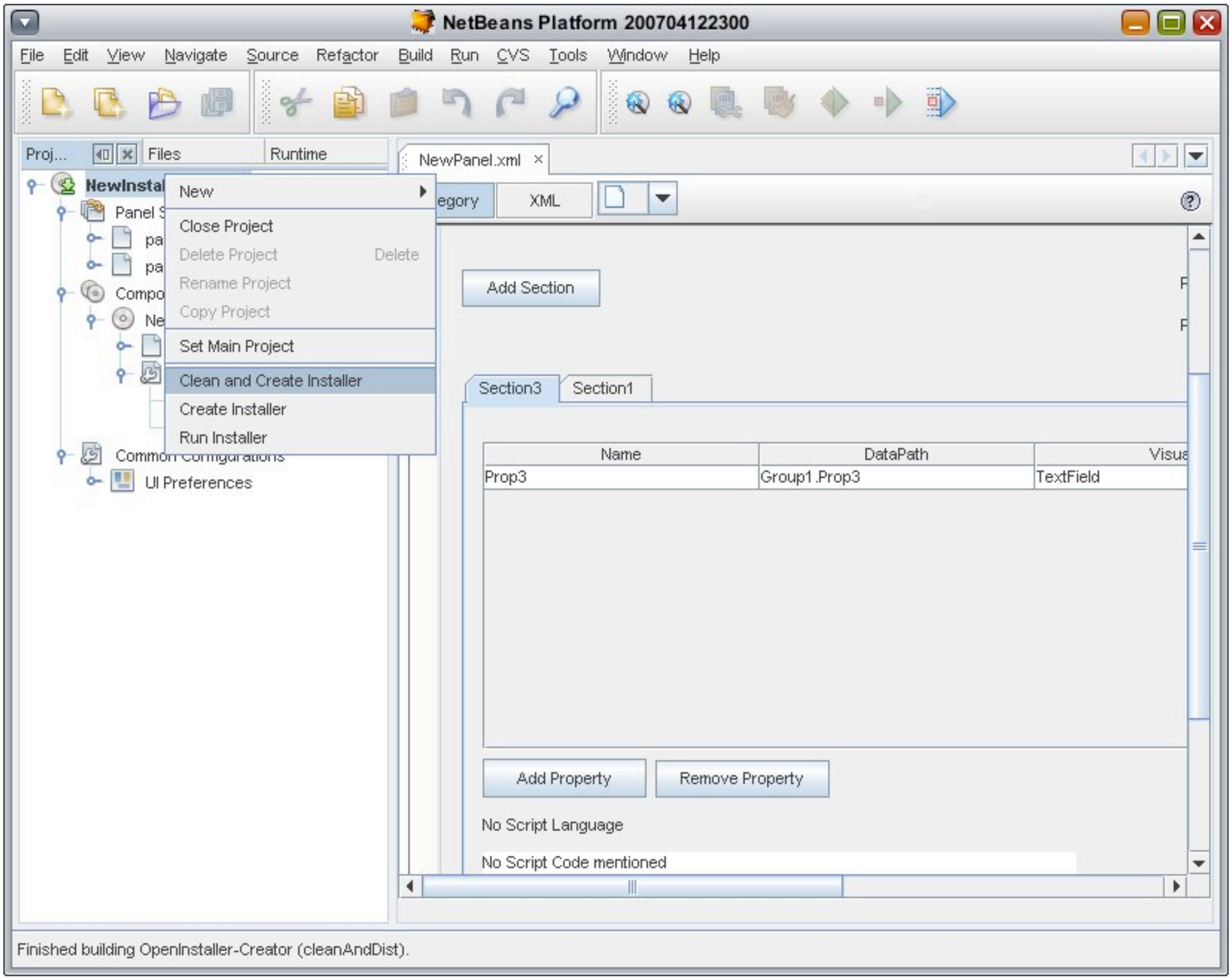


Name	DataPath	Visua
Prop3	Group1.Prop3	TextField

No Script Language

No Script Code mentioned

Now, let's build the installer. To do this, right click on the project node and select 'Create Installer'. This will build the installer and keeps it ready under the project folder. It also bundles the openInstaller framework runtime and packages it.

