

# **MAS / ProvideX Plug-in**

User Guide

February 25, 2013

# Contents

- Introduction ..... 1
  - Operating Systems..... 1
    - MS-Windows ..... 1
    - UNIX / Linux..... 1
    - Mac OS X ..... 1
  - Prerequisites ..... 2
- Install Prerequisites ..... 3
  - Java Run-time Environment ..... 3
  - ProvideX ..... 3
    - Sage 100 ERP Developers ..... 3
- Install Eclipse..... 4
  - Get Eclipse Distribution ..... 4
    - Standard Eclipse Distributions..... 4
    - Third-party Pre-built Distributions ..... 4
  - Unpack / Install Eclipse Distribution ..... 5
  - Documentation and Tutorials ..... 5
- Install Eclipse Plug-ins..... 6
  - Eclipse – Software Update ..... 6
  - Sage 100 ERP/ ProvideX Plug-ins..... 7
  - Version Control..... 10
    - Version Control – CVS ..... 11
    - Version Control - Subversion..... 13
    - Version Control – MKS ..... 16
      - Set Environment Variables for MKS..... 16
      - MKS Plug-in Install ..... 17
      - MKS Plug-in Features ..... 18
      - MKS Plug-in Limitations ..... 18
  - Configure Eclipse Environment ..... 19
- Using Eclipse – Sage 100 ERP Projects and MKS ..... 23
  - Sage 100 ERP Perspective ..... 23

Create Sage 100 ERP Project .....	25
Disable Automatic Builds .....	26
Assign MKS Sandbox to Project .....	26
Clean Build .....	28
Link Project to SAGE 100 ERP Build .....	29
Edit Build Links.....	29
MKS Plug-in Features and Limitations .....	30
Eclipse – ProvideX and Subversion.....	31
Set Perspective .....	31
Subversion – Connection Protocols .....	32
Specify Repository Location – SVN+SSH .....	32
Specify Repository Location – HTTP .....	33
Accessing a Subversion Repository – New Project Wizard .....	34
Create a New Project – from SVN.....	34
Checkout Project .....	35
Accessing a Subversion Repository – Browse SVN Repository.....	36
SVN Repository View .....	36
Link Project to Sage 100 ERP Build .....	37
Edit Build Links.....	37
Daily Workflow.....	38
Synchronize with Repository.....	38
Synchronize .....	38
Review Modifications - All.....	39
Review Modifications – Selected.....	39
Update – All.....	40
Commit Modifications .....	40
Appendix A - Windows Subversion Clients .....	43
Visual Studio Subversion Client .....	43
Windows Subversion Client .....	43
Appendix B – Troubleshooting.....	44
Resources do not show Version Control information .....	44
ProvideX Builder does not create Program Files.....	44

How to copy files to existing application folder ..... 45

- Create Project..... 45
- Create a Linked Folder ..... 46
- Link to Project Folder ..... 47

Program is created, but not updated in a Linked Folder ..... 47

## Introduction

The *Eclipse* IDE provides a common development environment for application development across a large number of programming languages.

The *MAS / ProvideX Plug-ins* enable application development for ProvideX and MAS 90/200 programmers within the Eclipse IDE framework.

## Operating Systems

The ProvideX Plug-in for Eclipse can be used on most of the operating systems that support Eclipse when the prerequisites (see below) are met.

It has been tested on:

- Linux :
  - o Red Hat Enterprise Linux: 4 (CentOS 4), 5 (CentOS 5)
  - o SUSE Linux Enterprise Server: 10, 11
  - o Ubuntu 8.04 LTS
- MS-Windows:
  - o XP, Vista, Windows 7 (32-bit, 64-bit)
- Apple Mac OS X (Intel)
  - o Snow Leopard (10.6.x), or newer

### MS-Windows

For a MS-Windows system, the download file will be a ZIP file that must be unpacked by the user (or an administrator) after the download has completed. Be sure to unpack the ZIP file with the “Use Folder Names” option set to maintain the folder structure as it is in the file.



On Windows 7 or Server 2008 that have *User Account Controls (UAC)* active, do not unpack the file in the “Program Files” folder.

### UNIX / Linux

For UNIX/Linux operating systems, the download file will be a compressed TAR archive file that must be unpacked by the user (or an administrator) after the download has completed

### Mac OS X

For Mac OS X operating system, the download file will be a compressed TAR archive file. This file must be unpacked by the user (or an administrator) after the download has completed.



The new Gatekeeper feature of OS/X Mountain Lion may not allow execution of the application. If this happens, right-click the application and select ‘Open’ from the menu – Gatekeeper will remember and not prompt again.

## ***Prerequisites***

Even though this document concentrates on MS-Windows platforms, most of the information does apply to installing Eclipse on other operating systems as long as all of the prerequisites are met.

The Eclipse IDE and ProvideX Plug-in require the following prerequisites:

- Java SE 6 (or newer) Run-time Environment
  - o The latest JRE for most operating systems can be found on the Java web site.
  - o With the release of Java SE 7, Java for OS X is now available from Oracle.
- Eclipse 3.6 (or newer); although it is recommended that a current version of Eclipse be used to ensure that all capabilities work as expected
  - o The Java Development Tools (JDT) plug-in
- ProvideX v7.x (or newer)
  - o Professional bundle
  - o 2 user developer license; a demo license can be used on a MS-Windows system but the normal demonstration system nag message will be displayed.

# Install Prerequisites

## *Java Run-time Environment*

Simply go to the Java web site (<http://www.java.com/>).

Before selecting a file to download, check to see if a Java Runtime Environment is already installed on your system using the link on this web page. This link will check for a Java Runtime Environment and display information about any version that is found.

If there is none installed, or it does not meet the prerequisites that were specified earlier, choose an appropriate JRE for your operating system – download and install it on your workstation.



If you are planning to do Java development, you should download and install the Java JDK instead. The JDK includes a Java Run-time Environment.

## *ProvideX*

If you do not already have a version of ProvideX installed on your workstation, download and install the most recent version of ProvideX that is compatible with your application.



A two-user developer license will be required to use this ProvideX with the ProvideX Plug-in for Eclipse. A “demo” activation can be used but you will be restricted by the limitations of that license and there will be regular nag messages.

## **Sage 100 ERP Developers**

For Sage 100 ERP developers, a full Sage 100 ERP application installation plus developer tools is required on the workstation. Also, a MAS developer activation key is required.

# Install Eclipse

## *Get Eclipse Distribution*

### Standard Eclipse Distributions

Download an Eclipse distribution from the Eclipse web site ( <http://www.eclipse.org> ). Be sure to choose the distribution that best matches the type of development that you will be performing. If you are planning to only develop using the MAS/ProvideX Plug-in, the distribution for Java developers is the smallest one that includes the pre-requisites that are required by the ProvideX Plug-in.

There are a large number of distributions available for download that have been created for specific development environments. It is best to review this list and choose the one that is closest to your requirements. If the selected distribution does not include all of the pre-requisites for the ProvideX Plug-in, these can be installed using the Software Installation mechanism that is included with Eclipse.

### Third-party Pre-built Distributions

There a large number of web sites providing alternate methods for packaging and distribution management of Eclipse that include:

- Yoxos - <http://eclipsesource.com/en/yoxos/>  
*"Eclipse Management for Professionals".*
- My Eclipse IDE - <http://www.myeclipseide.com/>  
*"The most innovative approach to offering affordable tools for Java and J2EE developers".*

For additional information, visit the Eclipse website and review the documentation and resources that are available there.



## ***Unpack / Install Eclipse Distribution***

Once the distribution has been downloaded to your workstation, simply unpack it on your system.



For a consistent install on multiple workstations with different operating system versions, create a new folder in the root directory of the primary drive (for example: apps) and unpack the file into this folder.

There is no setup program – once the file has been unpacked, the Eclipse install is complete.

Now is a good time to create a shortcut to the Eclipse executable file to make it easier to locate and use later.

## ***Documentation and Tutorials***

Once the installation has completed, this is the time to become more familiar with the various features of Eclipse. Please visit the Eclipse web site to review the available documentation and tutorials:

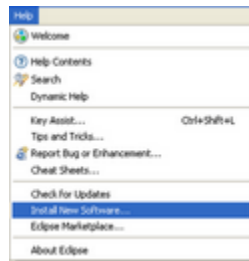
<http://www.eclipse.org/documentation> or the Eclipse.org Wiki – <http://wiki.eclipse.org>.

# Install Eclipse Plug-ins

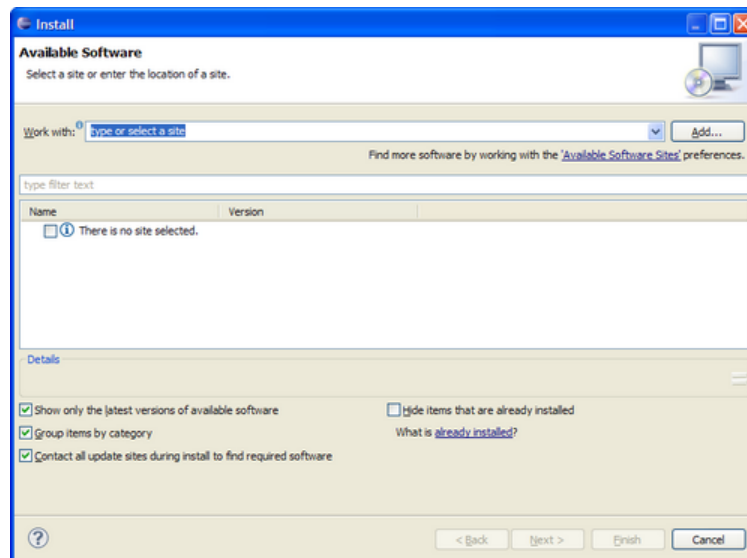
## *Eclipse – Software Update*

Once the Java JRE (or JDK), Eclipse, and ProvideX have been installed, the next step is to install the additional plug-ins that are needed. The simplest method for installing additional plug-ins is to use the “Software Update” capability within Eclipse.

