madcap software

USER GUIDE

MADCAP LINGO 11 r3

Source Control: Subversion

Copyright © 2023 MadCap Software. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of MadCap Software.

MadCap Software
9171 Towne Center Drive, Suite 335
San Diego, California 92122
858-320-0387
www.madcapsoftware.com

THIS PDF WAS CREATED USING MADCAP FLARE.

CONTENTS

CHAPTER 1

CHAPTER 2

CHAPTER 3

CHAPTER 4

APPENDIX

CHAPTER 1

Introduction

You can use the Lingo interface to perform various source control tasks for a project that is bound to Subversion.

General Information

- "Common Source Control Terms" on page 7
- "Source Control Icons" on page 8
- "Bind Detection, Disabling Providers, and Unbinding Providers–Subversion" on page 10

Process

1. Install and Set Up Subversion (done outside of Lingo)
2. "Binding a Project to Subversion" on page 14
3. (Other Team Members) "Importing From Subversion" on page 17
4. "Updating Source Control Files" on page 18
5. "Committing Source Control Files" on page 19
6. "Merging Source Control Files" on page 23

CHAPTER 2

General Information for Subversion

There are various pieces of general information you should know if you plan to use this feature.

This chapter discusses the following:

Common Source Control Terms

Following are definitions for some of the common phrases used in Lingo's integrated source control with Subversion.

- **Bind** This means to connect your project to Apache Subversion. After doing this, you can take advantage of all the automated source control tasks (such as commit, revert, update, and so on).
- **Commit** This means to send changes from your working copy of a Lingo file to the Subversion repository, on a server.
- **Revert** This means to undo changes you have made to a Lingo file. Changes are reverted to the way they were at the last commit.
- **Update** This means to update your working copy of a Lingo file with changes from the repository.
- **Lock** This means to prevent other users from committing changes to a Lingo file in the repository.
- **Unlock** This means to remove an existing lock from a Lingo file in the repository so other users can commit changes to the file.

> **NOTE** Lingo integrates with multiple source control providers to provide built-in source control support. Each of the source control providers built-in to Lingo uses different terms. As such, Lingo's source control interface is different depending on which source control provider you use. Please refer to the sections for each source control provider if you need to see information about the terms used by other built-in systems.

Source Control Icons

Following are descriptions for the primary icons that you may see next to files when using Subversion.

✓

Modified

This indicates that the file has been modified. You can commit the file when you are ready (if you are the user who has modified it, or if you have stolen the lock on the file from another user).

✚

New File (Add)

This indicates that you have a file in your project but have not yet added it to Subversion. This might occur, for example, if you create a new topic and do not add the file to source control during the topic creation process. To resolve this, simply right-click on the file and select **Source Control > Add**.

🔒

Locked by Another User

This indicates that another user has locked the file. You will often see this icon in conjunction with the 👤 icon, indicating that another user is using the file and has locked the file. You can make changes to this file even if another user has locked it.

If you need to commit the file in while another user is working on it, you can steal their lock. To do this, right-click on the file and select **Source Control > Lock**. In the Lock dialog, select **Steal the lock**, then click **Lock**.

🔒

Locked by You

This indicates that you have locked the file. You will often see this icon in conjunction with the ✓ icon, indicating that you have both locked and modified the file. Other users can make changes to this file even if you have locked it, but they cannot commit it unless they steal the lock from you first.

👤

In Use by Other User

This indicates that the file is currently being modified by another user. You will often see this icon in conjunction with the 🔒 icon, indicating that another user is using the file and has locked it.

Renamed

This indicates that a file has been renamed, but not modified in any other way. If you make any additional changes to the file, such as editing the text or adding a condition tag, the renamed icon is replaced by a ✓ icon.