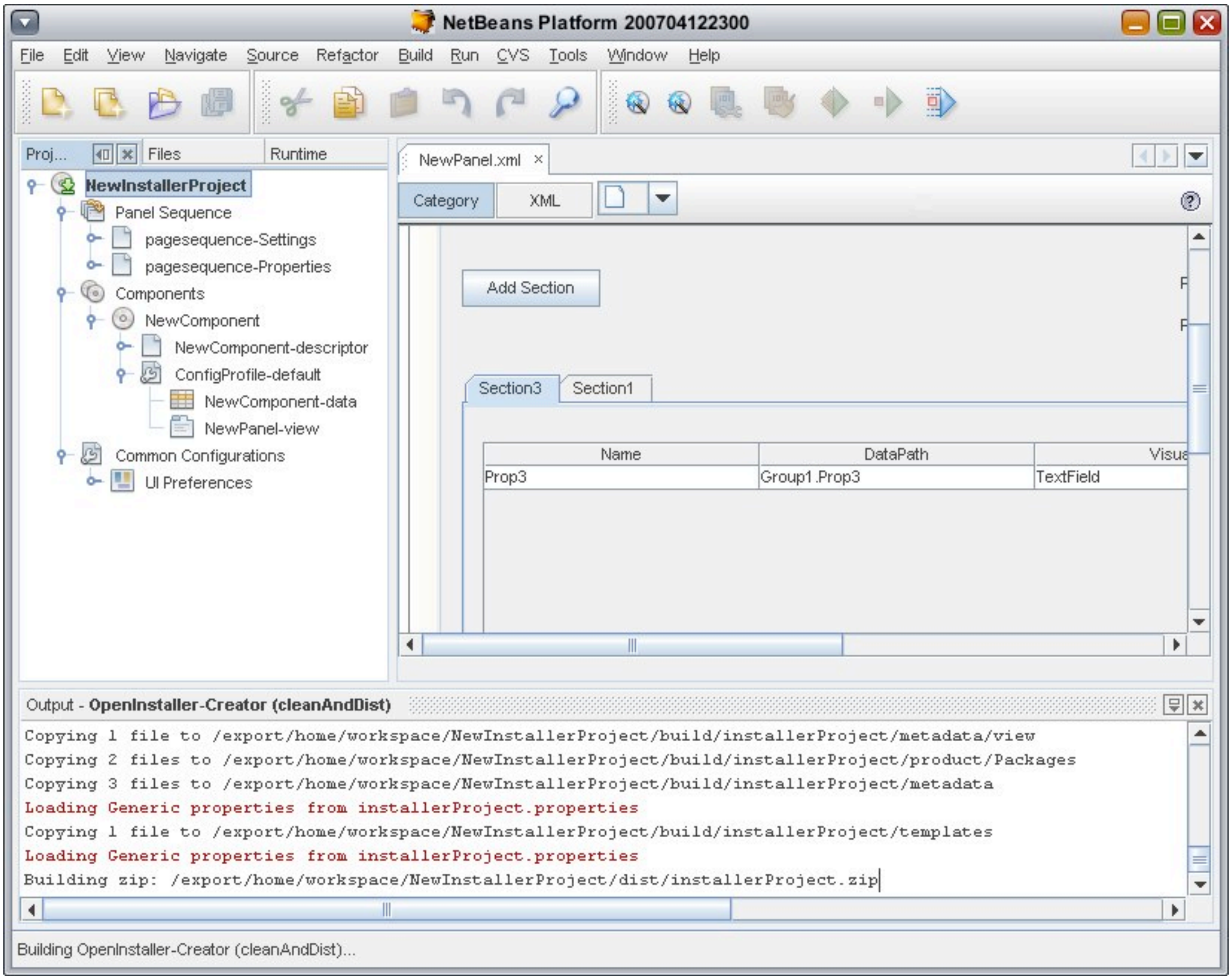


Name	DataPath	Visua
Prop3	Group1.Prop3	TextField

No Script Language

No Script Code mentioned

Finished building OpenInstaller-Creator (cleanAndDist).