1. Start Eclipse; you will be prompted to select a workspace, simply accept the default value.
2. From the Eclipse menu, choose Help ➤ Install New Software...



3. The “Available Software” screen will display...



## Sage 100 ERP/ ProvideX Plug-ins



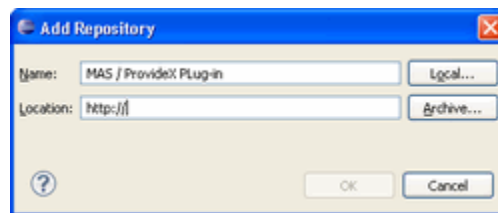
The pre-built distributions already include all plug-ins, skip ahead to the section for installing Version Control client plug-ins.

At a minimum, the ProvideX Plug-in and Sage 100 ERP Plug-in must be installed.

The Eclipse “Software Update” mechanism will be used to install the plug-ins. Start the “Software Update...” process as outlined in the earlier section ‘Eclipse – Software Update’ and follow the steps outlined below.

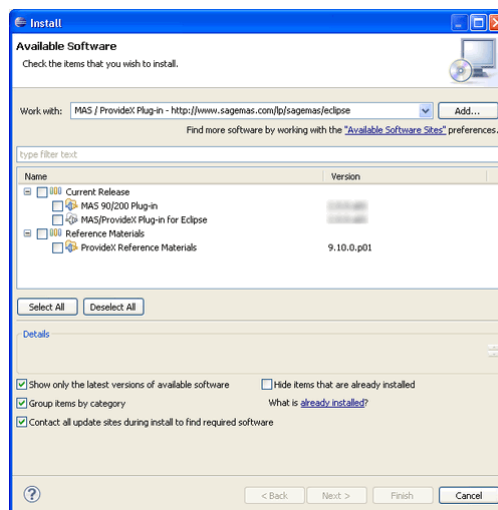
1. Before the Sage 100 ERP / ProvideX Plug-in can be installed, the location of the update site for the plug-in must be defined.

Click [**Add...**] to create a new update site; enter a name for the ProvideX Plug-in update site and the URL for the update site (such as <http://eclipse.sagemas.com>). Click [**OK**] to complete the creation of the repository definition.

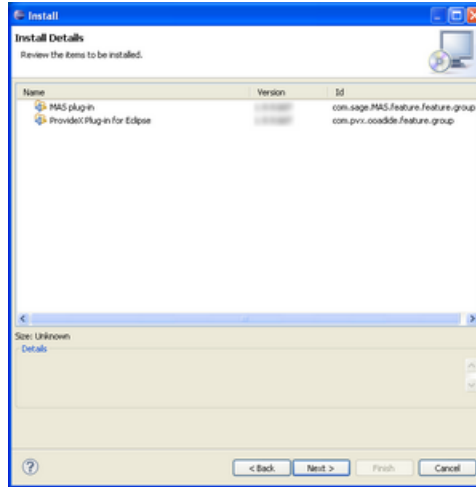


If a ZIP file was downloaded from the Plug-in Update site, it can be used as the installation source by clicking [**Archive...**] and selecting the ZIP file.

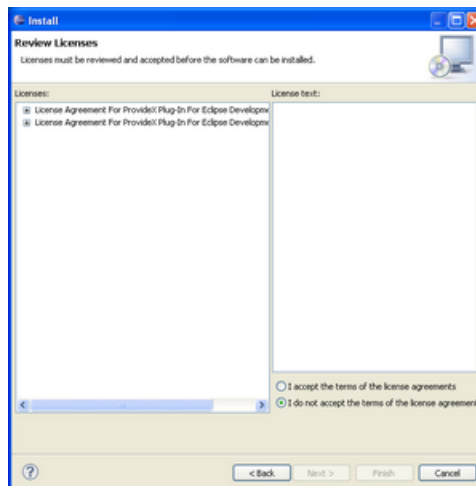
2. Expand the “Current Release” category and select the “ProvideX Plug-in for Eclipse” and “Sage 100 ERP Plug-in”. Click [**Next**] to continue the installation process.



3. A new dialogue will display with a list of the items that were selected for installation. Click **[Next]** to begin the next step of the install process.



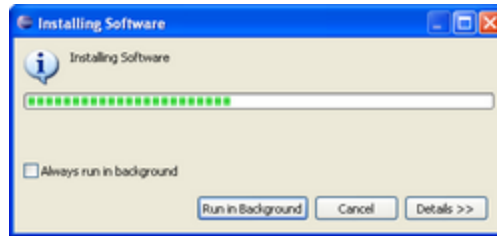
4. The next dialogue will display the licenses for the selected plug-in(s) for you to review.



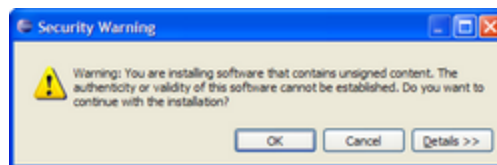
Once you have completed reviewing the licenses, select radio button below the license text to indicate whether you accept the terms of the license agreement.

If you do not accept the terms of the license agreement, you cannot continue and must click **[Cancel]** to abort the installation process. If you accept the terms of the license agreement, click **[Finish]** to continue the installation process.

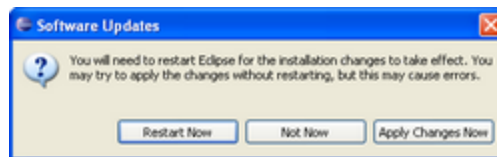
- The files for each of the selected plug-ins will be downloaded and installed on the workstation. A progress window will show the progress of the installation process.



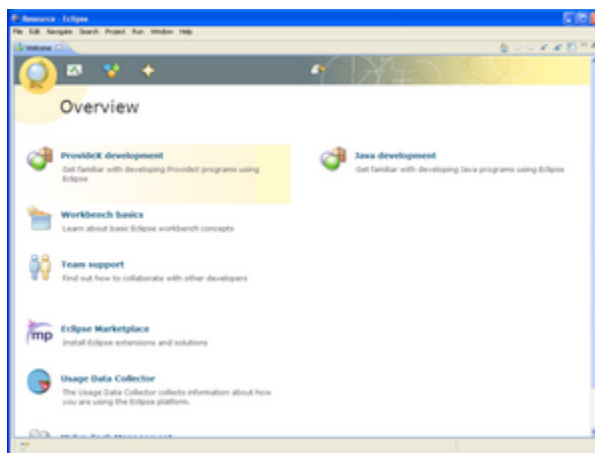
- If the installation package for any of the selected plug-ins is not signed, a security message will be displayed asking you to verify that the software should be installed. You must click **[OK]** to continue with the installation.



- Once the installation is completed, the software update process will ask to restart Eclipse for the modifications to take effect. Click **[Restart Now]** to restart Eclipse.



- Eclipse will restart and display the “Welcome” page which is used to access additional information about some of the registered features of the Eclipse installation.



- The selected plug-ins are now installed; close the welcome page by pressing the button on the toolbar to go to the workbench or by pressing the close button on the “Welcome” page tab.

## ***Version Control***

It is beyond the scope of this document to discuss the merits for a version control system to assist with the management of the source code for your application. It is recommended that any developer who manages a large application use some mechanism to control the source code for that application.

There are a large number of version control systems available today and it is possible to find an Eclipse plug-in for most of them.

The next few sections outline the steps needed to install plug-ins for two common open source version control systems and a third proprietary version control system.

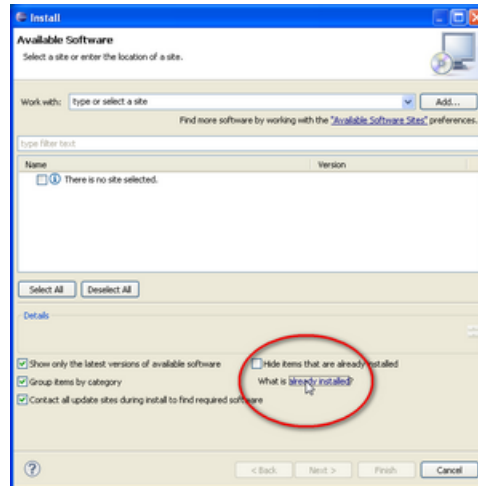
Even though this document cannot cover every version control system that is available, there should be enough information to help with the installation of a plug-in for most of them.

## Version Control – CVS

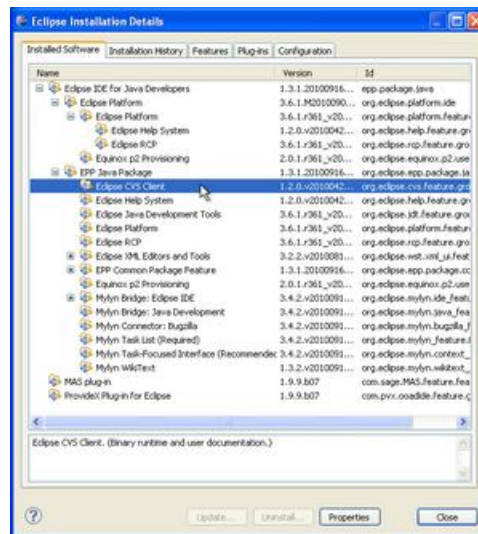
If you do not use CVS to manage a version control repository, skip this step.

If you are not sure that the installed distribution has a CVS client, use the “Software Update” capability to check the installed plug-ins. Start the “Software Update...” process as outlined in the earlier section ‘Eclipse – Software Update’ and follow the steps outlined below.

1. Click on the “already installed” link to view a list of the installed packages.

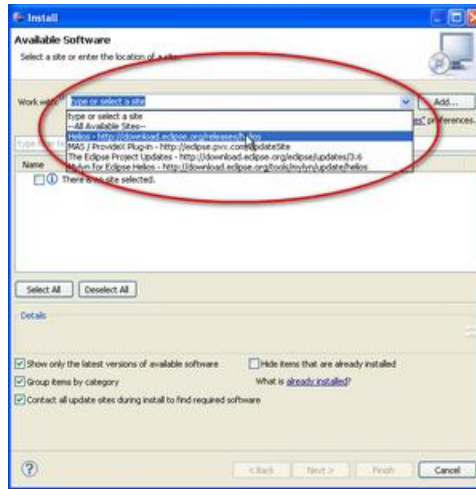


2. Search this list for a plug-in for your version control system – in this case, look for a “CVS Client”. The following image shows the “Eclipse CVS Client” plug-in is already installed.