Out of Date

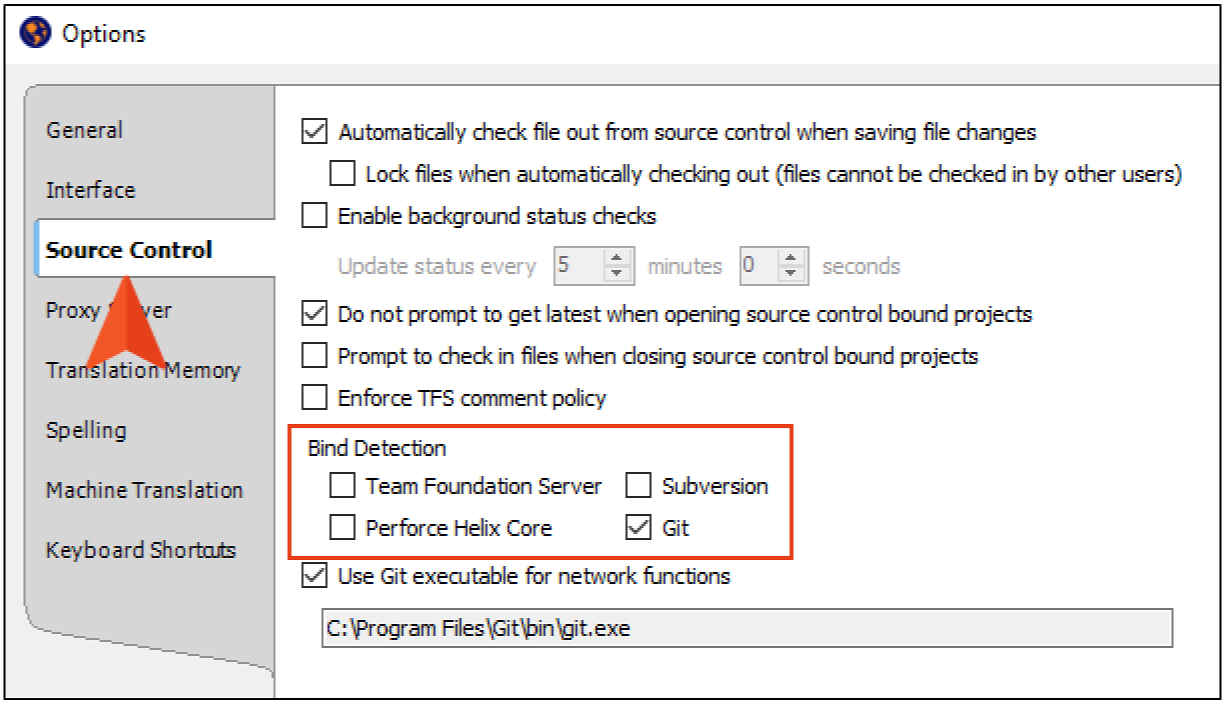
This indicates that the file is not current (i.e., the local copy of the file is older than the source control copy). This might happen, for example, if another user modifies the file and commits it to source control. If this occurs, you can modify the file or update the file from source control.

Bind Detection, Disabling Providers, and Unbinding Providers—Subversion

Lingo has options for bind detection, disabling providers, and unbinding providers. Although these are separate features, they are all somewhat related. This information is especially important if you are using an external tool to bind and manage your source control tasks.

Bind Detection

Lingo's bind detection settings are found on the Source Control tab of the Options dialog.



☑ Automatically check file out from source control when saving file changes

□ Lock files when automatically checking out (files cannot be checked in by other users)

□ Enable background status checks

Update status every 5 minutes 0 seconds

☑ Do not prompt to get latest when opening source control bound projects

□ Prompt to check in files when closing source control bound projects

□ Enforce TFS comment policy

Bind Detection

□ Team Foundation Server □ Subversion

□ Perforce Helix Core ☑ Git

☑ Use Git executable for network functions

C:\Program Files\Git\bin\git.exe

Bind detection scans your project when you load it to see if the project has been previously bound to source control. If a binding is detected, you then have the option of applying the binding and committing the project to source control. Depending on the provider you are using, Lingo may search the file system and its artifacts, as well as contact and query servers, to find potential source control bindings.

When you open a Lingo project that hasn't been bound to source control before, the bind detection option is disabled for Perforce Helix Core, Subversion, and Team Foundation Server. It is enabled by default for Git. If you bind a project to source control using the Lingo interface, the option is then automatically enabled.

NOTE You can use bind detection as an alternative to importing a Lingo project. If you have received a Lingo project file (e.g., by copying it from a server, by opening it from a network location), you can simply open the file and Lingo will search for and apply existing source control bindings.

