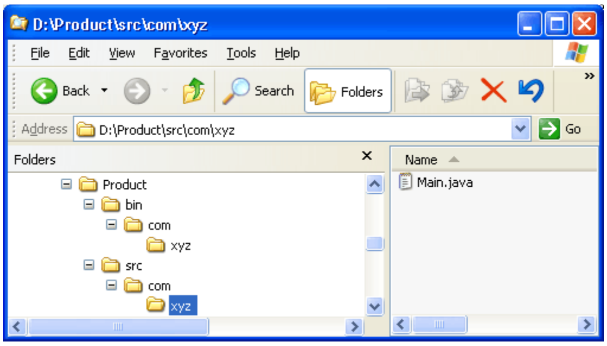
Detecting existing layout

START: The source files for the company's product are laid out in one directory "src" and class files are output in another directory "bin":


D:\Product\src\com\xyz

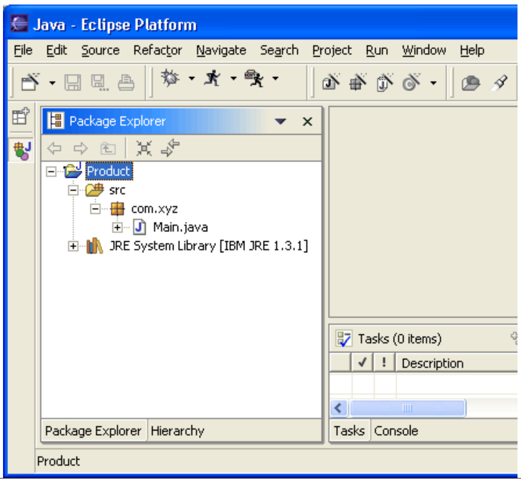
Address D:\Product\src\com\xyz

Folders
- Product
  - bin
    - com
      - xyz
  - src
    - com
      - xyz

Name
Main.java

END: You'll have a Java project with a "src" folder which contains the sources of the "Product" directory.

Note: This solution creates a ".project" file and a ".classpath" file in the "Product" directory. If you do not wish to have these files in the "Product" directory, you should use linked folders as shown in the Sibling products in a common source tree section.

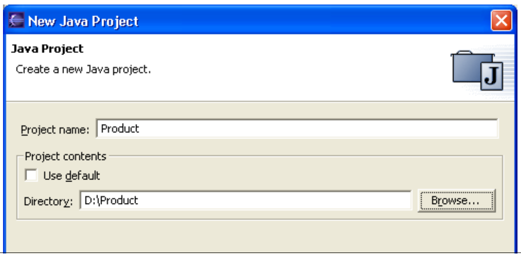

Java - Eclipse Platform

Package Explorer
- Product
  - src
    - com.xyz
      - Main.java
  - JRE System Library [IBM JRE 1.3.1]

Tasks (0 items)

Product

On the next page, type "Product" in the **Project name** field. Uncheck **Use default**. Click **Browse...** and choose the "Product" directory.


New Java Project

Java Project

Create a new Java project.

Project name: Product

Project contents

□ Use default

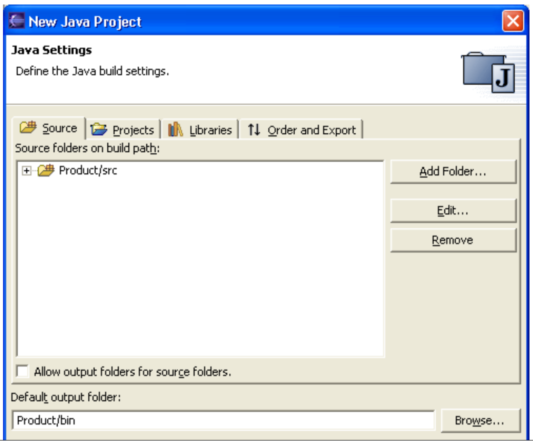
Directory: D:\Product

Click **Next**. Ensure that the source and output folders are detected.