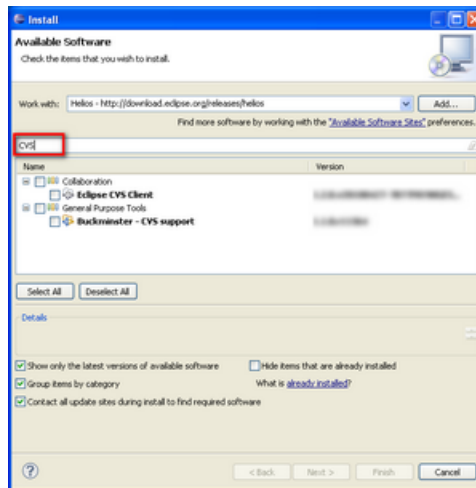


The list of installed software may not be the same as shown below depending on the distribution of Eclipse and additional plug-ins that have been installed.

3. If you were not able to locate a CVS client, close the window containing list of installed software and click on the drop-box to see a list of the defined update sites.



4. Select the site for the installed version of Eclipse – “Helios” in this case.
5. The list of available plug-ins at the selected update site will be downloaded and displayed.
6. Enter “CVS” into the *Filter* box to restrict the list of plug-ins to those that contain the entered text.



7. The list may include several plug-ins in addition to the specific CVS client that we are attempting to locate. Select the CVS client plug-in from the list and click [**Next**].
8. Continue through the installation of the CVS plug-in and respond to the dialogue screens in the same way as for the MAS / ProvideX plug-in installation (above).



## Version Control - Subversion

If you do not use Subversion to manage a version control repository, skip this step.

There are a several choices for Subversion client plug-ins that can be installed. This document will outline the steps for installing the Subversive plug-in that is SVN Team Provider is maintained by the Eclipse project.

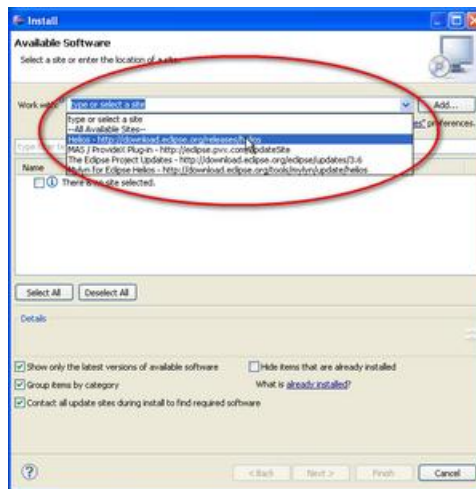


Subclipse is an alternative Eclipse Team Provider plug-in providing support for Subversion within the Eclipse IDE. The software is released under the Eclipse Public License (EPL) 1.0 open source license.

Check the web site (<http://subclipse.tigris.org/>) for the download and installation steps.

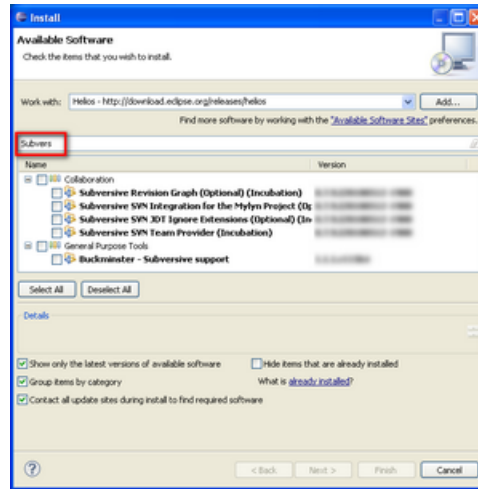
Most Eclipse distributions do not include a Subversion client. Start the “Software Update...” process as outlined in the earlier section ‘Eclipse – Software Update’ and follow the steps outlined below to install the client that is part of the installed version of Eclipse.

1. Click on the drop-box to see a list of the defined update sites.



2. Select the site for the installed version of Eclipse – “Helios” in this case.
3. The list of available plug-ins at the selected update site will be downloaded and displayed.

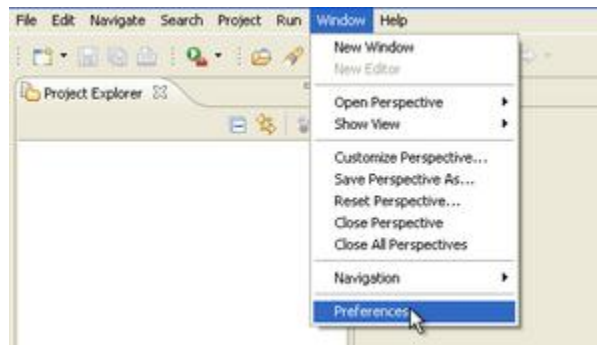
4. Enter “Subvers” into the *Filter* box to restrict the list of plug-ins to those that contain the entered text.



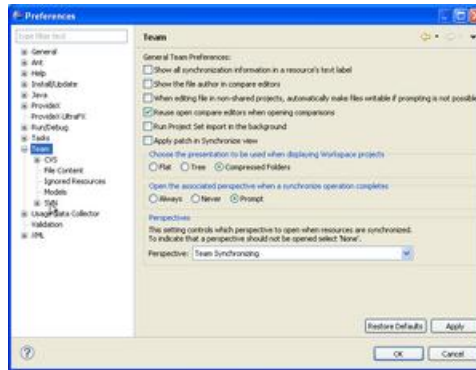
5. The list may include several plug-ins in addition to the specific Subversion client that we are attempting to locate. Select the “Subversive SVN Team Provider” from the list of plug-ins and click **[Next]**.
6. Continue through the installation of the Subversive plug-in and respond to the dialogue screens in the same way as for the MAS / ProvideX plug-in installation (above).

Once Eclipse has restarted, follow the steps outlined below to complete the installation of all of the components required for the Subversive client.

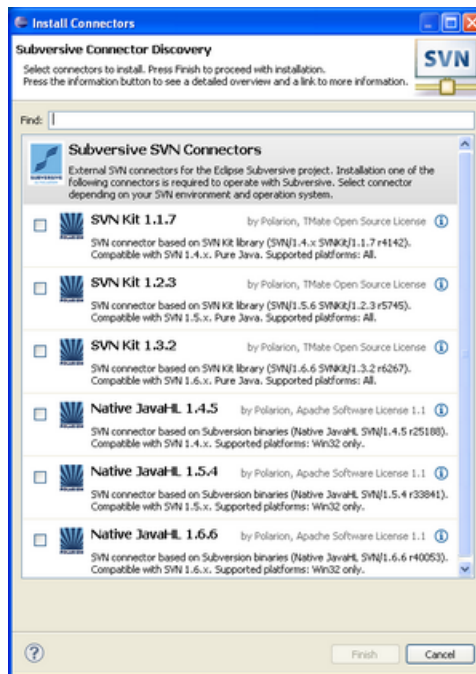
7. From the Eclipse Menu, choose Window ➤ Preferences.



8. Expand the “Team” section and choose the “SVN” preference page.



9. The first time that this page is accessed, the “Subversive Connectors Discovery” dialogue will be displayed.



10. Select the “SVN Kit 1.3.x” connector and click **[Finish]** to begin the installation.

11. Continue through the installation of the plug-in and respond to the dialogue screens in the same way as for the Sage 100 ERP / ProvideX plug-in installation (above).

## Version Control – MKS

If you do not use MKS to manage a version control repository, skip this step.

### Set Environment Variables for MKS

The MKS Source Integrity Eclipse Plug-in uses two Windows environment variables to locate and identify the Eclipse environment and version of the Source Integrity Plug-in that is installed in that environment.

The environment variables are:

<b>ECLIPSE_INSTALL_ROOT</b>	Eclipse installation root directory
<b>SI_ECLIPSE_FEATURE</b>	MKS Integrity Client 3.0 integrations directory

1. Run the Windows *Control Panel* and select the *System* applet.



You can access the “System” control panel applet directly by entering “System.cpl” into:

- the “Run...” dialogue on a Windows XP system or
- “Search Program and Files” box on a Windows 7 system

2. Go to the *Advanced* tab and click the button “Environment Variables”.
3. Under “System Variables”, click **[New]** ...
  - a. Define the variable ‘**ECLIPSE\_INSTALL\_ROOT**’ and set the value to the directory where Eclipse was installed. Click **[OK]** to save the variable.
  - b. Define the variable ‘**SI\_ECLIPSE\_FEATURE**’ and set the value to the full path to the “MKS/IntegrityClient/integrations/IBM/eclipse\_3.0” folder; the default location is:  
*C:/Program Files/MKS/IntegrityClient/integrations/IBM/eclipse\_3.0*

Click **[OK]** to save the variable.



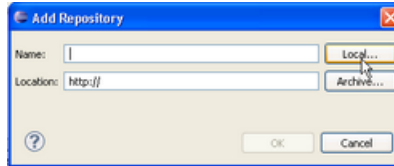
You must use *forward slashes* when setting the values for these variables.

- c. Click **[OK]** to close the “Environment Variables” window.
  - d. Click **[OK]** to close the “System Properties” window.
4. You will need to restart Eclipse so that these new environment variables will be used.

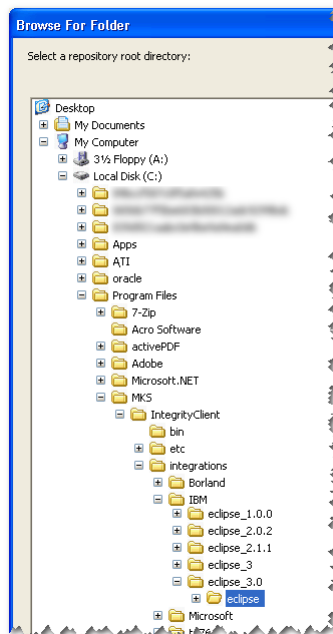
## MKS Plug-in Install

Start the “Software Update...” process as outlined in the earlier section ‘Eclipse – Software Update’ and then follow the steps outlined below to install the MKS Plug-in.

