ptc

integrity™ lifecycle manager

Integration with Microsoft Visual Studio

11.1

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Contents

1

About This Guide

PTC® provides a number of integrations for Integrated Development Environments (IDEs). IDE integrations allow you to access the workflow and configuration management functionality of Integrity Lifecycle Manager while working within your favorite development environment.

The Integrity Lifecycle Manager integration with Microsoft® Visual Studio® allows users to access Integrity Lifecycle Manager commands through Visual Studio. The integration provides a seamless development and configuration management experience.

This guide consists of the following sections:

Note

In Integrity 10.0, the default installation directory of the client changed. This change affects integrations that were installed with Integrity Client 10.4 or earlier. For more information, see the *Integrity Lifecycle Manager Help Center*.

2

Before You Start

Before you set up or use the integration, note the following:

- This guide assumes that you know how to use Microsoft Visual Studio and Integrity Lifecycle Manager. For more information about using a product, refer to the appropriate documentation.
- For suggested best practices on using the Visual Studio integration, read Best Practices on page 59.
- For details on supported integration versions of Microsoft Windows, Microsoft Visual Studio, and Integrity Lifecycle Manager, see www.ptc.com/support/integrity.htm
- The Visual Studio integration is designed to use change packages for submitting source code changes to the Integrity Lifecycle Manager repository. Ensure that change packages are enabled on the Integrity Lifecycle Manager server.

  To use change packages, the administrator must also enable Integrity Lifecycle Manager for startup on the server (`mksis.startup.im=true`). If change packages are mandatory, Integrity Lifecycle Manager for workflows and documents functionality must be enabled. This means that the policy for `IntegrityManagerEnabled` must be set to true).

  For more information on setting properties and policies on the server, see *Integrity Lifecycle Manager Help Center*.
- The following Integrity Lifecycle Manager server configurations are supported per Visual Studio solution:
  - One server configured for workflows and documents, and for configuration management
  - One server configured for configuration management

- One server configured for configuration management and one server configured for workflows and documents

- Ensure that the client path is set correctly for the `PATH` environment variable (that is, the Integrity Lifecycle Manager client's `<installdir>/bin` path appears first). Failure to set the path correctly displays a `Package Load Failure` error in Visual Studio. In addition, do not disable packages in Visual Studio. Disabling packages prevents you from selecting Integrity Lifecycle Manager as the source control provider.
- If you are using an older version of the Visual Studio integration, disable the integration in Visual Studio and the Integrity Lifecycle Manager client before upgrading to the new integration. To enable the new integration, see Setting Up and Configuring the Integration on page 16.

  If you have a Visual Studio solution that was placed under Integrity Lifecycle Manager source control using the previous SCC-based Visual Studio integration, you can migrate it for use with this integration version. For more information, see Migrating a Visual Studio Solution from the MKS SCC Visual Studio Integration on page 45.
- If you are upgrading from MKS Integrity 2007 SP 4 or earlier, this release of the Visual Studio integration does not use the MKS Worktray and MKS Change Package view found in previous integration releases. To improve management of work in progress and assigned work, this release uses Integrity Lifecycle Manager Work In Progress and Integrity Lifecycle Manager Items views. To use the Items view, one of your servers must be configured for workflows and documents. For more information, see Managing Work In Progress on page 23.
- When using the Integrity Lifecycle Manager integration with Microsoft Visual Studio, ensure that the necessary Access Control List (ACL) permissions are granted to users. For example, users adding Visual Studio solutions to Integrity Lifecycle Manager source control require the following permissions:

  - `CreateProject`
  - `ModifyProjectAttribute`
  - `CreateSubproject`

  For more information on configuring ACLs for Integrity Lifecycle Manager, see *Integrity Lifecycle Manager Help Center*.
- If you are working in a single Integrity Lifecycle Manager server environment, disable prompts for server information and credentials in the client. This provides a more seamless experience with the Visual Studio integration.

  If you are working in a multi-server environment, disable prompting for credentials. However, enabling server prompting so that you can connect to

the appropriate Integrity Lifecycle Manager server. If you do not disable prompting, Visual Studio displays prompts for server information and credentials when you perform Integrity Lifecycle Manager commands. Failure to disable prompting for credentials can cause errors when attempting to share a Visual Studio solution. To disable prompting in the Integrity Lifecycle Manager client, see the *Integrity Lifecycle Manager Help Center*.

- To avoid potential focus problems when using Visual Studio, close the client window, but do not shut down the client.

3

Setting Up and Configuring the Integration

This section describes how to set up and configure the Microsoft Visual Studio integration for use with Integrity Lifecycle Manager.

Installing the Visual Studio Integration

Supported versions of Visual Studio that are currently installed are registered for use with Integrity Lifecycle Manager when you run the Visual Studio integration installer.

Note

If you install another supported version of Visual Studio after you install the integration, you can register the version of Visual Studio by doing the following:

1. From the **Control Panel** ▸ **Add or Remove Programs** list, select **Integrity Integration for Microsoft Visual Studio**.
2. Click the link for **Click here for support information**.
3. Click **Repair**.

The Visual Studio integration registers the version of Visual Studio you installed.

Before you install the integration, ensure the following:

- A supported version of Integrity Lifecycle Manager is installed. For more information, see Before You Start on page 9.
- Visual Studio is shut down. If Visual Studio is running during the installation, it must be restarted after the installation completes to take effect.
- If you are currently using the Visual Studio integration available in Integrity client 2007 SP5 or 10.4, disable the integration in the client.

  To remove the Visual Studio integration, you must run the Integrity Lifecycle Manager client as an administrator, and then disable the integration.

To install the Visual Studio integration:

1. Run `Integrity_VS_Integration.msi`. The **Integrity Integration for Microsoft Visual Studio Setup** wizard displays.
2. Click **Next**.
3. Choose a directory to install the integration in. By default, the integration is installed in `C:\Program Files\Integrity\Integrations\Visual Studio`.
4. Click **Next**.
5. To install the integration click **Install**.
6. To exit the wizard, click **Finish**.

Enabling the Integrity Lifecycle Manager Plug-In in Microsoft Visual Studio

Visual Studio supports various source control providers. You must specify Integrity Lifecycle Manager as the source control provider.

1. In Visual Studio, select **Tools** ▸ **Options**. The **Options** window opens.
2. Select **Source Control** ▸ **Plug-in Selection**.
3. From the **Current source control plug-in** list, select Integrity Lifecycle Manager.
4. Click **OK**. The Integrity Lifecycle Manager toolbar, Items, and Work In Progress views display. If desired, dock the views to a location of your choice in Visual Studio.

   For more information on the toolbar, see Working With Active Change Packages on page 21.

   For more information on using the Work In Progress view and Items view, see Managing Work in Progress on page 23.

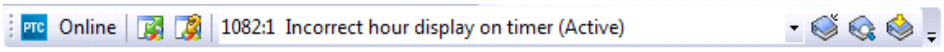
Showing and Hiding the Integrity Lifecycle Manager Toolbar

The toolbar displays connection status, a change package list, and toolbar buttons for managing changes to Visual Studio solutions and managing change packages. By default, the toolbar is shown when you select **Integrity** as the source control provider.

To hide and show the toolbar:

1. In Visual Studio, select **Tools** ▸ **Customize**. The **Customize** window opens.
2. On the **Toolbars** tab, select **Source Control - Integrity**. The toolbar is shown or hidden in Visual Studio.
3. Click **Close**.

   Once an open solution is under Integrity Lifecycle Manager source control, the toolbar buttons in the toolbar become active. For more information, see Sharing a Visual Studio Solution on page 32.

   For more information on connection status, see Working in Online Mode or Offline Mode on page 18.

   For more information on managing change packages, see Working With Active Change Packages on page 21.

For more information on managing changes to Visual Studio solutions, see Resynchronizing a Visual Studio Solution on page 49 and Reverting a Visual Studio Solution on page 49.

Setting Preferences

Once you enable the Integrity Lifecycle Manager plug-in, you can configure source control preferences. You can specify the default location for imported and branched Visual Studio solutions. This path displays when you import or branch a Visual Studio solution.

To set the default location:

1. Select **Tools** ▸ **Options**. The **Options** window opens.
2. Select **Source Control** ▸ **Settings**.
3. In the **Default Location for "Import Solution" and "Branch Solution"** field, type a path, or click **Browse** to select one. For example, the default location on Windows 7 is `C:\Users\<user name>\Documents`.
4. To save your preferences, click **OK**.

Working With Keywords

A keyword is a placeholder that can be inserted into text-based working files. This placeholder is a special variable (for example, `$Date: 2016/08/24 19:16:38IST $`, `$Author: Flett, David (dflett) $`, `$State: Exp $`) used to represent textual information in a working file. Keywords can be expanded (that is, replaced with their literal values) when a revision is checked out.

To enable keywords, set keyword preferences for the appropriate commands, such as check out and check in, in the Integrity Lifecycle Manager client preferences. For more information, see the *Integrity Lifecycle Manager Help Center*.

To use a keyword, simply include it in a working file, surrounded by dollar signs (for example, `$Date: 2014/06/09 11:23:07EDT $`) and check the file back into its archive.

Note

If you add keywords to solution and project files, Integrity Lifecycle Manager does not expand them on check in.

Configuring the Location of Visual Studio Solutions and Projects in the Integrity Lifecycle Manager Repository

Note

This section is intended for Integrity Lifecycle Manager administrators only. It is an optional feature related to sharing Visual Studio solutions and projects.