NOTE: The auto-detection works only if the source files contain the correct package statement, i.e. for "Main.java":

```
package com.xyz;
```

Warning: If the preference **Window** > **Preferences** > **Java** > **Compiler** > **Build Path** > **Scrub output folder on full build** is checked, clicking **Finish** will scrub the "bin" directory in the file system before generating the class files.


New Java Project

Java Settings

Define the Java build settings.

Source folders on build path:
- Product/src

□ Allow output folders for source folders.

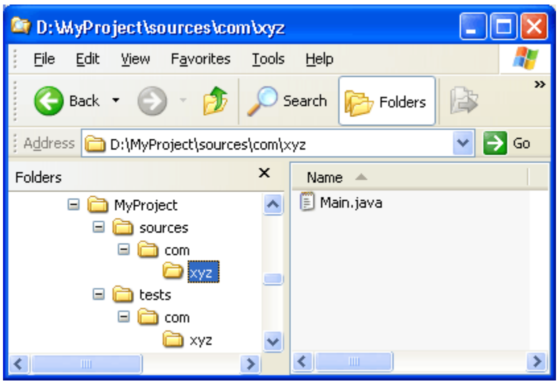
Default output folder:

Product/bin

Click **Finish**.

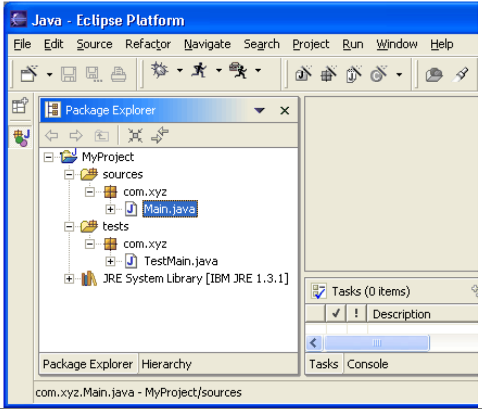
Organizing sources

START: You will create a new Java project and organize your sources in separate folders. Let's assume you want to put your sources in one folder and your tests in another folder:


D:\MyProject\sources\com\xyz

Address: D:\MyProject\sources\com\xyz

END: You now have a Java project with a "sources" and a "tests" folders. You can start adding classes to these folders or you can copy them using drag and drop.

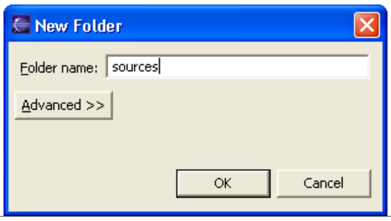

Java - Eclipse Platform

com.xyz.Main.java - MyProject/sources

On the **next** page, **type** "MyProject" in the **Project name** field and click **Next**.

Click **Add Folder...**. On the next dialog, click **Create New Folder...**

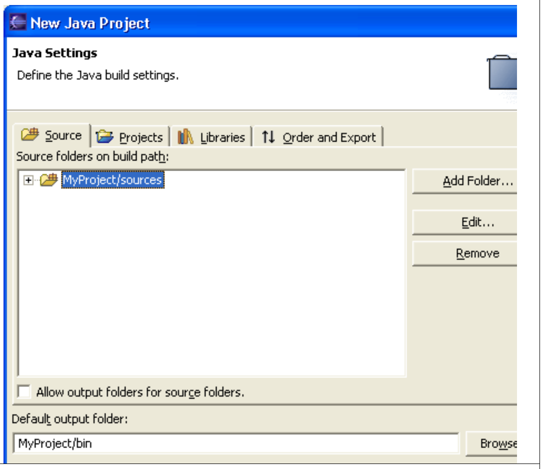
Type "sources" in the **Folder name** field.


New Folder

Folder name: sources

Click **OK** twice to close the two dialogs.

You will be prompted whether or not to remove the project as a source folder and update the build output folder to "MyProject/bin". The builder will write the generated .class files to the output folder. Click **Yes**


New Java Project

Java Settings

Define the Java build settings.