NOTE Source control providers are scanned in the following order:

1. Git
2. Subversion
3. Perforce Helix Core

TIP Detecting source control bindings may take a considerable amount of time. It is recommended that you select only the source control providers that you use to speed up the detection process.

Disabling Providers

By default, when a project is bound to source control, the provider (Git, Perforce Helix Core, Subversion, or Team Foundation Server) is enabled. This means that the source control interface elements in Lingo are visible, and you can use them to perform various tasks (e.g., commits, synchronize changes).

Disabling a provider means that the source control interface elements are no longer shown. This does not mean you cannot use source control. As long as the provider is still *bound* to the project, you can perform source control tasks in a third-party tool outside of Lingo.

For more details and steps, see "Disabling a Subversion Provider" on page 29.

Unbinding Providers

When you unbind a provider, it means you are removing the connection altogether between the Lingo project and the local repository.

You can unbind a provider via the Project Properties dialog or the Settings view in the Source Control Explorer. Click the **Unbind Provider** button.

For more details and steps, see "Unbinding a Subversion Provider" on page 48.

CHAPTER 3

Process for Subversion

Certain tasks must be completed in order when using this feature.

This chapter discusses the following:

Binding a Project to Subversion

Use the following steps if you have already created a Lingo project and want to bind ("connect") it to Subversion. You can also automatically detect existing source control bindings if your project has been previously connected to Subversion.

> **NOTE** The following steps show how to bind a project using the Lingo interface. It is also possible to bind a project outside of Lingo (e.g., using Git Bash). If you decide to do this, you should be aware of some additional aspects of source control, such as bind detection and disabling providers.

How to Bind a Project Using the Project Properties Dialog

1. Open the project.
2. Select **File > Project Properties**. The Project Properties dialog opens.
3. Select the **Source Control** tab.
4. Click **Bind Project**. The Bind Project dialog opens.
5. From the drop-down, select **Subversion**.
6. In the **Server** field, enter the IP address.
7. Next to the **Project Path** field, click [...].
8. Click on the Subversion folder to which you want to bind the Lingo project.
9. Click **OK**.
10. (Optional) In the **Comment** field, you can enter any internal comments.
11. In the Bind Project dialog, click **OK**.
12. If the Log In dialog opens, complete the **User name** and **Password** fields and click **OK**. Copies of the Lingo files are created and added to the folder you specified.
13. In the Project Properties dialog, click **OK**. The project is connected to Subversion, and you can now commit files as necessary.

How to Bind a Project Using the Explorer

1. Open the project.
2. Select **View > Source Control Explorer**. The Source Control Explorer opens.
3. From the drop-down or the Home pane, select **Settings**. The Settings pane opens.
4. Click **Bind**. The Bind Project dialog opens.
5. From the drop-down, select **Subversion**.
6. In the **Server** field, enter the IP address.
7. Next to the **Project Path** field, click ....
8. Click on the Subversion folder to which you want to bind the Lingo project.
9. Click **OK**.
10. (Optional) In the **Comment** field, you can enter any internal comments.
11. In the Bind Project dialog, click **OK**.
12. If the Log In dialog opens, complete the **User name** and **Password** fields and click **OK**. Copies of the Lingo files are created and added to the folder you specified. The project is connected to Subversion, and you can now commit files as necessary.

What’s Noteworthy?

TIP If you are having difficulty binding your project, try binding to a brand new directory in your source control provider. You should also ensure that the directory on your local machine (and its parent directories) is not already mapped to source control, as this may cause issues with binding.

NOTE You can also bind a new Lingo project to source control while creating it.

Importing From Subversion

This focuses on importing a Lingo project from source control. You might use this method, for example, if you are working on a project with several other translators and another member of the team has placed the Lingo project in Apache Subversion.