By default, sharing a Visual Studio solution allows you to choose where in the Integrity Lifecycle Manager repository you want to place a Visual Studio solution and Visual Studio project. In the Integrity Lifecycle Manager repository, they are placed as a top-level project and subproject. The Visual Studio project is placed in the Visual Studio solution's Integrity Lifecycle Manager project subtree. The solution and project are known as being "in-tree" in the repository structure.

As an alternative repository configuration, Integrity Lifecycle Manager administrators can set a policy that automatically defines for users where top-level Integrity Lifecycle Manager projects and subprojects are placed in the Integrity Lifecycle Manager repository. If set, users do not choose the location where top-level Integrity Lifecycle Manager projects and subprojects are placed when sharing a Visual Studio solution.

Note the following:

- This policy takes effect immediately. However, it only affects new Visual Studio solutions and projects. Existing shared solutions and projects are not affected.
- Once you add the policy lines, do not remove them. To enable or disable the policy, change the values of `integration.catalog.vssolution.enabled` and `integration.catalog.vsproject.enabled` to true or false.
- The `integration.catalog.vsproject` and `integration.catalog.vssolution` settings are not to point to the same top-level Integrity Lifecycle Manager project or subproject.
- You must specify configuration paths for top-level Integrity Lifecycle Manager projects and subprojects. Specifying project paths causes share operations to fail.

  You can view path information for a top-level Integrity Lifecycle Manager project subproject in the **Project Information** window.

To define where top-level projects and subprojects are placed in the Integrity Lifecycle Manager repository:

1. In the Administration Client, open the **Configuration Management** node and select **Policies**.
2. In the right pane, select **Global Policies**, and then select **Policies ▸ Edit**. The **Global Policies** window opens.
3. Click the **Other** tab.
4. To specify where top-level Integrity Lifecycle Manager projects for Visual Studio solutions are placed, add the following lines:
   ```
   integration.catalog.vssolution=<configuration path>
   integration.catalog.vssolution.enabled=<true/false>
   ```
   To specify where Integrity Lifecycle Manager subprojects for Visual Studio projects are placed, add the following lines:
   ```
   integration.catalog.vsproject=<configuration path>
   integration.catalog.vsproject.enabled=<true/false>
   ```
   where

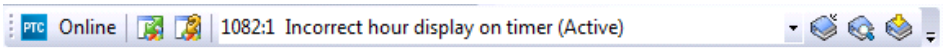
   `<configuration path>` specifies the configuration path of the top-level Integrity Lifecycle Manager project or subproject.

   For example, `#p=/abcFinancialToolkitApp/abcFinancialToolkitAppsln.pj`.

   `<true/false>` specifies if the policy is enabled or disabled.
5. To save your changes, click **OK**.

Working in Online Mode or Offline Mode

The Visual Studio integration uses connection status with the Integrity Lifecycle Manager client and an Integrity Lifecycle Manager server to determine the integration's operating mode. In Visual Studio, the Integrity Lifecycle Manager toolbar displays one of the following status indicators: **Connecting**, **Not Shared**, **Online**, or **Offline**.



When you open Visual Studio and no solution is open, the integration cannot determine the connection status, and displays **Not Shared** in the Integrity Lifecycle Manager toolbar. To establish a connection with the client and server, open a Visual Studio solution under Integrity Lifecycle Manager source control or place a solution under the source control. While attempting to establish a connection, Visual Studio displays **Connecting**.

When the integration connects to the Integrity Lifecycle Manager client and the client establishes a connection with the Integrity Lifecycle Manager server, the integration switches to **Online** mode. All Integrity Lifecycle Manager commands and views are available.

If the client cannot establish a connection with the server or the previously established connection is lost, the integration switches to **Offline** mode. In offline mode, you can continue working in Visual Studio. However, you can perform only the Integrity Lifecycle Manager commands that do not require a connection to the server, such as adding or editing files.

Any changes affecting Sandbox members are done without an active change package. This requires that you move the changes to an active change package when you switch to online mode. With the exception of the connection status indicator, all views and commands that affect the Integrity Lifecycle Manager repository are disabled. Most notably, you cannot rename files or directories. Additionally, the **Solution Explorer** only updates source control status icons for basic local operations, such as adding, deleting, and editing files. For more information, see Integrity Lifecycle Manager Source Control Status Icons in Visual Studio on page 20.

If you are working in online mode and the server suddenly goes down or there is a poor network connection, the integration automatically switches to **Offline** mode. This allows you to continue working in Visual Studio. When the server connection is re-established, the integration automatically switches to **Online** mode. All Integrity Lifecycle Manager commands and views automatically become active again, allowing you to resynchronize your changes with the Integrity Lifecycle Manager repository.

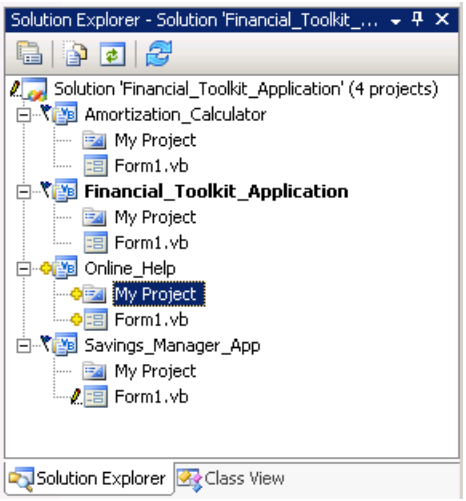
Note

Switching from offline to online mode refreshes the **Solution Explorer**, resynchronizing all resources in the Visual Studio solution and retrieving the latest source control status icons. Depending on how many files or Visual Studio projects are visible in the solution, this process can take a long time. However, it occurs in the background and does not prevent you from working in Visual Studio.

You can also manually switch the integration to offline mode. Working in offline mode is recommended when you know that you do not have access to the server for long periods of time. For example, you could be working remotely without an Internet connection. Or, you could intentionally disconnect your machine from the network by clicking **Online** and then **Go Offline**. In offline mode, you can edit and add local files. When you reconnect to the network by clicking **Offline** and then **Go Online**, you must move your offline changes to a change package for submission to the Integrity Lifecycle Manager repository.

Integrity Lifecycle Manager Source Control Status Icons in Visual Studio

The Visual Studio **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view are enhanced with Integrity Lifecycle Manager icons to indicate source control status. For example, a flag icon displays beside a Visual Studio solution or project if it is under Integrity Lifecycle Manager source control. A pencil icon displays beside a file that you have checked out and locked.



For files edited in Visual Studio, the integration automatically updates Integrity Lifecycle Manager source control status icons in the **Solution Explorer**.

The following Integrity Lifecycle Manager source control status icons display in the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view:

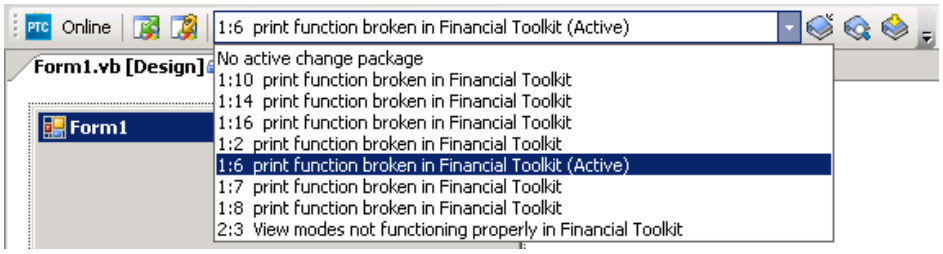
Icon	Description
	A Visual Studio Solution, Visual Studio project, or Web Application Project under Integrity Lifecycle Manager source control **Note** This icon does not display for websites under Integrity Lifecycle Manager source control.
	A locally modified file
	A locally added file
	A locally dropped file.
	A locally renamed or moved file
	A file that is a former member
	An internal error has occurred while retrieving the status
	An incoming change

Icon	Description
	A locally modified change that includes an incoming change
	A locally moved or renamed file that includes an incoming change
	A locally modified file that includes a lock by another user
	A file locked by another user
	A locally moved or renamed file that includes a lock by another user

Working With Active Change Packages

The Visual Studio integration is designed to use change packages for submitting source code changes to the Integrity Lifecycle Manager repository. The active change package is a change package that has been set as the default change package for member operations. This also allows other developers to identify what you are currently working on. They can see which files you have locked and the associated change package.

The Integrity Lifecycle Manager toolbar displays a change package list that allows you to select the active change package. It also displays buttons for creating, submitting, and viewing change packages. Creating a change package makes it the active change package. For more information of using the toolbar buttons, see Managing Work In Progress on page 23.



By default, the active change package label displays the change package ID, change package summary, and `(Active)`, for example, `1:6 print function broken in Financial Toolkit (Active)`. If no active change package exists, `No active change package` displays. Text that exceeds the size of the label is truncated. However, you can hover your mouse over the label. If the change package is the active change package, `Currently Active Change Package` displays in a tooltip. If you are connected to more than one server, the server and port that the change package resides on also appears.

To select a different active change package, click the drop-down arrow button next to the change package list and select a change package or `No active change package`.

Note the following about using change packages with the Visual Studio integration.