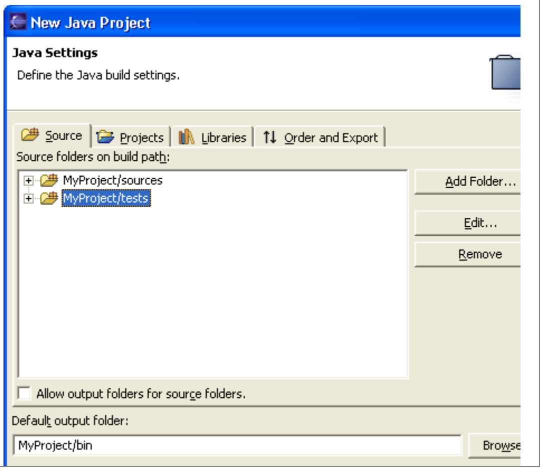
Source folders on build path:

MyProject/sources

□ Allow output folders for source folders.

Default output folder: MyProject/bin

To create the "tests" folder, repeat steps 2 to 5. Then click **Finish**.


New Java Project

Java Settings

Define the Java build settings.

Source folders on build path:

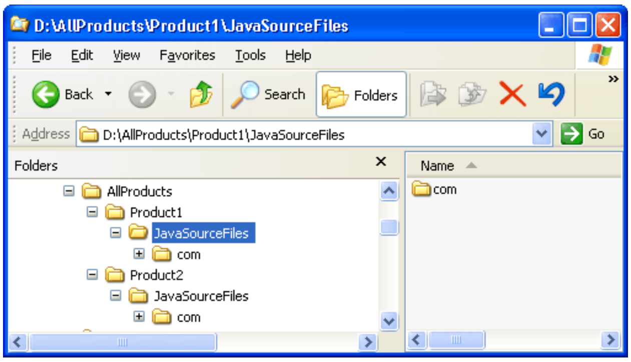
MyProject/sources
MyProject/tests

□ Allow output folders for source folders.

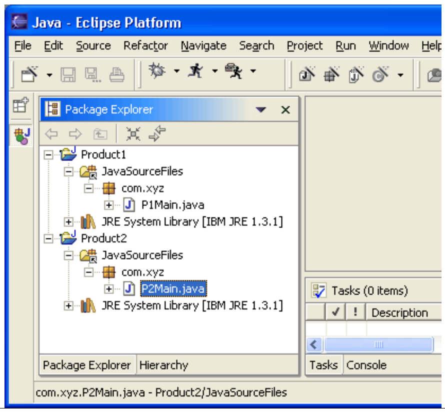
Default output folder: MyProject/bin

Sibling products in a common source tree

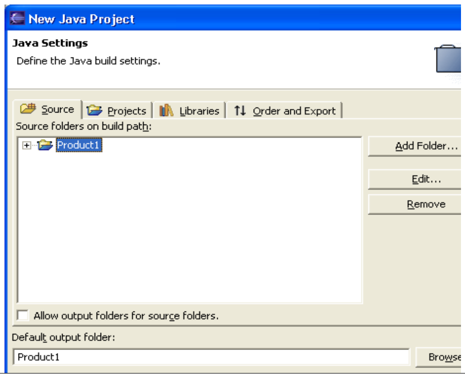
START: The source files for the company's products are laid out in one big directory that is version and configuration managed outside Eclipse:



END: You now have two Java projects which respectively contain the sources of "Product1" and "Product2".



On the next page, type "Product1" in the **Project name** field. Click **Next**.

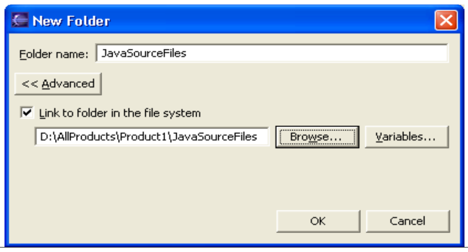


Default output folder: Product1

Select "Product1" source folder and click **Edit...**.

Click **Create New Folder...**. Type "JavaSourceFiles" in the **Folder name** field.

Click **Advanced**. Check **Link to folder in the file system**. Click **Browse....** and choose the "JavaSourceFiles" directory in "Product1".