How to Import a Project From Subversion

1. Select **File > New Project > Import Project**. The Import Project from Source Control Wizard dialog opens.
2. From the drop-down, select **Subversion**.
3. In the **Server** field, enter the name of the computer or server IP address.
4. Click **Next**.
5. Next to the **Project** field, click **Browse**. The Browse Source Control Files dialog opens. (You may need to log in with your user name and password.)
6. Find and click on the Lingo project file (LIPRJ) that you want to import. (You may need to log in with your user name and password.)
7. Click **OK**.
8. Click **Next**.
9. In the **Project name** field, the name of the project being imported is displayed. It is recommended that you leave the name as it is, especially if you are working with other users on the project. However, you can enter a different project name if you want.
10. In the **Project folder** field, either accept the default location for the new project or click ... to browse for and select a folder.
11. Click **Finish**. The project is imported and loaded into Lingo.

Updating Source Control Files

After you bind a Lingo project to Subversion, you can update any of the source control files. When you do this, you are copying the most current files stored in Subversion to your local Lingo project.

How to Update Source Control Files

1. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Update** (for selected files) or **Source Control > Update All** (for all files in the project).
   - **Right-Click** In the File List, right-click the file you want to update and select **Source Control > Get Latest Version**.
   - **Source Control Explorer** With the Pending Changes pane open, right-click the file you want to update and select **Get Latest Version**.

2. If the local and server files are the same, a message tells you so. Click **OK**.

   If the local file is different from the file on the server, the Resolve Conflicts dialog opens. If you want to accept all of the differences between the local and server files, thus merging them, click **Auto Merge All**. If you want to review the differences in the files side by side and resolve each conflict, click **Resolve**.

NOTE By default, when you open a project that is bound to source control, a message automatically asks if you want to update files. However, you can disable this prompt in the Options dialog (**File > Options**).

Committing Source Control Files

When you are finished editing files, you can commit them to source control. Committing a file overwrites the old copy of the file in the source control database with the new one from your local machine. So even if others will not be working on that file, it is a good idea to periodically commit files so that you have a backup in source control.

However, if that is not the case, the Resolve Version Conflict dialog opens to let you know that another user has already committed the file with changes. You can merge the files automatically if there are no conflicting changes (i.e., changes do not occur in the same location in the file). If there are conflicting changes, you can use the Merge Changes dialog to determine how changes are merged.

This chapter discusses the following:

How to Commit Files to Source Control

1. Do one of the following, depending on the part of the user interface you are using:

    - **Pending Changes Window Pane** From the **Source Control** ribbon, open the Pending Changes window pane. Select the files in the window pane that you want to commit, and in the local toolbar click Commit....
    - **Ribbon** Select **Source Control > Commit** (for selected files) or **Source Control > Commit All** (for all files in the project).
    - **Right-Click** In the File List, right-click the file you want to commit and select **Source Control > Commit** (for selected files) or **Source Control > Project > Commit All** (for all files in the project).

2. (Optional) Enter a comment tied to the commit. This enables you to keep an audit trail for a file. The comment can then be viewed from the History dialog, which can be accessed from the Source Control Explorer, the Source Control ribbon, or the Source Control button .
3. (Optional) If you want to see all files with pending changes (rather than only those you selected), click .

4. Make sure to click the check box next to each file you want to check in so that it contains a check mark.
5. If you have files locked and you want to keep them locked, select **Keep locks**. Doing this will overwrite the source control copies of the files so that they have the latest changes, but it lets you keep the file locked so others will know you are working on it.
6. Click **Commit**.

   If no other users have also made changes to the file and committed in while you were working on it, your version of the file is committed.

   However, if that is not the case, the Resolve Version Conflict dialog opens to let you know that another user has already committed the file with changes. You can merge the files automatically if there are no conflicting changes (i.e., changes do not occur in the same location in the file). If there are conflicting changes, you can use the Merge Changes dialog to determine how changes are merged. See "Merging Source Control Files" on page 23.

How to Commit Files to Source Control Using the Explorer

1. Select **View > Source Control Explorer**. The Source Control Explorer opens.
2. From the drop-down or the Home pane, select **Pending Changes**.

   The Pending Changes pane opens. Files that will be committed are listed under **Included Changes**, and files that will not be committed are listed under **Excluded Changes**. You can identify modified files because they say **[modified]** next to the file name.

   > **NOTE** When you modify a file in source control, you may sometimes see a SKL file alongside the XLF and original files. This skeleton file is a placeholder file. Be sure to commit all three associated files together. Committing just the SKL file may result in errors.