- A change package is a container for distinct tasks, not a generic container for all tasks performed during a project. A change package containing too many entries can affect the integration's performance. Before you perform a task that requires a change package, create a change package (this enables it as the active change package).
- An active change package cannot be used for multiple solutions.
- The active change package must be set in Visual Studio. If you set the active change package in the Integrity Lifecycle Manager client, the integration does not reflect the change.
- You can move added or modified change package entries from one change package to another. You cannot move renamed or moved change package entries from one change package to another.

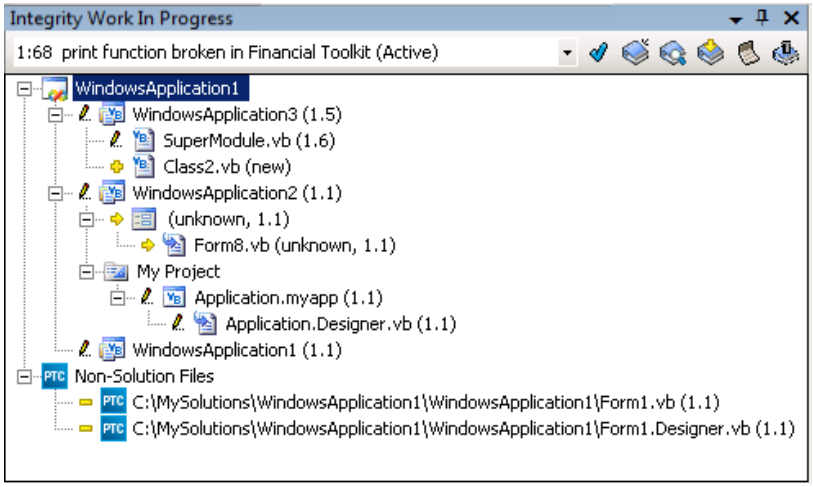
4

Managing Work in Progress

All Integrity Lifecycle Manager source control operations in the Visual Studio integration use change packages to track work. To manage your work in progress from the context of a change package, the Visual Studio integration includes an Integrity Lifecycle Manager Work In Progress view. This view displays your current change package as a collection of files, displayed in a Visual Studio solution hierarchy. Additionally, the Visual Studio integration includes an Integrity Lifecycle Manager Items view. This view display your assigned items.

Displaying the Work In Progress View

By default, the Integrity Lifecycle Manager Work In Progress view displays when you enable Integrity Lifecycle Manager as the source control provider. If the view is not displayed, select **View ▸ Other Windows ▸ Integrity Work In Progress**. The Work In Progress view displays.



From the Work In Progress view, you can do the following:

- View open change packages
- Submit change packages
- Move changes between change packages
- View work items not associated with change packages
- Revert changes
- Move unassociated changes into change packages
- Resolve conflicts with incoming changes
- Discard change packages

Note the following:

- All changed files are tracked with Integrity Lifecycle Manager locks. However, Visual Studio does not register changes unless there are content modifications to the files. This means files that have not changed do not appear in the Integrity Lifecycle Manager Work In Progress view. If you view the associated change packages, the files are locked. When you submit change packages, the integration re-examines the files for content modifications. Files with no changes are reverted (unlocked).
- The Work In Progress view dynamically updates according to changes in change packages.

The Work In Progress view displays your current change package as a collection of working files, displayed in a Visual Studio solution hierarchy. All nodes in the hierarchy display alphabetically. Multiple CP entries can be collapsed so that they are represented as a single working file.

Revision numbers for files display in brackets, for example, `Application.Designer.vb (1.1)`. You can view more detailed information about a file by using the **View Member Properties** command. For more information, see Advanced Integrity Lifecycle Manager Commands on page 56.

Note the following:

- After you branch a solution, the revision number that displays beside the solution node is the revision number of the solution file.
- Double-clicking a file in the Integrity Lifecycle Manager Work In Progress view opens it for editing.

File Status

The Integrity Lifecycle Manager Work In Progress view only displays the status represented by working files. This includes source control status icons for added, dropped, moved, renamed, and changed files. For more information, see Integrity Lifecycle Manager Source Control Status Icons in Visual Studio on page 20.

Incoming Changes

To notify you when one or more files need to be resynchronized, the Integrity Lifecycle Manager Work In Progress view displays source control status icons for conflicts (incoming and outgoing changes). This view also displays these icons for remote drops by other users. Resynchronizing puts the latest revision in your Sandbox. For more information, see Resynchronizing a Visual Studio Solution on page 49.

Managing Change Packages

The change package list allows you to view changes associated with a change package. From the toolbar, you can perform the following change package operations:

Command	Operation
Set Active Change Package 	Sets the selected change package as the active change package. (Active) appears after the change package name.
Create Change Package 	Creates a change package, making it the active change package. The **Create Change Package** window opens.
View Change Package 	Displays the change package for the currently

Command	Operation
	selected change package. If there is no selected change package selected, this command is disabled. The **Change Package** view displays.
Submit Change Package 	Submits the selected change package, committing the associated changes (files) to the Integrity Lifecycle Manager repository. By default, confirmation of a successful submission does not appear. To display a confirmation message, enable the **Show Successful Submit** option for the **Submit Change Package** command in the client.
View Integrity Item 	Displays the item associated with the selected change package. The **View Item Details** view displays.
Discard Change Package 	Discards the selected change package and its associated entries.

Managing Non-Solution Files

Files that are not in the open solution cannot be displayed in the solution hierarchy. Change package entries for non-solution files display as full working file paths in a flat list in the tree under the **Non-Solution Files** heading.

To commit the changes to the Integrity Lifecycle Manager repository, select the non-solution files and move them to a change package by right-clicking and selecting **Move to Change Package**. The **Move Work to Change Package** window opens. After you select a change package and click **OK**, you can submit the change package to commit the changes to the repository.

Managing Unassociated Changes

Unassociated changes are changes made outside of the context of a change package. They can be made while in offline mode. They can be made while **No active change package** is selected.

To view these changes, select **Unassociated Changes** from the top of the change package list in the Integrity Lifecycle Manager Work In Progress view. Once unassociated changes display, you can move them to a change package by right-clicking the files and selecting **Move to Change Package**. The **Move Work to Change Package** window opens. After you select a change package and click **OK**, you can submit the change package to commit the changes to the repository.

Note

Visual Studio automatically makes changes to files based on events in Visual Studio. An example event is changing the development server port number in Web application project files. Another example event is adding debug information to website configuration files. Assume that a Visual Studio project is shared and an associated change package is closed before Visual Studio makes changes. These changes appear under **Unassociated Changes** in the Integrity Lifecycle Manager Work In Progress view.

Removing Entries from a Change Package

You can have an edit or deferred add entry in a change package that you do not want to submit with the change package. You can remove the entry, making it an unassociated change.

Note the following:

- Drop, rename, and move entries cannot be moved out of a change package, or from one change package to another.
- In the Integrity Lifecycle Manager Work In Progress view, you cannot move non-solution files from the currently displayed active change package to another change package. To move non-solution files in an active change package, use the Integrity Lifecycle Manager client GUI.

With a change package displayed in the Integrity Lifecycle Manager Work In Progress view, right-click the entry you want to remove, and select **Move to Change Package**. The **Move Work to Change Package** window opens.

From the change package list, select **No active change package**, and click **OK**. The changes are unassociated with the change package and appear in the Integrity Lifecycle Manager Work In Progress view when you select **Unassociated Changes** from the list.

When your changes are complete, you can move the unassociated changes to a change package for submission to the Integrity Lifecycle Manager repository.

Using Shortcut Menu Commands

Right-clicking a file in the Integrity Lifecycle Manager Work In Progress view allows you to perform the following commands:

- **Move to Change Package** (only for edited or deferred add files)
- **Resynchronize** (only for incoming changes)
- **Revert** (only for edited files)

- **View Differences** (displays differences between the working file revision and current changes)
- **View Incoming Differences** (displays differences between the member revision and current working file; only for incoming changes)
- **View Annotated Revision**
- **View Member History**
- **View Member Properties**
- **Change Lock to Exclusive** or **Change Lock to Non-Exclusive**

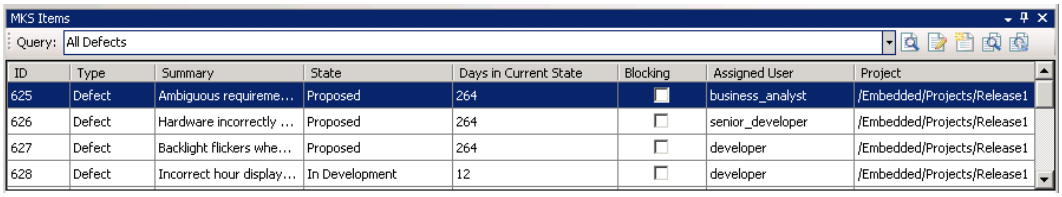
Displaying the Items View

The Integrity Lifecycle Manager Items view provides item functionality from within the Visual Studio interface, allowing you to manage your assigned items. This view replaces the Integrity Lifecycle Manager Worktray found in previous releases of the Visual Studio integration.

Note

If you are currently connected to your default Integrity Lifecycle Manager server, you are not prompted to establish a connection. If you are not connected to your default server, a prompt for choosing an existing connection can appear if you are connected to another server. Or, a prompt for creating a new connection can appear.

By default, the Integrity Lifecycle Manager Items view displays when you enable Integrity Lifecycle Manager as the source control provider. If the view is not displayed, select **View** ▸ **Other Windows** ▸ **Integrity Items**. The Integrity Lifecycle Manager Items view displays. By default, the Items view is empty. After you run a query, the Items view displays items returned by the query.