1. Click **[Add...]** to create a new update site. When the “Add Repository” window displays, click **[Local...]**.



2. Browse to the “eclipse” subfolder of the MKS installation on the workstation (C:/Program Files/MKS/IntegrityClient/integrations/IBM/eclipse\_3.0/eclipse/)...



Enter “MKS Plug-in” as the name of the repository and click **[OK]** to update.

3. Uncheck the box next to “Group items by category” to show the contents of this repository.
4. Select “MKS Source Integrity Integration” to install and click **[Next]** to begin the install process.
5. Continue through the installation of the plug-in and respond to the dialogue screens in the same way as for the MAS / ProvideX plug-in installation (above).



After restarting Eclipse, a new “Source Integrity” menu item will appear on the Eclipse menu.

## MKS Plug-in Features

1. In the Navigator view, select a sandbox under a MAS project and navigate the tree view to locate “resources”. Resources are programs, NOMADS libraries, and other files stored in the MKS repository. There are two folders under each MAS project.

<b>compiled</b>	This folder is used by the ProvideX builder to store compiled versions of the source code. Do not access or edit files in this folder – it is exclusively used by the ProvideX builder. If the files in this folder get deleted, you must perform a clean build as described in the section “ <b>Create MAS Project</b> ”.
<b>src</b>	This folder contains the members of the MKS sandbox, arranged under module subprojects. It is from within this folder that you edit programs and Nomads libraries, and interact with MKS to check out and check in sandbox members.

2. Adjacent to each sandbox member in the ‘src’ folder is its current MKS version along with the name of the person to whom it is checked out.
3. To access the various MKS processes that are available, right-click on a project resource and select Team. These same processes are also available from the Source Integrity menu on the Eclipse menu bar.

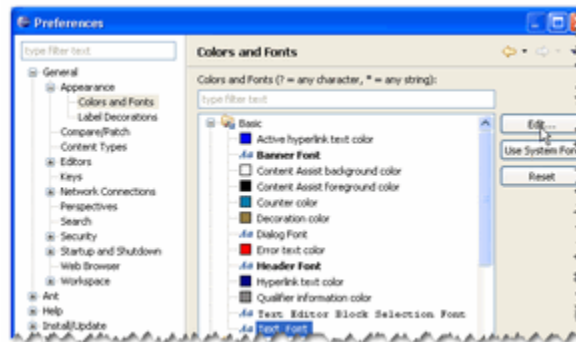
## MKS Plug-in Limitations

Many Integrity Client functions are not available from within Eclipse.

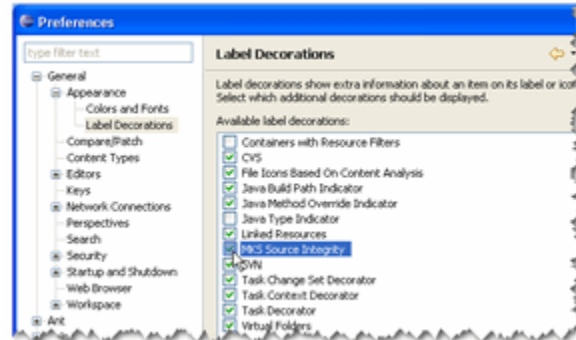
1. For example, you can check members out and back in, but you cannot unlock a member that you have checked out. To do this, select “Launch Source Integrity” from the Team menu and maintain the member using the Integrity Client.
2. You cannot resynchronize entire projects from within Eclipse. You must use the Integrity Client to do mass resynchronizations for an entire project or subproject. Once the synchronization is complete, the changes will be reflected in your Eclipse project.
3. To remove the read-only attribute of an MKS resource, right-click on it, select Properties ➤ Resource, and then clear the Read only checkbox

## Configure Eclipse Environment

1. From the Eclipse Menu, choose Window ➤ Preferences.
2. General ➤ Appearance ➤ **Colors and Fonts**
  - a. If you wish to change the font used by the editor, select Basic ➤ **Text Font**
  - b. Click [**Edit...**] and select a font (either *Consolas* or *Lucida Console* work well)

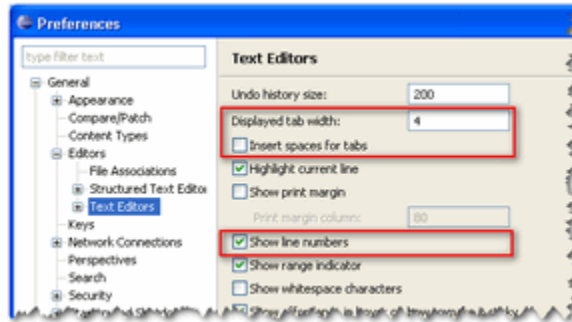


3. General ➤ Appearance ➤ **Label Decorations**
  - a. Select the option for the version control system (“MKS Source Integrity” or “SVN”).



- b. Do not click [**OK**] to close the preferences since additional preference settings need to be checked or modified.

4. General ➤ Editors ➤ Text Editors
  - a. Set the “Displayed tab width” to 4
  - b. Turn **off** “Insert spaces for tabs”
  - c. Turn **on** “Show line numbers”



## 5. ProvideX

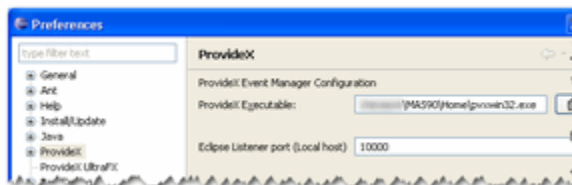
- a. For “ProvideX Executable”, select “pvxwin32.exe” from a MKS sandbox or MAS 90 installation.



Make sure that:

- A developer activation key is found in the “Lib\Keys” folder under the path where the ProvideX executable is located.
- The NOMADS designer is installed in the same location.

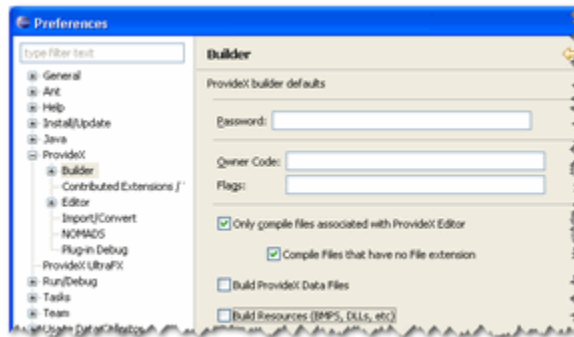
- b. If you have no other software installed on your system that is using port 10000, you can leave the “Eclipse Listener port” set to 10000. Otherwise, change this number to reference an available port.





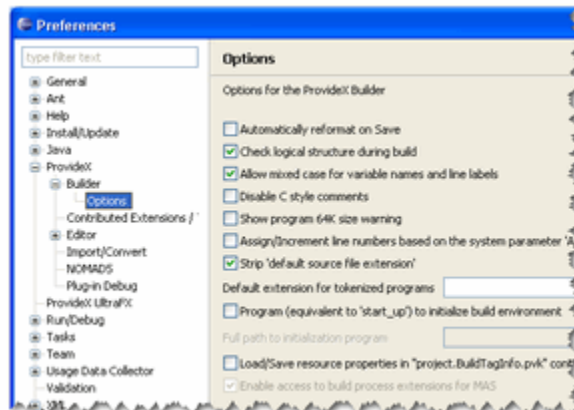
## 6. ProvideX ➤ Builder

- Select “Only compile files associated with ProvideX Editor”
- Select “Compile Files that have no File extension”
- If this workspace is to be associated with an existing MAS installation, clear all other options



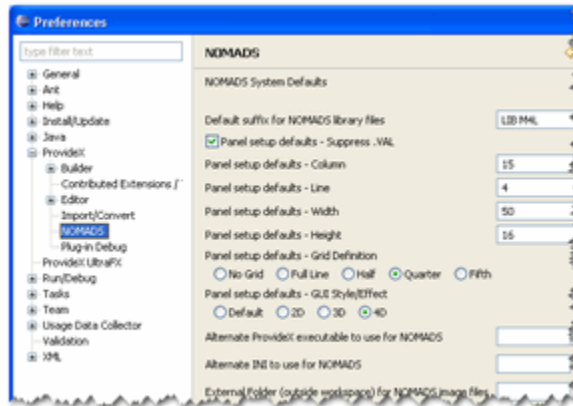
## 7. ProvideX ➤ Builder ➤ Options

- Select “Check logical structure during build”
- Select “Allow mixed case for variable names and line labels”
- Leave the remaining options at the default settings.



8. ProvideX ➤ **NOMADS**

- a. Set “Default suffix for Nomads library files”
  - i. If this workspace is to be used for developing or maintaining MAS applications, set the default prefix to “**LIB M4L**” (enter a space between LIB and M4L).
  - ii. Otherwise, set to the extension(s) used by the application to be maintained in this workspace.
- b. Select the “**Quarter**” and “**4D**” options.



9. Click **[OK]** to save the changes to the preferences and close the window.

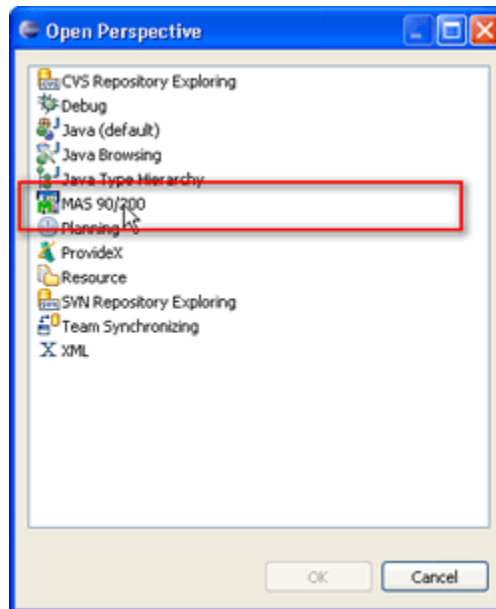
# Using Eclipse – Sage 100 ERP Projects and MKS

This section introduces some of the more important aspects of the Eclipse development environment for a Sage 100 ERP developer.