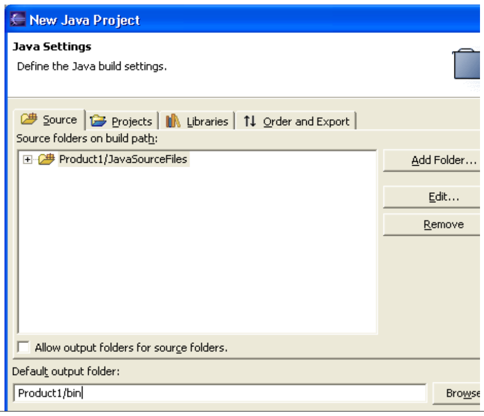
Folder name: JavaSourceFiles

☑ Link to folder in the file system

D:\AllProducts\Product1\JavaSourceFiles

Click **OK** twice to close the two dialogs.

Type "Product1/bin" in the **Default output folder** field.



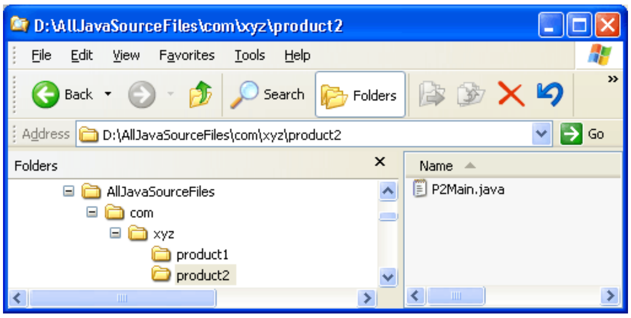
Default output folder: Product1/bin

Click **Finish**.

Repeat these steps for "Product2".

Overlapping products in a common source tree

START: The Java source files for the company's products are all held in a single package directory like this:

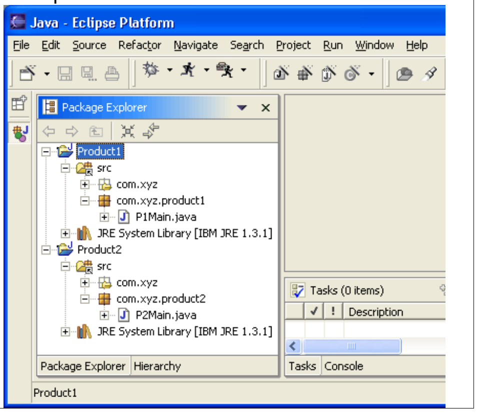

D:\AllJavaSourceFiles\com\xyz\product2
File Edit View Favorites Tools Help
Back
Search
Folders
Go
Folders
Name

Address: D:\AllJavaSourceFiles\com\xyz\product2

AllJavaSourceFiles
- com
  - xyz
    - product1
    - product2

P2Main.java

END: You now have two Java projects, which respectively contain the sources of "product1" and "product2".


Java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Tasks (0 items)
Description
Package Explorer Hierarchy
Tasks Console
Product1

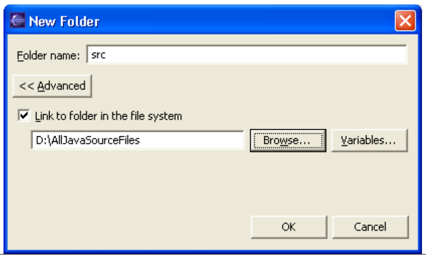
- Product1
  - src
    - com.xyz
    - com.xyz.product1
      - P1Main.java
  - JRE System Library [IBM JRE 1.3.1]
- Product2
  - src
    - com.xyz
    - com.xyz.product2
      - P2Main.java
  - JRE System Library [IBM JRE 1.3.1]

On the next page, type "Product1" in the **Project name** field. Click **Next**.

Select "Product1" source folder and click **Edit...**.

Click **Create New Folder...**. Type "src" in the **Folder name** field.

Click **Advanced**. Check **Link to folder in the file system**. Click **Browse....** and choose the "AllJavaSourceFiles" directory.