Queries

The Items view displays items based on the selected query. By default, no query is selected in the **Query** list. Query criteria must be defined and made visible from Integrity Lifecycle Manager before that query can be used from the Items view. The **Query** list is built when you display the Items view and displays the Quick Query by default.

Note the following:

- You can refresh the list by clicking the refresh query list button  .
- You can filter the **Query** list by typing the name of the query you are looking for. The query list displays the first matching query in the list and auto-completes.
- All queries visible to you display (favorites and non-favorites)

Toolbar

The following operations are available from the toolbar in the Integrity Lifecycle Manager Items view:

Tip

Right-clicking an item in the Items view allows you to perform the following commands: **View Item**, **Edit Item**, and **Create Change Package**. This assumes that a Visual Studio solution is open and under Integrity Lifecycle Manager source control.

Command	Operation
View Item 	Opens the selected item in the Item Details view.
Edit Item 	Opens the selected item for editing. **Note** Refreshing an item after an edit is successful only if the edit is invoked from the Integrity Lifecycle Manager Items view. Assume that you view the item and then edit it from the Item Details view. The item does not refresh in the Items view until you run the query again.
Create Item 	Creates an item.

Command	Operation
Refresh Query List 	Refreshes the **Query** list, updating the definitions of available queries.
Re-run Query 	Runs the selected query in the **Query** list. When you run a query for the first time, a table appears below the toolbar. The column names are fields defined by the query's column set. The table is sorted to match the sort direction and fields in the query definition. The default sort direction is the sort defined by the query definition. You can sort the tables by clicking the column headers. **Note** Rich content appears as plain text in fields that support rich content.

5

Placing Visual Studio Solutions Under Integrity Lifecycle Manager Source Control

This section provides information on placing Visual Studio solutions, projects, Web Application Projects, and websites under Integrity Lifecycle Manager source control. In addition, this section describes how to manage solutions under source control.

Sharing a Visual Studio Solution

Sharing is the process that associates a Visual Studio solution and its Visual Studio projects with an Integrity Lifecycle Manager project (the Visual Studio solution) and Integrity Lifecycle Manager subprojects (the Visual Studio projects). Sharing stores the relative layout of the solution and projects in a Sandbox, which does not have to be the layout in the Integrity Lifecycle Manager repository.

Once you place a Visual Studio solution and its projects under source control, you can perform Integrity Lifecycle Manager operations. For example, you can checking in files or view annotated revisions. To allow other users to work with the solution, they must import the solution. This creates a Sandbox of the solution under source control. For more information, see Importing a Visual Studio Solution on page 36.

Note the following:

- The Visual Studio integration supports Web Application Projects and websites on file systems. The integration does not support websites located via FTP and remote HTTP. HTTP links must be converted to local paths before placing the Web Application Project or website under Integrity Lifecycle Manager source control.
- Sharing Visual Studio projects within a website is not supported.
- To avoid potential issues with updating the Integrity Lifecycle Manager repository, always submit change packages from Visual Studio, not the Integrity Lifecycle Manager client.
- Your administrator can set policies that automatically define where Integrity Lifecycle Manager projects and subprojects for Visual Studio solutions and projects are placed on the Integrity Lifecycle Manager server. If these policies are defined, some of the steps in the following procedure do not necessarily appear.

To share a Visual Studio solution:

1. Do one of the following:
   - To share an existing Visual Studio solution, select the solution file in the `Solution Explorer` and select **File ▸ Source Control ▸ Share Solution**.
   - To share a new Visual Studio solution, select **File ▸ New ▸ Project**. The **New Project** window opens. Specify the project information and enable **Share Solution**, and then click **OK**.

   If you did not disable prompting for server information and credentials in the Integrity Lifecycle Manager client, a window opens. You are prompted to select a connection or click **New** to specify a new server to which to connect.

The **Share Solution** window opens.

2. Sharing a solution requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
3. In the **Share Name** and **Share Description** fields, provide a unique name and description for the solution. The name and description appear to users when they import the solution in Visual Studio. To proceed, click **Next**.

> **Tip**
>
> You can edit the descriptions of the Visual Studio project's shares, by clicking **Advanced**.

4. Specify where on the Integrity Lifecycle Manager server you want to add the solution.
   - In a new Integrity Lifecycle Manager project, do the following:
     a. Specify the root path, project path, and name of the top-level project.
     b. Click **Next**.
   - In a new Integrity Lifecycle Manager subproject within an existing Integrity Lifecycle Manager project, do the following:
     a. From the list, select the top-level project to which you want to add the new subproject.
     b. Click **Next**
     c. To filter the project list, type in the **Show project names containing** field to display top-level projects matching the name that you type.
     d. Specify the name and location of the new project to be created as a subproject.
     e. Click **Next**.
   - In an existing Integrity Lifecycle Manager project or subproject, do the following:

     > **Note**
     >
     > If the existing project contains identical files to the ones in the solution, the working files in the solution are used to update the files in the project.

     a. From the list, select the project to which you want to add the solution.
     b. Click **Next**.

c. To filter the project list, type in the **Show project names containing** field to display top-level projects matching the name you type.

5. To add Visual Studio solution files to the Integrity Lifecycle Manager repository, do the following:

   a. Choose one of the following options:

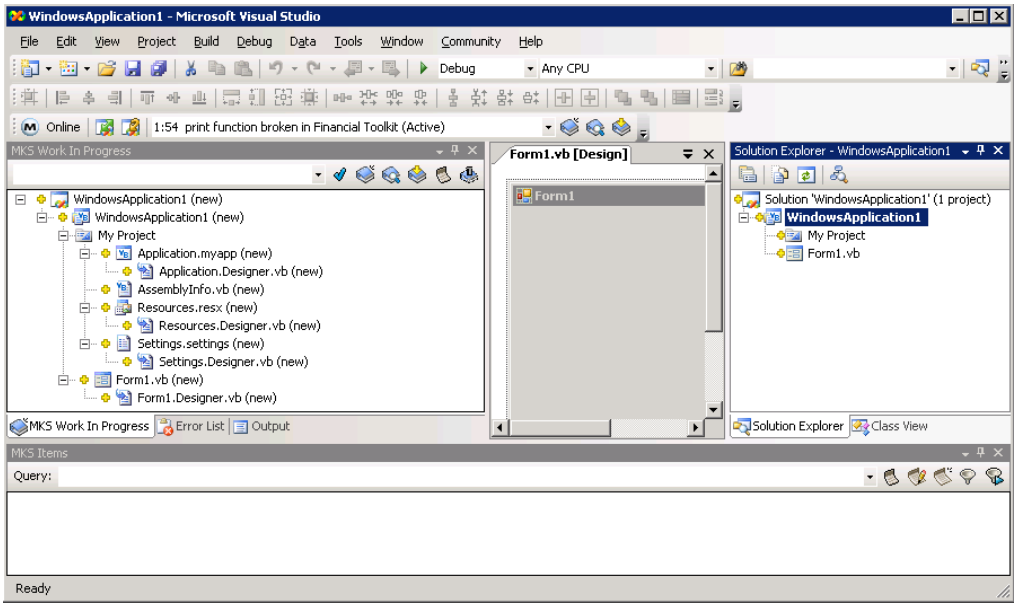
      - **Add all files in the Visual Studio solution to source control using the change package <CP>**. This option is the default.
      - **Add only the Visual Studio solution and Visual Studio project files to source control using the change package <CP>**. This option is recommended if you are sharing a Visual Studio solution containing a large number of files (over several hundred). After the Integrity Lifecycle Manager project and Sandbox are created, add the files to the solution in small batches using a separate change package for each batch.

   b. Click **Next**. A summary of your choices displays.

6. To share the solution, click **Share**. The integration informs you that the solution is made public (available for importing). This occurs when the change package associated with the share operation is submitted to the Integrity Lifecycle Manager repository.

7. Click **OK**.

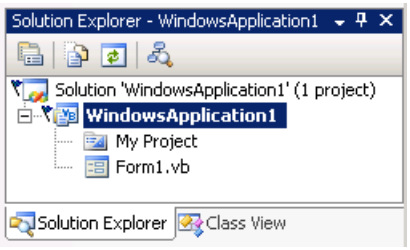
   In the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view, the solution, project, and its files display the added file icon  . This source control status icon indicates that they are deferred member adds. This means that they are files not yet committed to the Integrity Lifecycle Manager repository.



8. Submit the solution, project, and its files to the Integrity Lifecycle Manager repository by submitting the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view. The **Create Archive** window opens. Follow the procedure for creating archives, as described in the *Integrity Lifecycle Manager Help Center*.

> **Note**
>
> To avoid potential issues with updating the Integrity Lifecycle Manager repository, always submit change packages from Visual Studio, not the Integrity Lifecycle Manager client.

After the change package is submitted successfully, the change package and its entries disappear from the Integrity Lifecycle Manager Work In Progress view. In the **Solution Explorer**, the add member icon disappears from the solution, project, and files. The solution and project display the flag icon, indicating they are under source control.



Importing a Visual Studio Solution

To allow other users to use a solution under Integrity Lifecycle Manager source control, they must import the solution into Visual Studio. Importing a solution creates a Sandbox from the solution.

Note the following:

- Sharing a solution records the information necessary to import a solution. Thus, you must share a solution before you can import it, even if it is already in the repository.
- If the import fails, any Sandboxes created during the import remain on the file system.