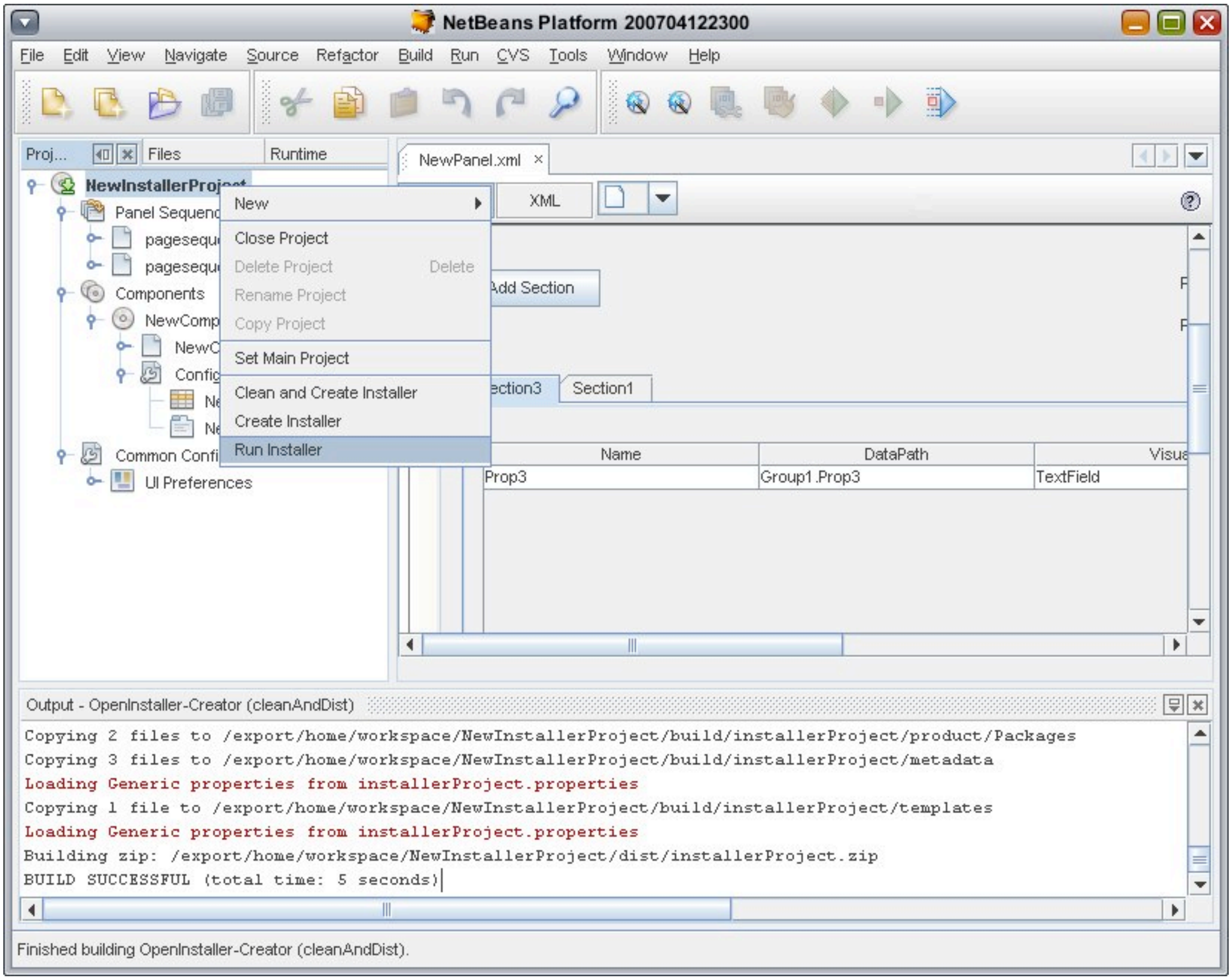


Name	DataPath	Visua
Prop3	Group1.Prop3	TextField

```
Copying 1 file to /export/home/workspace/NewInstallerProject/build/installerProject/metadata/view
Copying 2 files to /export/home/workspace/NewInstallerProject/build/installerProject/product/Packages
Copying 3 files to /export/home/workspace/NewInstallerProject/build/installerProject/metadata
Loading Generic properties from installerProject.properties
Copying 1 file to /export/home/workspace/NewInstallerProject/build/installerProject/templates
Loading Generic properties from installerProject.properties
Building zip: /export/home/workspace/NewInstallerProject/dist/installerProject.zip
```