Folder name: src

☑ Link to folder in the file system

D:\AllJavaSourceFiles

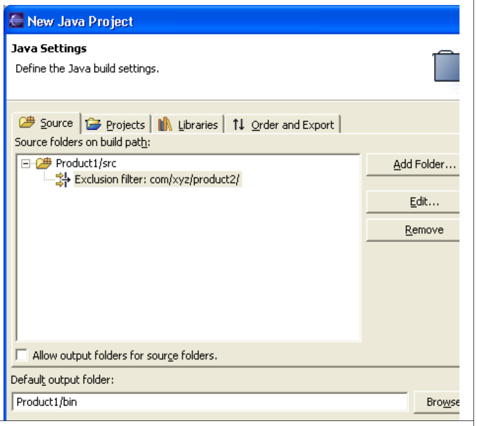
Click **OK** twice to close the two dialogs.

Expand the "src" source folder. Select **Exclusion filters** and click **Edit...**.

Click **Add...**. Type "com/xyz/product2/" in the **Add Exclusion Pattern** dialog.

Click **OK** twice to close the two dialogs.

Type "Product1/bin" in the **Default output folder** field.



Java Settings

Define the Java build settings.

Source folders on build path:

- Product1/src
  - Exclusion filter: com/xyz/product2/

☐ Allow output folders for source folders.

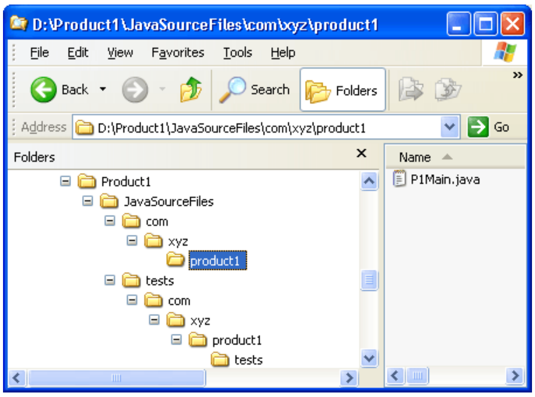
Default output folder: Product1/bin

Click **Finish**.

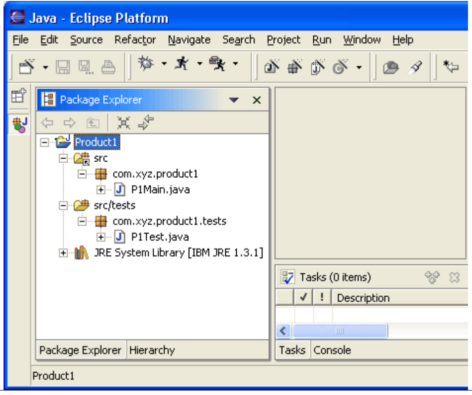
Repeat these **steps** for "Product2", excluding "com/xyz/product1/" instead.

Product with nested tests

START: The Java source files for a product are laid out in package directory like this:


D:\Product1\JavaSourceFiles\com\xyz\product1
File Edit View Favorites Tools Help
Back Search Folders
Address D:\Product1\JavaSourceFiles\com\xyz\product1 Go
Folders
Product1
JavaSourceFiles
com
xyz
product1
tests
com
xyz
product1
tests
Name
P1Main.java

END: You now have a Java project with a "src" folder and a "tests" folder which contain respectively the "JavaSourceFiles" directory and the "tests" directory.

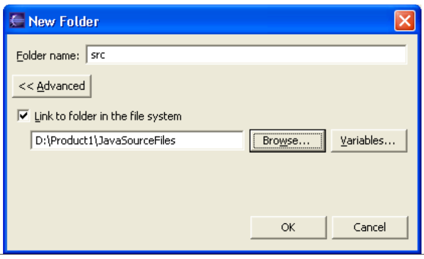


On the next page, type "Product1" in the **Project name** field. Click **Next**.

Select "Product1" source folder and click **Edit...**.

Click **Create New Folder...**. Type "src" in the **Folder name** field.

Click **Advanced**. Check **Link to folder in the file system**. Click **Browse....** and choose the "JavaSourceFiles" directory.