To import a solution:

1. In Visual Studio, select **File** ▸ **Source Control** ▸ **Import Solution**. If you have a Visual Studio solution open that is under Integrity Lifecycle Manager source control, you are warned that importing a solution closes the currently open solution. Before you import a solution, Integrity Lifecycle Manager recommends submitting any solution changes to the Integrity Lifecycle Manager repository with an associated change package.

   To continue, click **Yes**. The **Import Solution** window opens.
2. From the list, select the solution you want to import. Then, click **Next**. To filter the results, type in the **Show share names containing** text field to display solutions matching the name you type.

> **Note**
>
> A branched solution displays the development path name in brackets after the solution name, for example, `FinancialToolkit(ServicePack1)`. For more information on branching a solution, see Branching a Visual Studio Solution on page 47.

3. Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.
4. To import the solution, click **Import**. The solution displays in the **Solution Explorer**. The solution and projects display the flag icon, indicating that they are under Integrity Lifecycle Manager source control.

Adding a Visual Studio Project to a Shared Solution

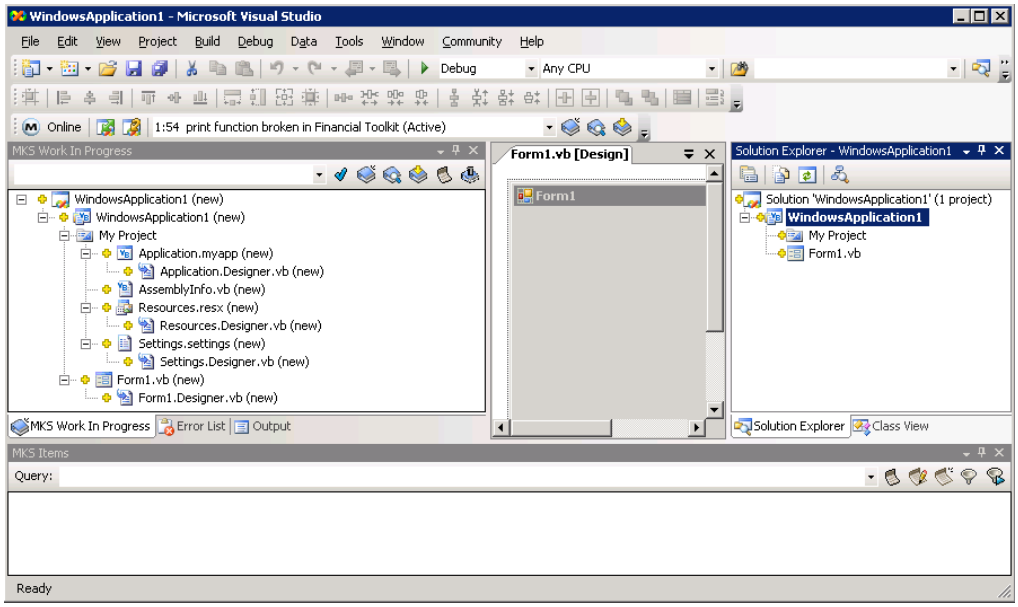
You can place a new or existing Visual Studio project under Integrity Lifecycle Manager source control by adding it to the shared Visual Studio solution it belongs to.

To add a Visual Studio project to a shared Visual Studio solution:

1. With a shared solution open, do one of the following:
   - To add an existing Visual Studio project to the solution, right-click the project in the **Solution Explorer** and select **Share Project**.
   - To create a new Visual Studio project and add it to the solution, right-click the solution and select **Add ▸ New Project**. Then, specify the project information and click **OK**.

   The **Share a Visual Studio Project** window opens. Sharing a project requires associating it with a change package.
2. Select a change package from the **Change Package** list, or create one by clicking **Create**.
3. To add the Visual Studio project to the solution, click **OK**. In the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view, the project and its files display the added file icon , indicating that they are deferred member adds.

Adding a project to a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, the pencil icon displays in the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view.

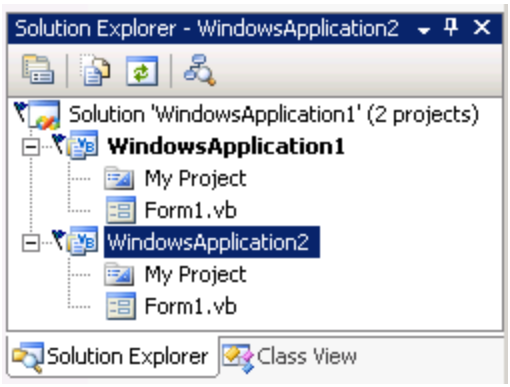


4. Submit the project and its files to the Integrity Lifecycle Manager repository by submitting the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view. The **Create Archive** window opens. Follow the procedure for creating archives, as described in the *Integrity Lifecycle Manager Help Center*.

   The **Check In** window opens for the solution file.

5. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*.

   After the change package is submitted successfully, the change package and its entries disappear from the Work In Progress view. In the **Solution Explorer**, the added file icon disappears from the project and its files. Additionally, the project displays the flag icon , indicating it is under source control.


Solution Explorer - WindowsApplication2
Solution 'WindowsApplication1' (2 projects)
WindowsApplication1
My Project
Form1.vb
WindowsApplication2
My Project
Form1.vb
Solution Explorer
Class View

6

Managing Visual Studio Solutions

This section provides information about managing Visual Studio solution under Integrity Lifecycle Manager source control. You can drop and import projects and well as perform many different actions on solutions.

Dropping a Visual Studio Project from a Shared Solution

You can drop a Visual Studio project from a Visual Studio solution under Integrity Lifecycle Manager source control by removing it from the solution. Removing the project removes the project information and project share information from the solution. The corresponding Integrity Lifecycle Manager subproject and any subproject files are not dropped from Integrity Lifecycle Manager source control. By design, this prevents the subproject from being dropped from source control if it is shared with another Visual Studio solution.

After you remove the project from the solution in Visual Studio, you must manually drop the subproject in the Integrity Lifecycle Manager client GUI to completely remove it from Integrity Lifecycle Manager source control. Before you drop the subproject, you want to verify that it is not shared with another Integrity Lifecycle Manager project.

To drop a Visual Studio project:

1. With an active change package set, right-click the project you want to remove in the **Solution Explorer** and select **Remove**. Visual Studio informs you that this removes the project.
2. Click **OK**. The project disappears from the **Solution Explorer**.

   Removing a project from a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, the pencil icon displays in the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view.
3. To submit the updated solution to the Integrity Lifecycle Manager repository, submit the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view.

   The **Check In** window opens for the solution file.
4. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*.

   After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view and the **Solution Explorer**. The solution now displays the flag icon , indicating the changes to the solution file are committed to the Integrity Lifecycle Manager repository.
5. Select **File ▸ Source Control ▸ View Solution Sandbox**. The **Sandbox** view for the solution displays.
6. Select the subproject that corresponds to the Visual Studio project you removed in Visual Studio.

7. Select **Member** ▸ **Drop**. The **Drop Subproject** window opens.
8. Follow the procedure for dropping a subproject, as described in the *Integrity Lifecycle Manager Help Center*. The subproject is removed from the Integrity Lifecycle Manager project.

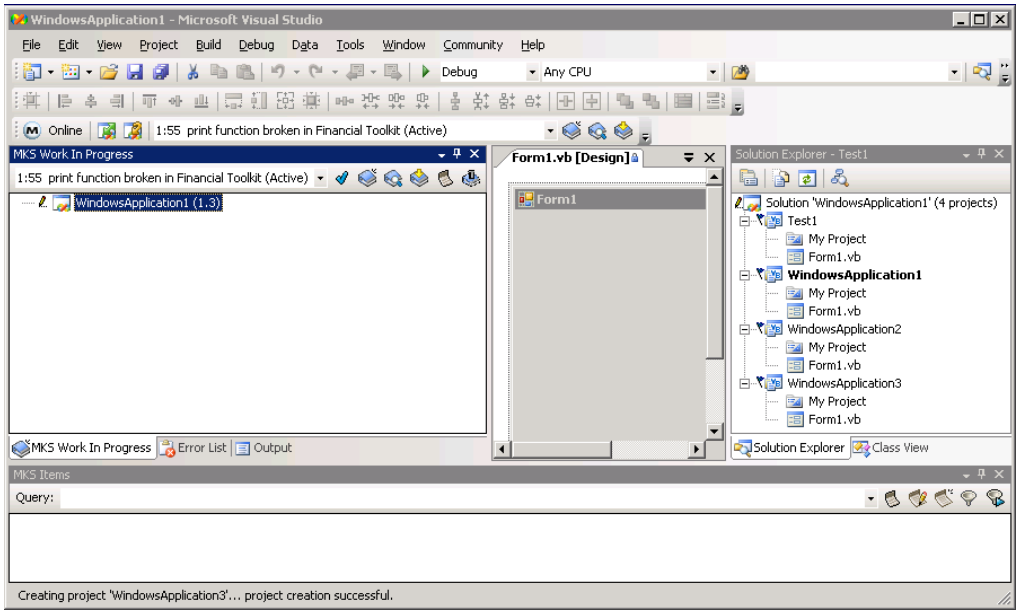
Importing a Visual Studio Project

Importing a Visual Studio project is useful if you want to add a Visual Studio project in a Visual Studio Solution to another Visual Studio solution. This adds the Visual Studio project to the solution's top-level Integrity Lifecycle Manager project as a shared subproject. Both Visual Studio solutions must be under Integrity Lifecycle Manager source control.