## Sage 100 ERP Perspective

An Eclipse “perspective” defines the set of views and their layout in the Workbench window. The Sage 100 ERP perspective is aimed at configuring the workspace environment to best suit Sage 100 ERP development needs.

1. From the Eclipse menu, choose Window > Open Perspective > Other...
2. Select *Sage 100 ERP* from the list and then click [OK]



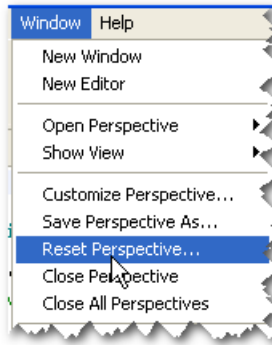
3. The MAS perspective consists of the following views:

<b>Navigator</b>	<i>Top-Left</i>	Tree view of projects and their members
<b>Outline</b>	<i>Bottom-Left</i>	Components of program selected in the Navigator view or loaded in the Editor depending on which has focus
<b>Editor</b>	<i>Top-Right</i>	Program editor
<i>Grouped Views</i>	<i>Bottom-Right</i>	Problems, Console, Properties, Tasks, NOMADS, etc.

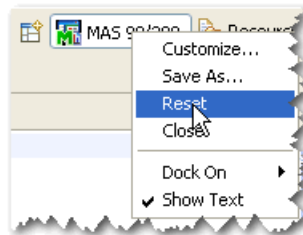
4. Each view in the perspective can be resized, maximized, hidden, detached, and moved to another location.

a. To restore the perspective to its default settings:

i. Click Window ➤ Reset Perspective...



ii. Right-click on the perspective name at the top-right of the Workbench window and click Reset.



5. If the Eclipse Welcome view is still visible when selecting the MAS perspective, close it by clicking its “X” button.



You can restore the Welcome view by choosing Help ➤ Welcome

## Create Sage 100 ERP Project

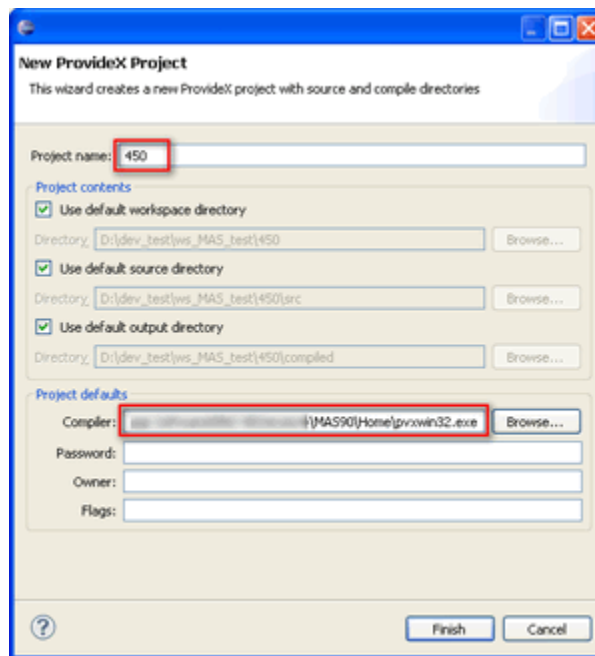
Eclipse uses a project metaphor for managing source files and compiled builds. For Sage 100 ERP development, an Eclipse project equates to an MKS sandbox.

1. Choose **File** > **New** > **ProvideX Project**
2. Enter a project name.



For consistency, the suggested naming standard is to use simple numbers that equate to the Sage 100 ERP level, such as 430 or 440 or 500. When creating sandboxes for service level releases, use dots to segregate the main level from the service level, such as 430.1 or 440.2.

3. Leave the workspace, source, and output directories set to their default values.
4. For “Compiler”, specify the appropriate PvxWin32.Exe for the Sage 100 ERP release.
5. The “Password”, “Owner”, and “Flags” fields must remain blank.

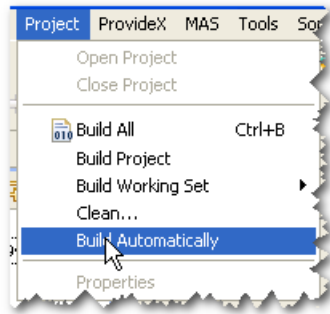


6. Click **[Finish]** to create the project. It will then be visible in the upper-left Navigator view.

## Disable Automatic Builds

Whenever a source file is created or modified, Eclipse will automatically force a build of the source file. Before assigning a MKS sandbox to the MAS project, the automatic build feature must be disabled.

1. Click on **Project** ➤ **Build Automantically** to remove the checkmark. This will disable the automatic builds.



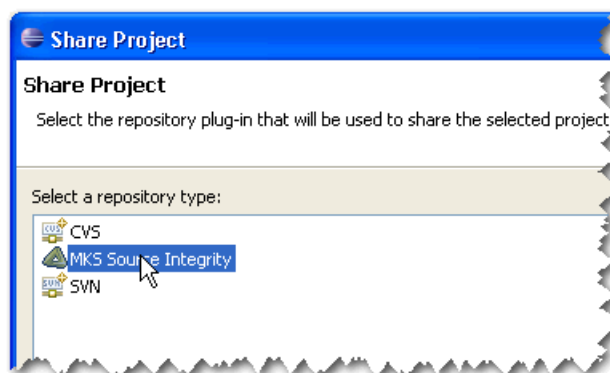
## Assign MKS Sandbox to Project

Follow these steps to associate the project with a MKS sandbox.

1. Right-click on the ProvideX project and select **Team** ➤ **Share Project...**

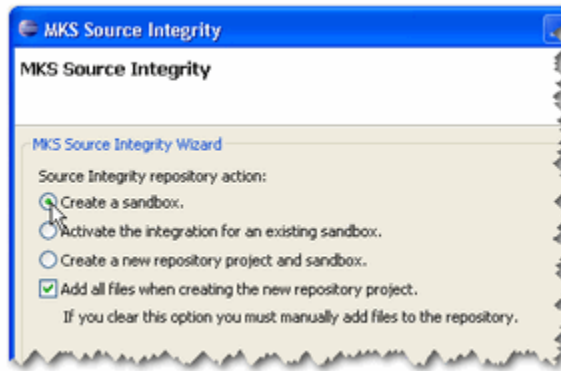


2. On the Share Project window, select **"MKS Source Integrity"** and click **[Next]**.




If MKS Source Integrity does not appear in the list, or Eclipse jumps straight into CVS or Subversion setup, then the MKS Plug-in is not installed properly. Refer back to the section 'Version Control – MKS'.


3. Select “**Create a sandbox**” and click [Finish].




- a. MKS connects to the Source Integrity server and starts up the Create Sandbox Wizard.
  - b. You may be prompted for your MKS user name and password. To configure the MKS client to connect automatically, see Appendix B.
4. In the Create Sandbox Wizard window click [Select...].
  5. Select a **mas90.pj** project.



 You must select the top-level **mas90.pj** project or the MKS Eclipse integration will not work.

 If you need a sandbox based on a single module or subset of modules, you must create the sandbox from **mas90.pj** and then delete any unwanted projects.

6. Click [OK] to close the window.
7. Click [Next].
8. The Sandbox Location defaults to the project’s workspace folder.

 Do **not** change the Sandbox Location or the MKS Eclipse integration will not work.

9. Click [**Finish**] to create the sandbox.
10. You will be prompted to recurse into subprojects. To conserve disk space and speed up the sandbox build process, you may wish to do this selectively, skipping demo data and other projects that you do not need in your sandbox.



You cannot resynchronize files from within the MKS Eclipse integration. If you do not synchronize files when creating the sandbox, you must use the MKS Integrity Client to synchronize files.

11. After the build process finishes, to conserve space and to make the project easier to view and manage, expand the project in the Navigator view, then right-click on a subproject and select [**Delete**]. Some subprojects that consume a lot of space, yet are not needed by many developers include:

- Demo data – MAS\_ABC, MAS\_ABX, MAS\_EED, MAS\_FDD
- MAS\_System
- Reports
- Dictionaries
- Help
- Launcher
- Readme
- Tutorials

## ***Clean Build***

When you save changes to a program under a MAS “src” project, the ProvideX builder compiles and saves it into the “compiled” folder. NOMADS libraries and other non-program files are copied into the compiled folder when saved. After assigning the MKS sandbox resources to the MAS project, you must do a clean build.

1. From the Eclipse menu, select **Project** ➤ **Clean...**
2. Select “Clean projects selected below”.
3. Select the project you want to build and then press [**OK**].



If the MKS HOME subproject was included in the project, you will be prompted for passwords for various ProvideX programs during the build process. There are also a few SOA programs that use passwords. You will need to press [**Cancel**] when prompted for a password, unless you know the correct password.

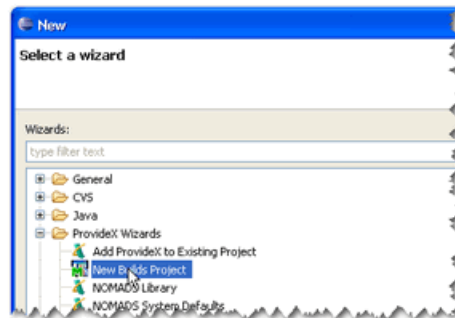
4. After the clean build finishes, turn automatic builds back on, click **Project** ➤ **Build Automatically**



## Link Project to SAGE 100 ERP Build

With the ProvideX Eclipse plug-in, both programs and Nomads libraries can be copied into a Sage 100 ERP build area during the build process.

1. Choose **File** ➤ **New** ➤ **Other**
2. Under ProvideX Wizards, select **New Builds Project** and then click **[Next]**.