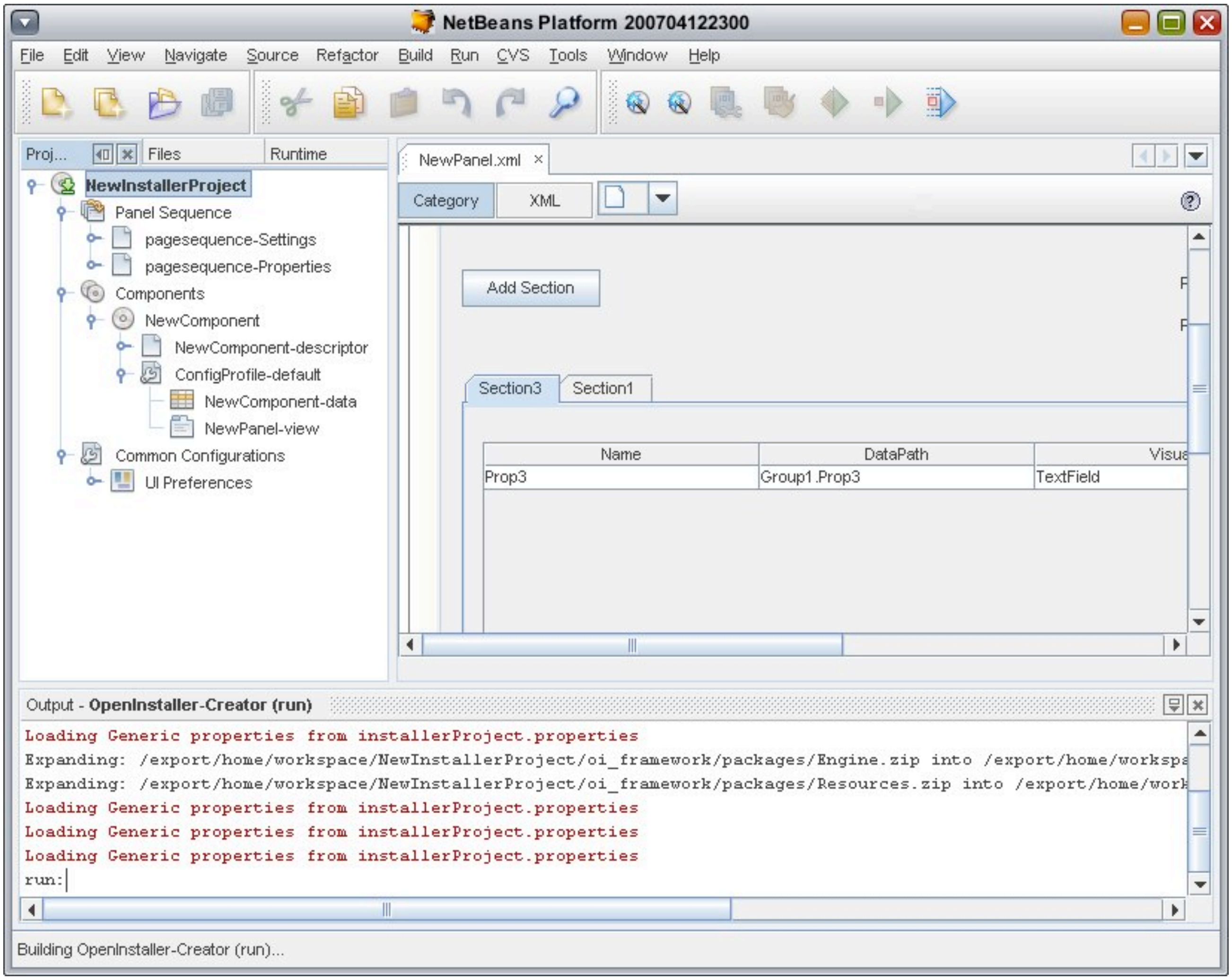
Building OpenInstaller-Creator (cleanAndDist)...

Now that we have built the installer, lets run it to see how it looks. Right click on the project node and select 'Run Installer'. The following screen shots shows the process in action and the last screen shot shows the Installer running.