To import a Visual Studio project:

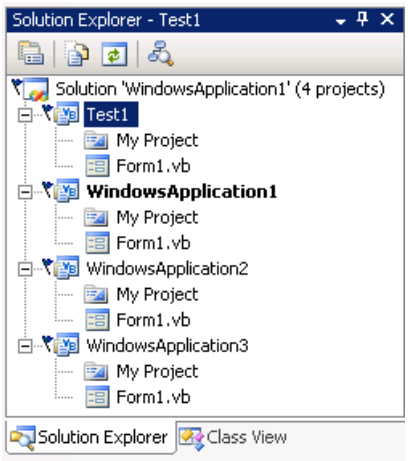
1. Open the Visual Studio solution you want to add a project to, and select **File** ▸ **Source Control** ▸ **Import Project**. The **Import Project** window opens.
2. Importing a project requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
3. From the list, select the solution containing the project you want to import, then click **Next**. To filter the results, type in the **Show share names containing** text field to display solutions matching the name you type.
4. Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.
5. To import the project, click **Import**. In the **Solution Explorer**, the solution displays the imported project. The project displays the flag icon  , indicating it is under Integrity Lifecycle Manager source control.

   Importing a project to a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, the pencil icon  displays in the **Solution Explorer** and Integrity Lifecycle Manager Work In Progress view.



6. To submit the updated solution to the Integrity Lifecycle Manager repository, submit the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view.
7. The **Check In** window opens for the solution file. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*.

   After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. The solution now displays the flag icon , indicating the changes to the solution file are committed to the Integrity Lifecycle Manager repository.



Migrating a Visual Studio Solution from the MKS SCC Visual Studio Integration

If you have an existing Visual Studio solution that was placed under version control using the previous MKS SCC-based Visual Studio integration, you can migrate the solution for use with the current integration.

To migrate a Visual Studio solution from the MKS SCC Visual Studio integration:

1. In the MKS SCC-based Visual Studio integration, open the Visual Studio solution and remove the SCC binding by selecting **File ▸ Source Control ▸ Change Source Control ▸ Unbind**.
2. Ensure that all Visual Studio projects are saved and work properly.
3. Enable the new Visual Studio integration, as described in Setting Up and Configuring the Integration" on page 13.
4. Share the solution, as described in Sharing a Visual Studio Solution on page 32.

> **Note**
>
> After you submit the change package associated with the share operation, some Visual Studio projects can appear in the Work In Progress view under **Unassociated Changes**. These represent changes for unbinding the projects. When you share a Visual Studio solution, the integration does not submit any changes in existing Sandboxes unless the Visual Studio integration changed the files during the share operation.

5. If you add a new Visual Studio project to the migrated solution, right-click the project and select **View member properties**. The **Member Properties** window opens.
6. Click **Allow The Integration To Add Visual Studio Projects To Source Control Automatically**. The button disappears.

   Additional projects added to the solution are automatically added to source control; this button does not appear.
7. Click **Close**.

   For each solution you migrate, repeat this procedure

Ignoring Visual Studio Entities from Integrity Lifecycle Manager Source Control

There can be times when you want to ignore certain Visual Studio entities from source control, for example, temporary files (`.tmp`). When you ignore Visual Studio entities from source control, the entities are not annotated with icons in the **Solution Explorer** and are not valid selections for Integrity Lifecycle Manager commands. The ignored entities are saved to an `.mksignore` file with the Visual Studio project you applied it to, adding it as a member of the Integrity Lifecycle Manager subproject.

To apply the same ignore list to other projects, you must set them on a per-project basis.

Note

Do not edit the `.mksignore` file.

To ignore Visual Studio entities:

1. With an active change package set, right-click the Visual Studio project you want to create an ignore list for in the **Solution Explorer**, and select **Ignore from Source Control**. The **Files Ignored From Source Control** window opens.
2. To add a file type to the filter list, click **Add** and type the file type. To select multiple files, use wildcard characters (`*` or `?`). For example, to ignore all temporary files, type `*.tmp`.
3. To preview the files in the Visual Studio project that you are ignoring, select the filter from the ignore list and click **Preview**. A window opens, displaying a list of files to ignore.
   - If a file has a lock on it, you cannot create an ignore filter for that file. The integration warns you about the affected files, but does not add them to the ignore list. Existing locks are not removed.
   - If you choose to ignore file types that are currently in change packages, the integration warns you about the affected files. However, it does not add them to the ignore list. To ignore the file types, resolve the change packages. You can either submit the change packages or remove specific change package entries. Then, add the file types to the ignore list.
4. Click **OK**.

5. To save your changes, click **OK**. In the Work In Progress view, the `.mksignore` file displays the added file icon , indicating it is a deferred member add.
6. To submit the file to the Integrity Lifecycle Manager repository, submit the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view. The **Create Archive** window opens. Follow the procedure for creating archives, as described in the *Integrity Lifecycle Manager Help Center*.

After the change package is submitted successfully, the `.mksignore` file disappears from the Work In Progress view.

Branching a Visual Studio Solution

Development paths are used to deliberately create a parallel branch of development for the purpose of experimenting with research or performing post-release maintenance. Integrity Lifecycle Manager allows multiple developers to point to the same development path, each using their own variant Sandbox.

When you need to create a new branch (development path) from an existing Visual Studio solution under Integrity Lifecycle Manager source control, you branch the solution. Branching a solution does the following:

- Checkpoints the solution
- Creates a development path on all Integrity Lifecycle Manager projects associated with the solution in the Integrity Lifecycle Manager repository
- Creates a Sandbox for the branched solution

Note

Before you branch a solution, you want to submit solution changes to the Integrity Lifecycle Manager repository with an associated change package. If you perform the branch operation without committing your changes, the uncommitted changes do not appear in the branched solution.

To branch a Visual Studio solution:

1. With the Visual Studio solution open that you want to branch, select **File** ▸ **Source Control** ▸ **Branch Solution**. The integration warns you that the operation creates a branch of the open solution, based on the data currently committed to the Integrity Lifecycle Manager repository. In addition, the open solution closes and a new Sandbox is created.

Tip

If no solution is open, you can select **File** ▸ **Source Control** ▸ **Branch Solution**; however, you are prompted to select a solution from a list of shared solutions.

2. To continue, click **Yes**. The **Branch Solution** window opens.
3. Branching a solution requires associating it with a change package. Select a change package from the **Change Package** list, or create one by clicking **Create**. To proceed, click **Next**.
4. Type a unique name for the **Development Path**, then click **Next**.
5. In the **Share Name** and **Share Description** fields, provide a unique name and description for the solution. The name and description appear to users when they import this branch of the solution in Visual Studio. To proceed, click **Next**.

Tip

You can edit the descriptions of projects, by clicking **Advanced**.

6. Select the location on your local drive to create the Sandbox, then click **Next**. You can type the location or click **Browse** to select a directory. A summary of your choices displays.
7. To branch the solution, click **Branch**. The integration informs you that the branched solution is made public (available for importing) when the change package associated with the branch operation is submitted to the Integrity Lifecycle Manager repository.
8. Click **OK**.

   Branching a solution requires writing to the solution file. To indicate that the solution file is checked out and locked for editing, the pencil icon displays in the **Solution Explorer** and Work In Progress view.
9. To submit the branched solution to the Integrity Lifecycle Manager repository, submit the associated change package. You can accomplish this by clicking the submit change package button in the Integrity Lifecycle Manager toolbar or the Work In Progress view.

   The **Check In** window opens for the solution file.
10. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*.

After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. In the **Solution Explorer**, the pencil icon disappears. Other users can now create a Sandbox from the branch by importing the solution. In the **Import Solution** window, a branched solution displays the development path name in brackets after the solution name. An example is `FinancialToolkit(ServicePack1)`.

Resynchronizing a Visual Studio Solution

Resynchronizing a Visual Studio solution under Integrity Lifecycle Manager source control updates your Sandbox with the latest solution, project, and files. For example, assume that you see the icon in the Work In Progress view and **Solution Explorer**. This means that a new member revision is available. You want to resynchronize the solution.

To resynchronize the solution, do one of the following:

- Select **File** ▸ **Source Control** ▸ **Resynchronize Solution**.
- Click the resynchronize solution button in the Integrity Lifecycle Manager toolbar.
- Right-click the solution in the **Solution Explorer** and select **Resynchronize Solution**.

The integration updates the solution, removing any incoming changes from the Integrity Lifecycle Manager Work In Progress view and updating source control status icons in the **Solution Explorer**.

Reverting a Visual Studio Solution

Reverting a Visual Studio solution reverts all files in the solution containing working file changes. However, it does not remove any recently added files to the solution from the file system. In most Visual Studio projects, a revert operation removes recently added files from the project and associated change package. In a website project, a revert operation removes recently added files in the associated change package. Additionally, it displays the files under **Unassociated Changes** in the Integrity Lifecycle Manager Work in Progress view. To remove the files from the website project, you delete them.

Note

If change package reviews are enabled, submit any pending revisions before you revert a Visual Studio solution.

To revert the solution, do one of the following:

- Select **File** ▸ **Source Control** ▸ **Revert Solution**.
- Click the revert solution button  in the Integrity Lifecycle Manager toolbar.
- Right-click the solution in the **Solution Explorer** and select **Revert Solution**.