3. For “MAS Project”, click **[Browse...]** and select a MAS project name.
4. For “Builds Project name”, enter “**Builds**”.
5. Under “Directory”, click **[Browse...]** and select the **MAS90** folder of a build.
6. Click **[Finish]** to create the builds project and link the build process to the MAS project so that it updates the specified build directories.
7. In the Navigator view, you should now see the “Builds” project.
8. Expand the “Builds” node and you’ll find a “MAS90” linked folder.



In the Navigator view, a linked folder provides an explorer view of a file system. You can traverse the folder hierarchy and open/rename/delete files from this view.

### Edit Build Links

1. To edit the build links for a project, right-click on the “**compiled**” folder under the project, and select **Properties**.
2. Select “ProvideX” and then use the **[Add]**, **[Search]** and **[Remove]** buttons to modify the MAS 90 builds that get updated when changes are made to the project.

## **MKS Plug-in Features and Limitations**

Many Integrity Client functions are not available from within Eclipse.

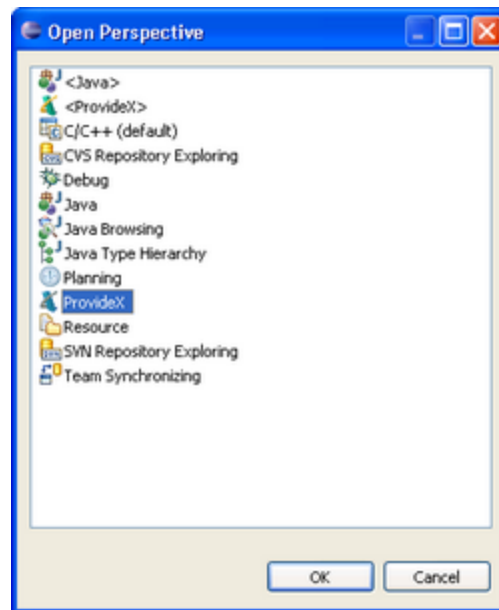
For example, you can check members out and back in, but you cannot unlock a member that you have checked out. To do this, select “Launch Source Integrity” from the Team menu and maintain the member using the Integrity Client.

## Eclipse – ProvideX and Subversion

Once the Eclipse distribution has been installed on the desktop system, the next step is to retrieve the project files from the Subversion repository.

### *Set Perspective*

From the Eclipse menu, choose Window ➤ Open Perspective ➤ Other...



Select the 'ProvideX' perspective the list of available options. This will open the collection of views that have been defined for the ProvideX perspective.

Once the perspective has been selected, a button will be placed in the upper right corner of the Eclipse workbench to indicate the current perspective and to allow easy switching between various perspectives.



## Subversion – Connection Protocols

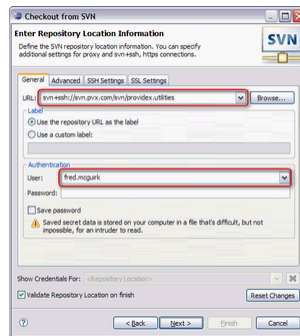
There are a number of methods that can be used to initiate a connection to a Subversion repository. Before discussing how to connect to a Subversion repository, it is necessary to

Subversion uses a URL to identify the location of the repository. There are several different protocols that can be used to communicate with a Subversion server. The protocol that is used for a specific Subversion server is determined by the configuration of that server.

### Specify Repository Location – SVN+SSH

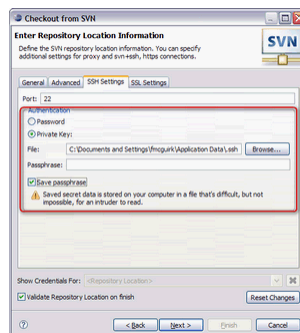
This protocol uses an SSH tunnel for encrypting the communication between the client and the server. It can also use a SSH key file to authenticate the user to the Subversion server instead of using passwords.

The format of the URL will be similar to “*svn+ssh://svn.server.com/svn/RepositoryName*” where “*RepositoryName*” is replaced by a published repository location for the server.



Select the ‘SSH Settings’ tab to continue the configuration of the user authentication to be used for accessing the Subversion repository.

Select the ‘Private Key’ option and then browse to and select the SSH private key file that was used to authenticate the user to the CVS server.

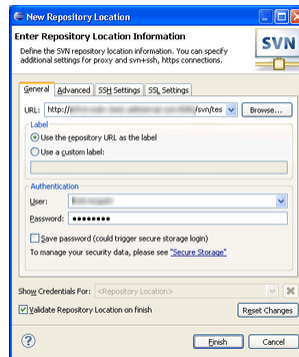


Do not enter a password, and check the box to ‘Save Password’ so that you are not prompted for a password for each connection to the repository.

## Specify Repository Location – HTTP

This protocol makes Subversion repositories available to clients via the WebDAV/DeltaV protocol, which is an extension to HTTP 1.1 (see <http://www.webdav.org/> for more information). This protocol takes the ubiquitous HTTP protocol, and adds writing—specifically, versioned writing—capabilities.

The format of the URL will be similar to “<http://svn.server.com/svn/RepositoryName>” where “*RepositoryName*” is replaced by a published repository location for the server.



## User Authentication

The web server that is hosting the Subversion server can be configured to use Active Directory (or LDAP) for user authentication which will make integration into an existing corporate network much easier.

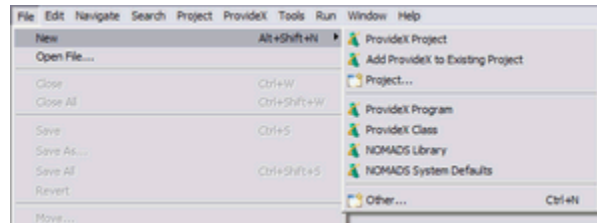
Enter the user name and password to be used to authenticate with the server.

Click **[Finish]** to complete the setup.

## Accessing a Subversion Repository – New Project Wizard

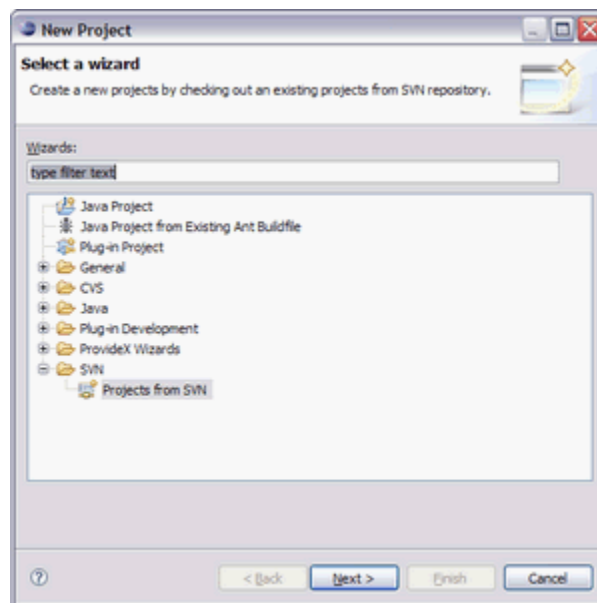
### Create a New Project – from SVN

All work that is performed in Eclipse is organized into projects which are a collection of resources (files and folders). A new project can be created by selecting the 'File' > 'New' menu option (or right click in the 'Navigator' view and choose 'New' from the pop-up menu).



The menu lists a number of choices for creating a new project; in this case, select the 'Project...' option.

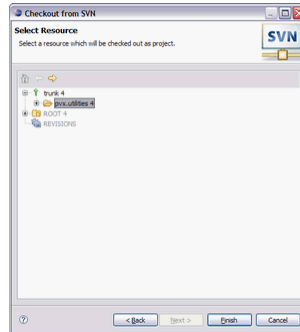
The 'New Project' dialogue window is displayed containing a list of project wizards; select 'SVN' > 'Projects from SVN' from the list.



Configure the connection to the Subversion repository using one of the protocols discussed earlier in this document.

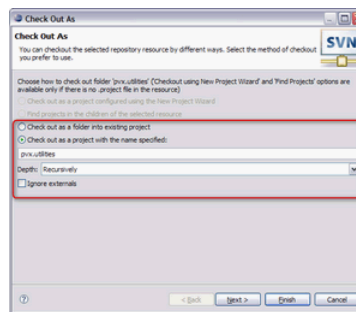
## Checkout Project

A list of the resources in the repository will be displayed. To get the development version of a project, expand the 'trunk' item and select the project from the list.



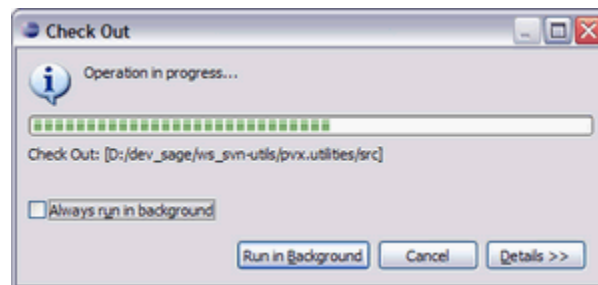
Click **[Finish]** to complete the check-out of the project.

Next, enter the name to give this project in the current workspace; the default will use the same name as the project in the repository.



Click **[Finish]** to continue with the check-out of the “trunk” or development version of the project.

A progress bar will be displayed during the check-out process.



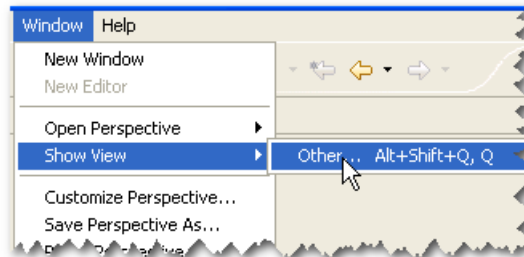
Once the project files have been transferred to your system, an automatic build of the files will be performed.