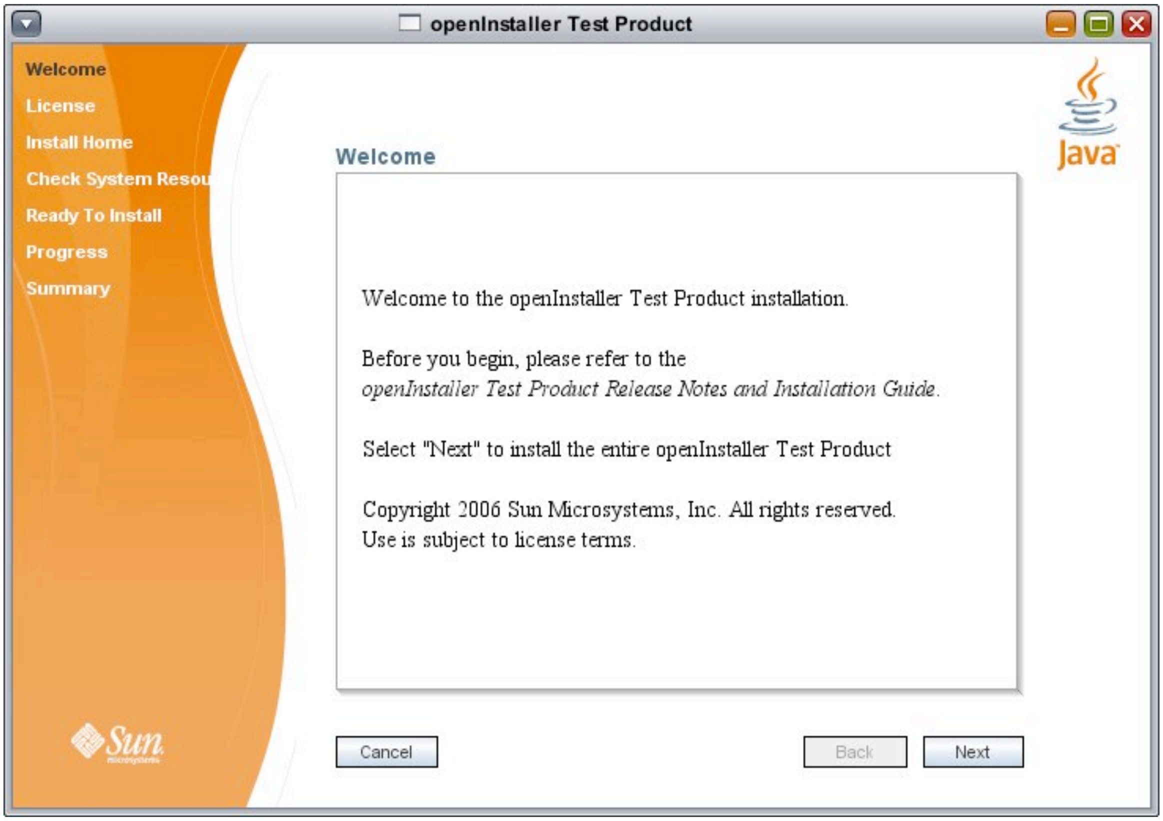
```
Copying 2 files to /export/home/workspace/NewInstallerProject/build/installerProject/product/Packages
Copying 3 files to /export/home/workspace/NewInstallerProject/build/installerProject/metadata
Loading Generic properties from installerProject.properties
Copying 1 file to /export/home/workspace/NewInstallerProject/build/installerProject/templates
Loading Generic properties from installerProject.properties
Building zip: /export/home/workspace/NewInstallerProject/dist/installerProject.zip
BUILD SUCCESSFUL (total time: 5 seconds)
```

Finished building OpenInstaller-Creator (cleanAndDist).



Name	DataPath	Visua
Prop3	Group1.Prop3	TextField

```
Loading Generic properties from installerProject.properties
Expanding: /export/home/workspace/NewInstallerProject/oi_framework/packages/Engine.zip into /export/home/workspspa
Expanding: /export/home/workspace/NewInstallerProject/oi_framework/packages/Resources.zip into /export/home/work
Loading Generic properties from installerProject.properties
Loading Generic properties from installerProject.properties
Loading Generic properties from installerProject.properties
run:
```



Welcome

Welcome to the openInstaller Test Product installation.

Before you begin, please refer to the
openInstaller Test Product Release Notes and Installation Guide.

Select "Next" to install the entire openInstaller Test Product

Copyright 2006 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.

This concludes the operation of openInstaller Developer Tools. For more information, please look at the following sites:

www.openInstaller.org
http://wiki.glassfish.java.net/Wiki.jsp?page=OpenInstallerDeveloperTools