A window opens, displaying a warning. The warning indicates that reverting the solution overwrites all of your changes that have not been committed to the Integrity Lifecycle Manager repository. It also indicates that this operation cannot be canceled.

To proceed, click **OK**. The integration updates the solution, removing the change package entries from the Integrity Lifecycle Manager Work In Progress view and updating the icons in the **Solution Explorer**.

Checkpointing a Visual Studio Solution

Checkpointing a Visual Studio solution creates a new revision of the associated Integrity Lifecycle Manager project and adds it to the project history. When you checkpoint a solution, you save all the information needed to recreate the solution completely as it existed when you checkpointed it. The saved information includes the solution and project structure and the list of files with their revision numbers.

To checkpoint a Visual Studio solution:

1. Open the Visual Studio solution you want to checkpoint.
2. Select **File** ▸ **Source Control** ▸ **Checkpoint**. The **Checkpoint Solution** window opens.
3. Add a **Label** and a **Description**.
4. Click **OK**.

Viewing a Sandbox for a Visual Studio Solution

You can view the top-level Sandbox for an open Visual Studio solution by selecting **File** ▸ **Source Control** ▸ **View Solution Sandbox**. The **Sandbox** view displays.

For more information on the Sandbox view, see the *Integrity Lifecycle Manager Help Center*.

7

Working With Visual Studio Files

To provide a more seamless Integrity Lifecycle Manager experience within Visual Studio, most basic Integrity Lifecycle Manager file commands occur implicitly as deferred operations when you perform the equivalent Visual Studio commands. Basic operations include adding members and checking out files.

Before you run a command, ensure that an active change package is selected in the change package list. After you complete the command, submit the change package to commit the changes to the Integrity Lifecycle Manager repository. For more information on submitting change packages, see Managing Work In Progress on page 23.

Most file commands require writing to the solution or project file. When you perform a file command, the solution or project can display the pencil icon  in the **Solution Explorer** and Work In Progress view. This icon indicates that the solution or project file is checked out and locked for editing.

Adding Members to an Integrity Lifecycle Manager Project

Adding a file to a Visual Studio project performs a deferred add operation on the file.

Note

When using the **Copy** command, the revision history is not copied from the source location to the target location.

To add a member to an Integrity Lifecycle Manager project:

1. In Visual Studio, add a file to a Visual Studio project under Integrity Lifecycle Manager source control. In the Work In Progress view and **Solution Explorer**, the pencil icon displays next to the new file. This same icon also displays next to the project file.
2. Submit the change package associated with the add operation. The **Create Archive** window opens. Follow the procedure for creating archives, as described in the *Integrity Lifecycle Manager Help Center*.

   The **Check In** window opens for the project file.
3. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*. After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. In the **Solution Explorer**, the added file icon and pencil icon disappear.

Dropping Members From an Integrity Lifecycle Manager Project

Deleting or removing a file from a Visual Studio project performs a deferred drop operation on the file. After the change package is submitted, the file is deleted from the file system and removed from the Integrity Lifecycle Manager repository.

Note the following:

- For a website under Integrity Lifecycle Manager source control, the Visual Studio **Exclude From Project** command simply removes the file from the website. The file is left in the file system and under Integrity Lifecycle

Manager source control. The Visual Studio **Delete** command deletes the file from the file system and removes it from the Integrity Lifecycle Manager repository.

- You can delete files from a Visual Studio project while in offline mode and without an active change package. However, the deleted files are not dropped from Integrity Lifecycle Manager source control and do not appear in the Integrity Lifecycle Manager Work In Progress view under **Unassociated Changes**. In addition, the files are deleted from the Visual Studio project and file system, preventing you from moving them to a change package. To allow you to recover or resolve deleted files, you want to delete files in online mode and with an active change package.

To drop a member from an Integrity Lifecycle Manager project:

1. In Visual Studio, delete or remove a file from a Visual Studio project under Integrity Lifecycle Manager source control. The file is removed from the **Solution Explorer**. In the Work In Progress view, a dropped file icon displays next to the file under the **Non-Solution Files** heading. This icon indicates that the file is no longer associated with the solution. In the **Solution Explorer** and Work In Progress view, the pencil icon displays next to the project file.
2. Submit the change package associated with the drop. The **Drop Member** window opens.
3. Follow the procedure for dropping members, as described in the *Integrity Lifecycle Manager Help Center*.

   The **Check In** window opens for the project file.
4. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*. After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. In the **Solution Explorer**, the dropped member icon and pencil icon disappear.

Checking Out Members

In Visual Studio, open a file for editing and make changes. After you save changes to the file, the pencil icon displays next to it in the Integrity Lifecycle Manager Work In Progress view and **Solution Explorer**. This icon indicates that you have checked out and modified the file.

Checking In Members

In Visual Studio, submit the change package associated with a file checked out by you. The pencil icon appears next to such a file in the Work In Progress view and **Solution Explorer**.

The **Check In** window opens for the file. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*. After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. In the **Solution Explorer**, the pencil icon disappears.

Renaming Members

Renaming a file in Visual Studio performs a deferred rename operation on the file.

Note

If you attempt to rename a file that you have locked in another change package, the rename operation is performed in the change package containing the locked file. For example, assume that you locked `Form.vb` in change package `12:1` and your active change package is `12:2`. Renaming `Form.vb` to `Form2.vb` performs the rename operation in `12:1` because `Form.vb` exists in this change package. This also applies to moving a file or locking a file that is a deferred rename operation in a change package.

To rename a file:

1. In Visual Studio, rename a file in a Visual Studio project under Integrity Lifecycle Manager source control. In the **Solution Explorer**, the renamed file displays the pencil icon. In the Work In Progress view, the renamed file displays the renamed or moved file icon. In Work In Progress view and the **Solution Explorer**, the Visual Studio project displays the pencil icon.
2. Submit the change package associated with the rename. The **Check In** window opens for the project file.
3. Follow the procedure for checking in a member, as described in the *Integrity Lifecycle Manager Help Center*. The **Check In** window opens for the renamed file.
4. Check in the member. After the change package is submitted successfully, the change package and its entries disappear from the Work In Progress view. In the **Solution Explorer**, the pencil icon disappears.

Moving Members

You can move folders and files between projects in a Visual Studio solution, which performs a deferred move operation.

Note the following:

- Dragging and dropping or using the **Cut** and **Copy** commands to move a folder or file performs a deferred drop and add operation, effectively erasing the revision history. If you use these commands to move a file in offline mode without an active change package, the file is removed from the Visual Studio project. However, the file is not removed from Integrity Lifecycle Manager source control. To move a file, you want to use the **Integrity Move From** and **Integrity Move To** commands in online mode with an active change package.
- Moving a file to a location where a file with the same name exists results in an error.
- If you move an entire tree of files, the source folders are still visible in the **Solution Explorer**. After moving a tree of files, you must manually remove the source folders.

To move a folder or file between projects:

1. Do one of the following:
   - In the **Solution Explorer**, right-click the file or folder which you want to move and select **Integrity Move From**.

     Select the project to which you want to move the file or folder. Then, right-click and select **Integrity Move To**.
   - Drag the file or folder you want to move and drop it in the desired project.
   - Right-click the file or folder you want to move and select **Cut** or **Cut**.

     Select the project to which you want to copy the file or folder. Then, right-click and select **Paste**.

   In the **Solution Explorer**, a moved file displays the pencil icon  . In the Integrity Lifecycle Manager Work In Progress view, the renamed or moved file icon  displays.
2. Submit the change package associated with the move. After the change package is submitted successfully, the change package and its entry disappear from the Work In Progress view. In the **Solution Explorer**, the pencil icon disappears.

Accessing Advanced Integrity Lifecycle Manager Commands

You can access advanced Integrity Lifecycle Manager member functionality from within Visual Studio. You accomplish this by right-clicking a Visual Studio solution, Visual Studio project, or file in the **Solution Explorer** or Work In Progress view. You can then select a command from the shortcut menu.

Note

In the **Solution Explorer**, shortcut menu operations operate on the selected file. The shortcut menu in the editor window operates on the currently open file. If Visual Studio does not allow you to select the file and display a shortcut menu, you can click **Show All Files** in the **Solution Explorer**. You can then use the context menu operations from there.

The following Integrity Lifecycle Manager member commands are available:

Command	Function
Resynchronize	Equivalent to the Integrity Lifecycle Manager **Resynchronize** command. Gets the latest version of the selected file and puts it in your working directory. This command is only available from the **Solution Explorer** and Work In Progress view when incoming changes are present. An overwrite window can open.
Revert	Equivalent to the Integrity Lifecycle Manager **Revert** command. Replaces the working file with the revision that was checked out, as it appeared before modification. It also unlocks the file and removes it from the associated change package. This command is only available from the Integrity Lifecycle Manager Work In Progress view. An overwrite window can open. Note the following: • You can revert a file in a change package that is not the active change package. Reverting the file removes it from the change package. • If you add a file to a Visual Studio project and want to revert the operation, remove the file from source control.