## Accessing a Subversion Repository – Browse SVN Repository

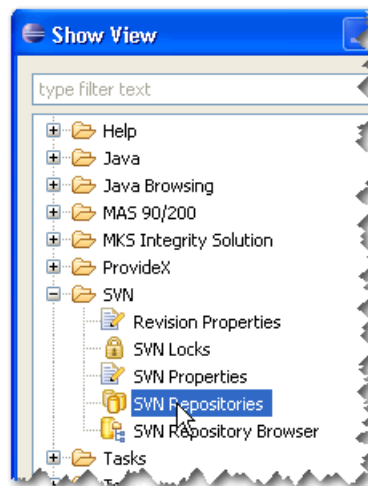
### SVN Repository View

An alternative method for viewing the contents of a Subversion repository is to use the SVN Repository view:

1. From the Eclipse menu, choose **Window** > **Show View** > **Other...**



2. Select **SVN** > **SVN Repositories** and then press **[OK]**.



3. The view will open on the bottom-left of the window.
4. Right click in this view and select **New** > **Repository Location...**



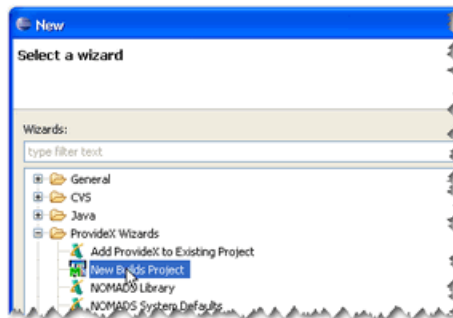
5. Configure the connection to the Subversion repository using one of the protocols discussed earlier in this document.
6. Once the repository location has been defined, you will be able to check-out projects, view history, show properties, etc.



## Link Project to Sage 100 ERP Build

With the ProvideX Eclipse plug-in, both programs and Nomads libraries can be copied into a Sage 100 ERP build area during the build process.

9. Choose **File** ➤ **New** ➤ **Other**
10. Under ProvideX Wizards, select **New Builds Project** and then click **[Next]**.



11. For “MAS Project”, click **[Browse...]** and select a MAS project name.
12. For “Builds Project name”, enter “**Builds**”.
13. Under “Directory”, click **[Browse...]** and select the **MAS90** folder of a build.
14. Click **[Finish]** to create the builds project and link the build process to the MAS project so that it updates the specified build directories.
15. In the Navigator view, you should now see the “Builds” project.
16. Expand the “Builds” node and you’ll find a “MAS90” linked folder.



In the Navigator view, a linked folder provides an explorer view of a file system. You can traverse the folder hierarchy and open/rename/delete files from this view.

### Edit Build Links

3. To edit the build links for a project, right-click on the “**compiled**” folder under the project, and select **Properties**.
4. Select “ProvideX” and then use the **[Add]**, **[Search]** and **[Remove]** buttons to modify the MAS 90 builds that get updated when changes are made to the project.

## Daily Workflow

The workflow will be very similar regardless of the version control system that is used. The examples in the following sections are using Subversion.

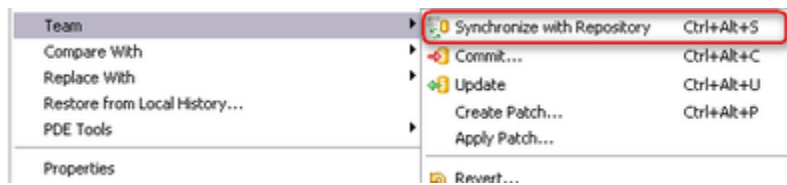
### *Synchronize with Repository*

Before making any modifications to files or folders within a project, it is good practice to synchronize your workspace with the repository. This will ensure that you are working with the most recent version of the files and will also reduce the chance of encountering a conflict when attempting to check-in your modifications.

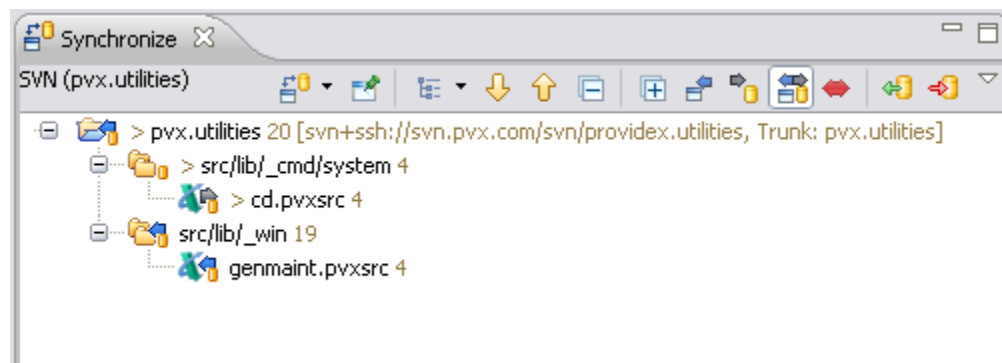
This option can also be used to check-in or commit your modifications to a project as a group of files rather than as individual modifications.

#### Synchronize

Select the project(s) to be synchronized with the repository, press the right mouse button, select 'Team' from the pop-up menu, and then select 'Synchronize with Repository' from the submenu.



This option will compare the files in your workspace against those in the repository and list the files where there are modifications. The "Synchronize View" will open with a list of the files and folders that have changed between the local workspace and the repository.

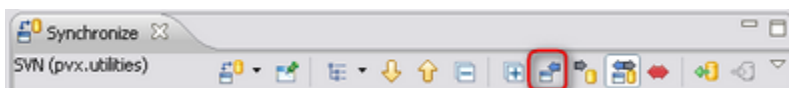


The status bar will show a summary of the differences that were found. These are classified in three categories: new version in repository (blue arrow pointing to the left), new version in local workspace (grey arrow pointing to the right), changes in the repository as well as local copy (red arrows pointing in both directions).

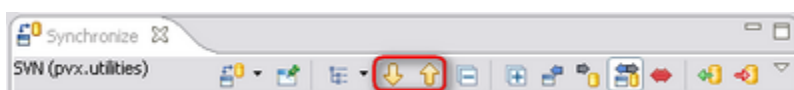


## Review Modifications - All

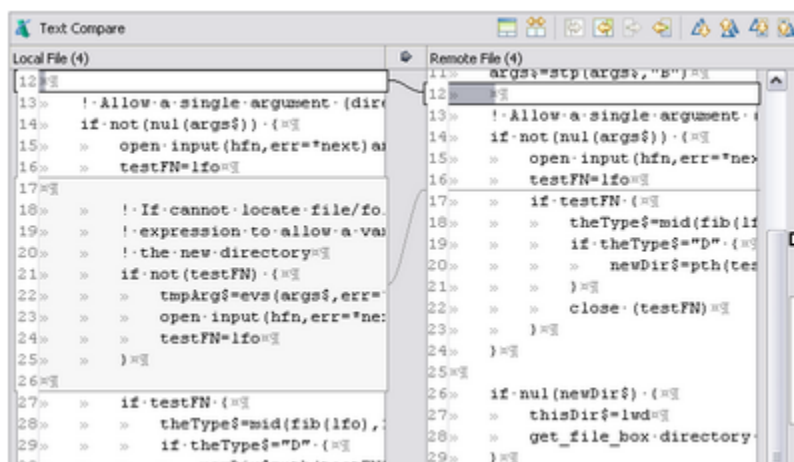
You can review the modifications that have been made before applying the changes to your local workspace. First, restrict the view to only the files/folders that have been changed in the repository and not the local workspace by pressing the toolbar button with the blue arrow that points to the left.



Use the up/down arrow buttons on the toolbar to review the modifications that have been made to the files.



Pressing these toolbar buttons will navigate through each of the modified files and show a comparison of the file in the repository and your local workspace copy. Each modification will be highlighted in sequence in a “Compare Editor” view.



## Review Modifications - Selected

You can also review the modifications for selected files by simply double clicking on the file in the “Synchronize” view or pressing the right mouse button and selecting “Open in Compare Editor” from the pop-up menu.

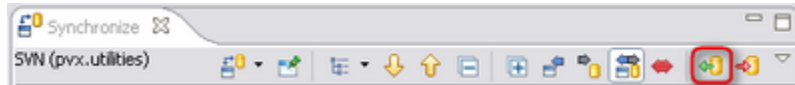
The “Compare Editor” toolbar can be used to merge selected modifications into your workspace as well as navigating through all of the modifications on the selected file.



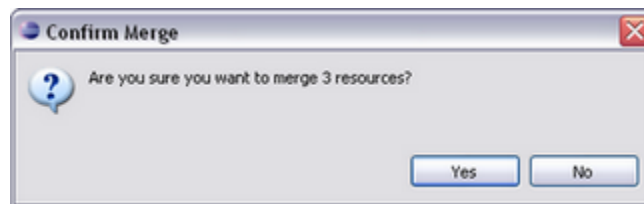
## Update – All

All updated files/folders in the repository should be updated to your local workspace before attempting making modifications to any of them. Otherwise, you will be required to reconcile the conflicts when the new modifications are checked-in.

To update all files in your workspace that have been changed in the repository where there are no conflicts (local changes), press the identified button on the toolbar (repository with arrow pointing to the right).



You will be prompted to confirm the update.



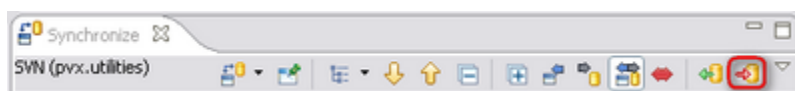
Once the update has been completed, the “Synchronize” view will be updated with the list of remaining files/folders.

## Commit Modifications

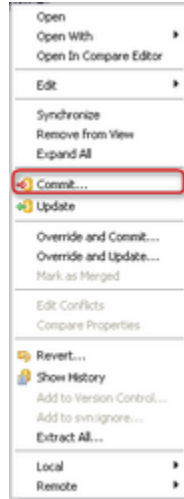
The modifications that have been made in your local workspace must be checked-in before anyone else can use them. It is best practice to only check-in modifications that are in a complete and workable state.

Therefore, you will need to decide whether all of the modifications can be checked in as a single change-set or if selected groups of modifications should be checked in. Subversion will track the modifications by change-set; when a group of files are checked in at the same time, they will be considered to be a combined set of modifications that must be made at the same time – a change-set.