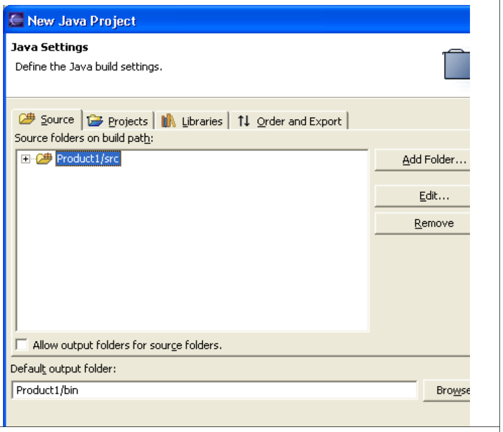
Folder name: src

☑ Link to folder in the file system

D:\Product1\JavaSourceFiles

Click **OK** twice to close the two dialogs.

Type "Product1/bin" in the **Default output folder** field.

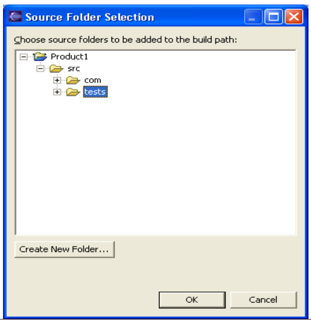


☐ Allow output folders for source folders.

Default output folder: Product1/bin

Click **Add Folder...**.

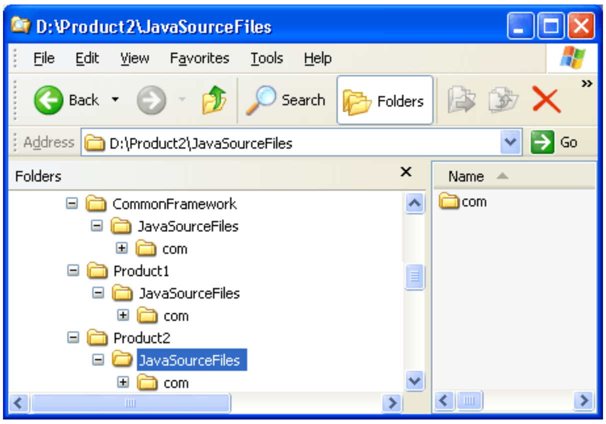
Expand "Project1", then "src" and select "tests".



Click **OK**. You get **an** information dialog saying that exclusion filters have been added. Click **OK**. Click **Finish**.

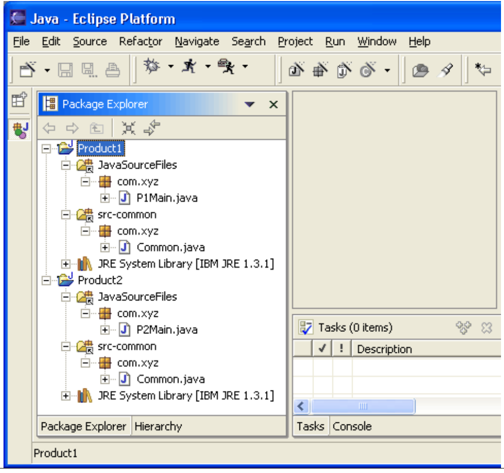
Products sharing a common source framework

START: The Java source files for two products require a common framework:

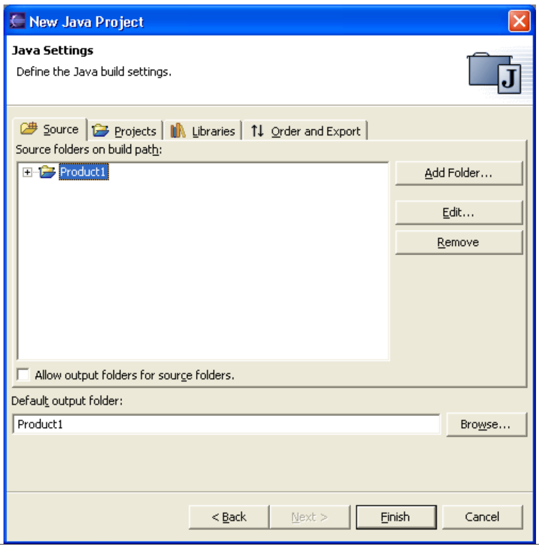


END: You now have two Java projects which respectively contain the sources of "Product1" and "Product2" and which are using the sources of "CommonFramework".