Command	Function
	Or, if you removed it from the project, add it again. If you rename or move a file and want to revert the operation, move or rename the file again. The **Revert** command is intended for reverting changes to a working file.
View differences	Equivalent to the Integrity Lifecycle Manager **View Differences** command. Compares the selected working file with the member revision. Visual Difference automatically launches and displays the two files.
View annotated revision	Equivalent to the Integrity Lifecycle Manager **View Annotated Revision** command. Displays the annotated revision history of the selected file. The **Annotated Revision** view displays.
View member history	Equivalent to the Integrity Lifecycle Manager **View Member History** command. Displays the revision history of the selected file. The Member History view displays.
View member properties	Equivalent to the Integrity Lifecycle Manager **View Member Information** command. Displays the member information of the selected file. A window opens, displaying server name and port number, solution share name, solution project, and member name. It also displays the project share name, project configuration path, member name, Sandbox, member revision, working revision, and development path (if applicable). To view additional Integrity Lifecycle Manager information, click **Integrity Member**. The Member Information view displays.

8

Best Practices

This section describes best practices for using the Visual Studio integration. It points out efficiencies that can be gained by using the integration in a certain way. Additionally, it identifies any risks, constraints, or other limits within the integration or within a particular implementation of the integration.

Efficiencies

You can gain efficiencies by using the Visual Studio integration in a certain way.

- Displaying working file status for overlapping Sandboxes and subsandboxes

  Ensure correct status displays for working files in overlapping Sandboxes and subsandboxes.

  - Do not change CreateSubproject policies (creating subprojects for every folder when adding or moving members) without also manually restructuring existing project trees.
  - Avoid using mixed CreateSubproject policies across Visual Studio solutions and projects.
- Notification of file changes in Visual Studio Solutions

  To prevent file change notifications from appearing when you resynchronize or revert a Visual Studio solution, set the following preferences in Visual Studio:

  1. Select **Tools ▸ Options ▸ Environment ▸ Documents**.
  2. Enable **Detect when file is changed outside the environment** and **Auto-load changes, if saved**.
  3. To save your preferences, click **OK**.
- Creating In-Tree Visual Studio projects and websites

  As a best practice, place Visual Studio projects and websites in-tree with the associated Visual Studio solution to simplify source control operations for other users. Visual Studio projects and websites that are out-of-tree can cause confusion or problems when users import or branch solutions.
- Setting Up Projects to Reuse Code

  The ability to reuse code is an important part of managing software development. If you set up your project structure so that each component is self-contained, you can share that component with other solutions by using subprojects. By referencing the original subproject, Integrity Lifecycle Manager allows you to share a subproject and the members it contains between two or more projects. Shared subprojects do not have to be located within the same directory structure or project hierarchy.

  To share project code, import an existing Visual Studio project to the Visual Studio solution that you want to share the project with. Projects are shared at the solution level; websites are shared at the solution root level. For more information on importing Visual Studio projects, see Importing a Visual Studio Project on page 43.
- Creating and sharing a Visual Studio project

You always want to enable the following option in Visual Studio before you create a Visual Studio project: **Tools ▸ Options ▸ Projects and Solutions ▸ General ▸ Save new Projects when created**. By default, this option is enabled. If this option is disabled, a Visual Studio project cannot be shared.

- Ignoring files from source control

  There are times where you want to ignore certain Visual Studio entities from Integrity Lifecycle Manager source control. For example, you probably want to ignore temporary (.TMP) files. As a best practice, specify the entities you want to ignore before you add files to a Visual Studio project. For best results, you do not want ignored files to be placed under source control.

- Java memory usage

  If you are working with large solutions or large change packages, increase the Java heap size on the Integrity Lifecycle Manager client for better performance. For example, you can increase it to 512 MB. For more information, contact PTC Technical Support.

- Working with multiple Visual Studio Solutions

  Opening a Visual Studio solution closes the currently open Visual Studio solution. You want to save your changes and submit any associated change packages before you open another Visual Studio solution.

- Submitting change packages in Visual Studio

  To avoid potential issues with updating the Integrity Lifecycle Manager repository, always submit change packages from Visual Studio, not the Integrity Lifecycle Manager client.

- Resolve conflicts before submitting change packages

  Integrity Lifecycle Manager allows you to resolve conflicts when checking in members; however, PTC recommends resolving conflicts by resyncing the repository before submitting a change package. This avoids potential workflow issues and allows compiling and testing with the incoming changes in the Sandbox.

- Refactoring in a team environment

  When working in a team environment, there are refactoring scenarios that you need to be aware of to avoid or prevent conflicts.

Note

To ensure that you can recover or resolve refactored files, perform all refactoring operations in online mode with an active change package. Such operations include adding, dropping, moving, and renaming members.

- Renaming the same member

  David renames `Module1.vb` to `Module2.vb`, but does not submit the change package.

  Mary renames `Module1.vb` to `Module3.vb` and submits the change package.

  In his Sandbox, David sees that a rename and lock operation are removed from his change package, and that `Module2.vb` is now a former member.

  To recover and save his work, David must manually make changes.
- Renaming a locked member

  In a change package, Mary has a lock on `Application1.vb`.

  David cannot rename `Application1.vb` because of the lock.

  David can drop `Application1.vb` and add it as `Application2.vb`, ignoring Mary's lock on `Application1.vb`.
- Adding the same member

  David adds `Form1.vb` to a project, but does not submit the change package.

  Mary also adds `Form1.vb` to the same project and submits her change package.

  In his Sandbox, David sees that the project file contains a local modification and an incoming change. `Form1.vb` displays as an add operation.

  If David submits his change package, an error message displays informing him that the member is already a member of the project. Refreshing the **Solution Explorer** removes the add operation icon.

  The change package can no longer be submitted. Each attempt to submit the change package results in a new member revision with the changes David made before submitting the change package.

- Backing up Visual Studio share information

  Information about shared Visual Studio solutions and projects is stored to the following file on the Integrity Lifecycle Manager server:
  `<Integrity Lifecycle Manager server installdir>/data/vsi/vsibinding.properties`

  PTC recommends backing up this file along with other Integrity Lifecycle Manager server files; however, do not edit the file. Editing the file can cause shares to not work properly.
- Viewing visible fields in query results

When running a query from the Integrity Lifecycle Manager Items view in Visual Studio, query results are displayed with the **ID**, **Type**, **State**, and **Summary** columns. This occurs even when additional columns are set as visible. To see all visible fields, you must set the column set to **Custom**.

To see all visible fields when running a query, open the client and select the target query. Choose **Query ▸ Edit ▸ Column Set** and select the **Custom** option. When running the query, all visible columns are then displayed.

Limitations

- While loading a Visual Studio solution, Integrity Lifecycle Manager commands do not necessarily appear consistently in shortcut menus.
- Migrating a build Sandbox from the previous SCC-based Visual Studio integration to the current SDK integration is not supported. This is because a migration causes the member files to change, which is a limitation of build Sandboxes.
- Assume that two users simultaneously add Visual Studio projects to a Visual Studio solution. The project information is not saved in the merged Visual Studio solution file for the fist user who submitted the solution file. This generates an error message stating that Visual Studio cannot load the added project.

  To avoid this error, select `Option 2 - resynchronize the member revision via change package, merging as needed`. If you do not select Option 2, you must resynchronize the Visual Studio solution containing the unavailable project.
- If you enabled password prompting on the Integrity Lifecycle Manager client, the integration cannot use the specified default password. When prompted in the integration, you must retype your password.
- Moving a file from one Visual Studio project to another can take a long time to complete if your Recycle Bin contains too many items. This is a known issue with Visual Studio. Emptying your Recycle Bin improves the speed of the move operation.
- The Visual Studio integration does not support renaming Visual Studio solutions, Visual Studio projects, or websites. You can rename directories; however, you must commit any changes to files in a directory before the rename operation. If you are moving a directory, you must also commit any changes to files in the directory before the move operation.
- Importing a Visual Studio solution containing unshared Visual Studio projects displays an error message for each unshared project in the solution.

- Date-only fields can incorrectly display as date/time fields in the Integrity Lifecycle Manager Items view.
- Assume that you create two top-level Integrity Lifecycle Manager projects for a website solution and website project, and two separate Sandboxes associated with each Integrity Lifecycle Manager project. When you share the website solution in Visual Studio, the website project files are not included in the change package used for the share operation. The website project files display as **Unassociated Changes** in the Work In Progress view and only the solution is shared. To resolve this issue, you must move the website files from **Unassociated Changes** to a change package and submit the change package.

Troubleshooting

You can use the `Integrityvsi.log` file to help in diagnosing issues that can arise when using the Visual Studio integration. This file is created on the machine where Visual Studio and the Integrity Lifecycle Manager client are installed. The file records information, warnings, and error messages.

To open the file, do the following:

1. Select **Tools** ▸ **Options** ▸ **Source Control** and then select the **Integrity** plug-in. The **Logging** option displays.
2. Click **Logging**.
3. In the right pane, click **Open**.

If you require assistance, contact PTC Technical Support and provide them with the log file. In addition, include the version numbers of the Integrity Lifecycle Manager Visual Studio integration and the Integrity Lifecycle Manager client with which it is communicating. To obtain this information, in Visual Studio, click the PTC button  in the Integrity Lifecycle Manager toolbar.

Getting Help

PTC Technical Support is focused on delivering the right solutions to issues as they arise. Online support provides easy access to e-mail, Web request services, automatic product notifications, and the PTC Integrity eSupport portal. This secure database provides helpful resources such as product documentation, knowledge base articles, product downloads, user forums, and presentations. For online support, browse to http://www.ptc.com/support/integrity.htm.

PTC global technical support professionals comprise a tightly knit team of problem solvers. By sharing critical information, they help you resolve issues in the shortest possible time with optimal results. Support representatives can provide you with various product-related tips and innovative solutions to your unique requirements.

integrity™ lifecycle manager