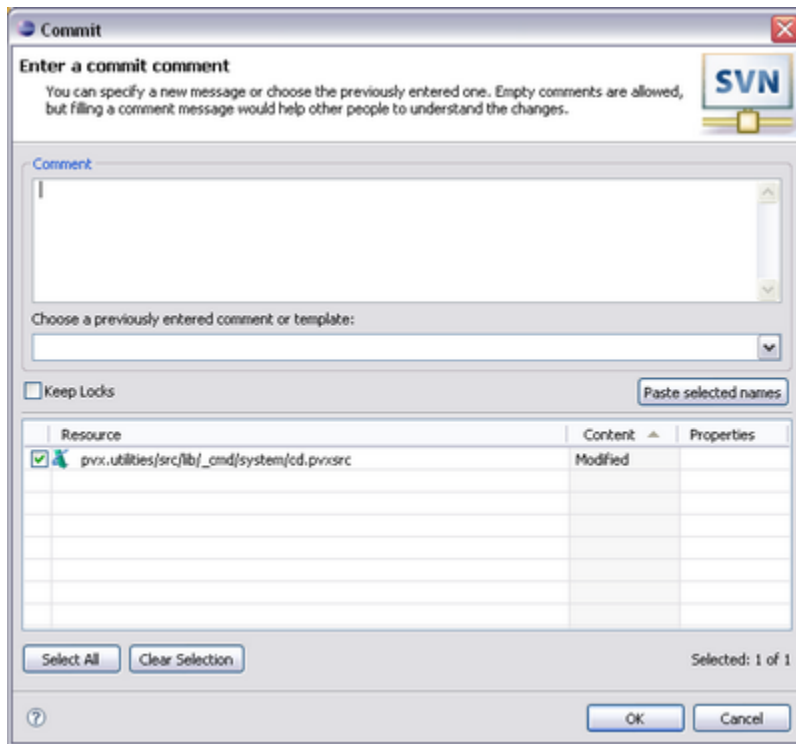
All modifications can be checked in together by using the “Synchronize” view toolbar button with the right arrow pointing towards the repository.



Alternatively, you can select the specific files to check-in, press the right mouse button and choose “Commit...” from the pop-up menu.



Once you have decided on the programs to be checked in, a dialogue will display asking for a comment to describe the modifications.



# Appendix

## **Appendix A - Windows Subversion Clients**

### ***Visual Studio Subversion Client***

AnkhSVN is a Subversion Source Control Provider for Microsoft Visual Studio 2005, 2008 and 2010.

AnkhSVN provides source code management support to all project types supported by Visual Studio and allows you to perform the most common version control operations directly from inside the Microsoft Visual Studio IDE.

Visit their web site <http://ankhsvn.open.collab.net/> for additional information and to locate a download file for your system.

### ***Windows Subversion Client***

TortoiseSVN is an easy to use Subversion client that is implemented as a windows shell extension. This client is only necessary for those who will not be using Eclipse to access all projects on the Subversion repository.

The most recent version can be found on their web site at <http://tortoisesvn.net/downloads>.

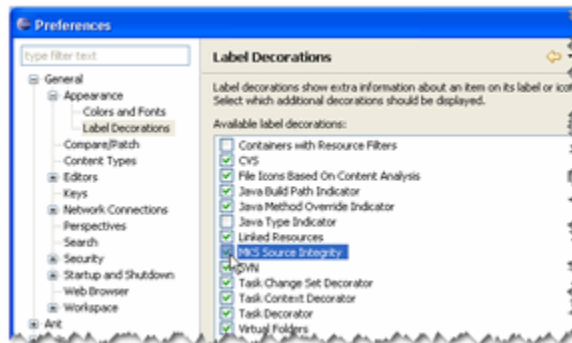
## Appendix B – Troubleshooting

### *Resources do not show Version Control information*

Once the version control plug-in has been installed, all projects and resources listed in the Navigator view should include information from the version control repository for each resource. The information that is displayed will vary depending on the repository and plug-in that is used.

To enable this information:

1. From the Eclipse Menu, choose Window ➤ Preferences.
2. General ➤ Appearance ➤ **Label Decorations**
  - a. Select the item for the version control system; the image below shows “MKS Source Integrity”.



### *ProvideX Builder does not create Program Files*

Occasionally, the ProvideX builder will indicate that it has successfully created the program file, but it cannot be loaded.

There are several ways that this can happen:

- There is a “src” folder in the Eclipse project, but the folder is named “Src” in the file system.
- A file exists in the output folder with the same name but a different mix of upper / lower case characters. Eclipse is case sensitive, even on a Windows system. This means that a file with the name “MyFile.pvx” is not the same as a file with the name “myFile.pvx” because the case is different.
- The output file name of the resource may not be set to the correct value.



## ***How to copy files to existing application folder***

The MAS/ProvideX plug-in can automatically copy files from the project output folder into an application folder that is outside of the Eclipse workspace and not managed by Eclipse.

A “link” must be created in the current workspace to the application folder. This link should be created in a *dummy* project in the current workspace and then create the linked folders in this project.

While there are other methods for linking folders within Eclipse, the steps below illustrate the preferred approach.

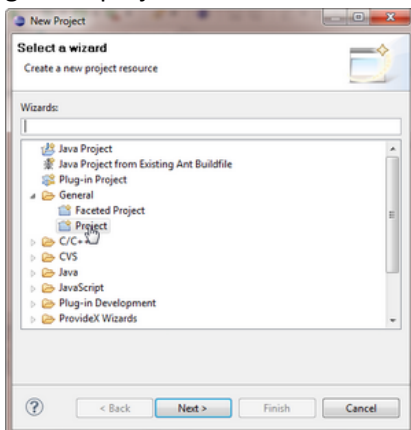


Eclipse is case sensitive – be consistent with the names that are used for any files created within Eclipse projects.

### **Create Project**

If you do not already have a “*dummy*” project in the current workspace, the following steps describe how to create one.

1. Select the menu item **New** ➤ **Project...** to start the wizard.
2. Expand the **General** category and select **Project** from the list and press [**Next>**] to create a generic project for the linked folders.



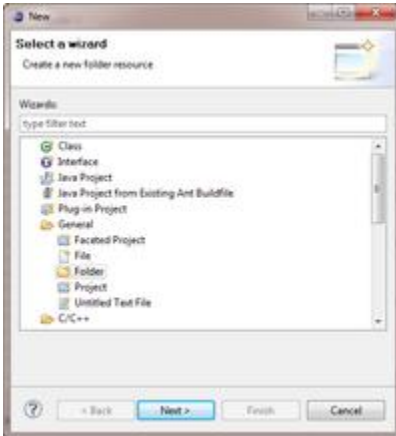
3. Enter a unique name, such as *zzLinkedFolder*, that will be easy to identify.
4. Press [**Finish**] to complete the creation of the project.

## Create a Linked Folder

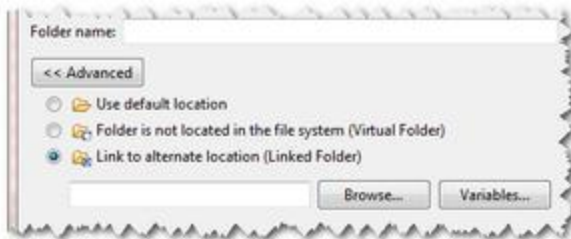


Once the linked folder has been created, it cannot be changed to reference a different application folder. A new linked folder must be created for each application folder to be referenced.

1. Right click on the *dummy* project and choose **New** ➤ **Other...** from the pop-up menu.
2. Now choose the **General** ➤ **Folder** wizard and press [**Next >**].



3. Enter a unique name for the folder that is derived from the application and version to make it easy to identify later, such as *myTestApp*.
4. Press [**Advanced>>**] and select “Link to alternate location (Linked Folder)”.



5. Press [**Browse...**] and select the application folder to be linked to the Eclipse project.



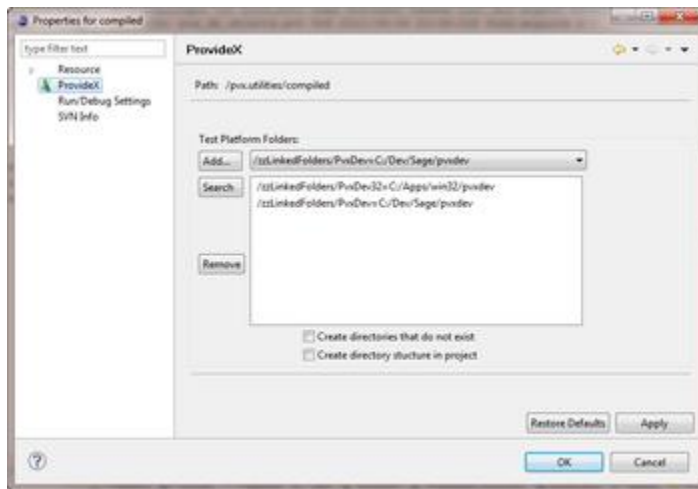
Be careful when selecting an application folder - the folder structure of the Eclipse project will be recreated in the selected application folder.

6. Press [**Finish**] to complete the creation of the linked folder.

## Link to Project Folder

Now that the “linked folder” has been created, it must be associated with the project that has the application programs/files that need to be copied during the MAS/ProvideX Plug-in build process.

1. Expand the project to be associated with the linked folder.
2. Right-click on the output folder (usually named “compiled”) and choose “Properties”.
3. Select the “ProvideX” properties page.



4. Press [**Search**] and enter the name of the linked folder in the filter field - do not press ENTER since the filter begins to work as you type.
5. Choose the linked folder that was just defined (such as *myTestApp*) from the list of matching resources and press [**OK**].
6. The selected resource will be added to the test platforms. Repeat step 4 for each application to be associated with this project.
7. Press [**OK**] to complete the process.

## ***Program is created, but not updated in a Linked Folder***

An updated copy of a program is not copied into a “Linked” folder when a build is performed.

In this case, there is probably a file in the linked folder that has the same name but uses different case for the name than the file created by the build process. Eclipse is case sensitive.