Note: Files in "src-common" are shared. So editing "Common.java" in "Product1" will modify "Common.java" in "Product2". However they are compiled in the context of their respective projects. Two "Common.class" files will be generated; one for each project. If the two projects have different compiler options, then different errors could be reported on each "Common.java" file.



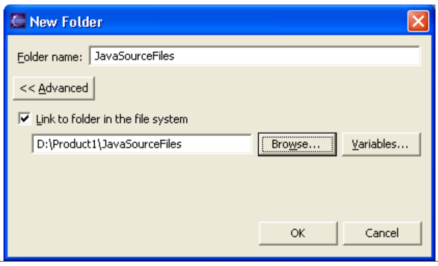
On the next page, type "Product1" in the **Project name** field. Click **Next**.


New Java Project
Java Settings
Define the Java build settings.
Source Projects Libraries Order and Export
Source folders on build path:
Product1
Add Folder...
Edit...
Remove
Allow output folders for source folders.
Default output folder:
Product1
Browse...
< Back
Next >
Finish
Cancel

Select "Product1" source folder and click **Edit...**.

Click **Create New Folder...**. Type "JavaSourceFiles" in the **Folder name** field.

Click **Advanced**. Check **Link to folder in the file system**. Click **Browse....** and choose the "JavaSourceFiles" directory in "Product1".

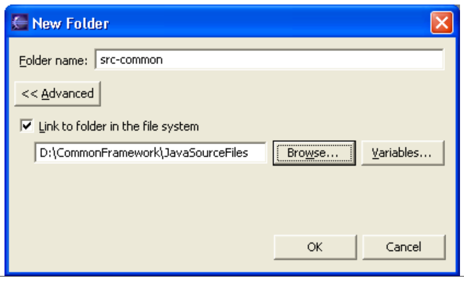

New Folder
Folder name: JavaSourceFiles
<< Advanced
☑ Link to folder in the file system
D:\Product1\JavaSourceFiles
Browse...
Variables...
OK
Cancel

Click **OK** twice to close the two dialogs.

Click **Add Folder...**.

Click **Create New Folder...**. Type "src-common" in the **Folder name** field.

Click **Advanced**. Check **Link to folder in the file system**. Click **Browse....** and choose the "JavaSourceFiles" directory in "CommonFramework"



New Folder

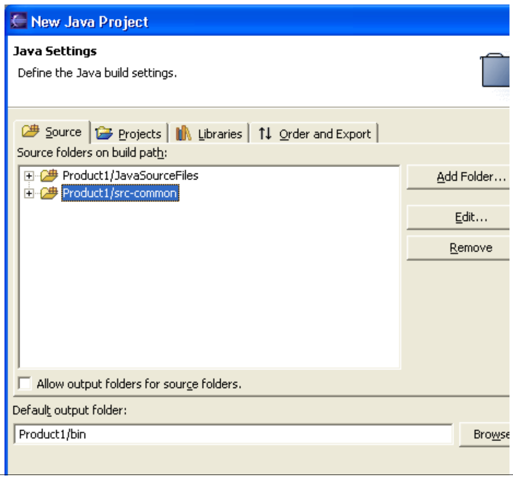
Folder name: src-common

☑ Link to folder in the file system

D:\CommonFramework\JavaSourceFiles

Click **OK** twice to close the two dialogs.

Type "Product1/bin" in the **Default output folder** field.



New Java Project

Java Settings

Define the Java build settings.

Source folders on build path:

- Product1/JavaSourceFiles
- Product1/src-common

□ Allow output folders for source folders.

Default output folder:

Product1/bin

Click **Finish**.

Repeat these steps for "Product2".