3. (Optional) In the **Comment** field, enter a comment tied to the commit. This enables you to keep an audit trail for a file. The comment can then be viewed from the History dialog, which can be accessed from the Source Control Explorer, the Source Control ribbon, or the Source Control button.
4. (Optional) If you want to select the files or folders that you include in the commit, right-click a file or folder and select one of the following options from the context menu.

   - **Exclude** Excludes the selected file from the commit
   - **Exclude Unselected** Excludes all unselected files from the commit
   - **Include** Includes the selected file in the commit
   - **Include Unselected** Includes all unselected files in the commit

5. Click **Commit Included** to commit all of the files in the Included Changes list.

   If no other users have also made changes to the file and committed in while you were working on it, your version of the file is committed.

   However, if that is not the case, the Resolve Version Conflict dialog opens to let you know that another user has already committed the file with changes. You can merge the files automatically if there are no conflicting changes (i.e., changes do not occur in the same location in the file). If there are conflicting changes, you can use the Merge Changes dialog to determine how changes are merged. See "Merging Source Control Files" on the next page.

What’s Noteworthy?

NOTE Subversion will automatically lock a modified file when saving changes (if it is not already locked) if you selected the option to automatically check out files from source control when saving them. Because there is not a "checked out" status for Subversion files, these files will be marked as modified ✓ and can be committed.

Merging Source Control Files

There may be times when you need to merge changes from different authors when committing a file. The merge occurs automatically if there are no conflicting changes (i.e., changes do not occur in the same location in the file). If there are conflicting changes, a dialog opens, allowing you to determine how changes are merged.

How to Merge Source Control Files

1. Go through the process of updating files from source control or committing files. See"Updating Source Control Files" on page 18 and "Committing Source Control Files" on page 19. If your local copy of the file is different from the server copy (e.g., another author has already committed the same file), the Resolve Conflicts dialog opens.
2. Click **Auto Merge All**. If changes from the other author do not conflict with your changes, this will merge all changes. A message lets you know that a backup of your local copy has been made. This lets you restore that file if you do not want to keep the merged version. You do not need to complete the rest of the steps below.

   However, if your changes conflict with those from another author, a message displays to tell you. In this case, continue with the next step.
3. Click **OK** on the conflict message.
4. In the Resolve Conflicts dialog click **Resolve**. The Resolve Version Conflict dialog opens. From this dialog, you can choose from the following options.

   - **Merge changes for me** Automatically merges changes within the same file that are not part of the same element. If changes have been made to the same element (e.g., the same segment), Lingo will display a prompt to merge the changes using the merge tool.
   - **Merge changes in merge tool** Opens a merging interface, which lets you see exactly what changes were made and choose which to keep.
   - **Undo my local changes** Automatically removes your changes and keeps changes from other users.
   - **Discard external changes** Automatically removes changes from other users and keeps your changes.

5. If you selected the option to use the merge tool, the Merge Changes dialog opens. Use this dialog to view and select changes. You can take actions in the following ways.

   - **Click a change** You can click a change on either the remote or local side. This lets you select a particular change. Use the key at the top of the merge changes dialog, as well as the color coding on the local and server sides, to determine if a change has been added (new), deleted, changed, or moved.When you select a change, the change you selected will display with a solid colored background, and the conflicting change will display with a striped background. If you select the other change, the background shading will switch.
   - **Type content** If you want to use your changes as well as those from another user, and even tweak the paragraph a bit more, you can click in the area at the bottom of the dialog and simply type content.
   - **Previous/next conflict** When you are finished resolving the first conflict, you can use the "Previous Conflict" and "Next Conflict" buttons at the bottom of the dialog to work on other conflicts in the file.

   NOTE If you selected "Merge as Text" in the local toolbar and are working in the code, you can click on text with a hatched background to keep the change in it. After you click on text with a hatched background, the hatched lines are removed, leaving a solid color.

6. After all conflicts have been resolved, a message lets you know that a backup of your local copy has been made. This lets you restore that file if you do not want to keep the merged version. Click **OK**.

☆ **EXAMPLE** – Auto Merging

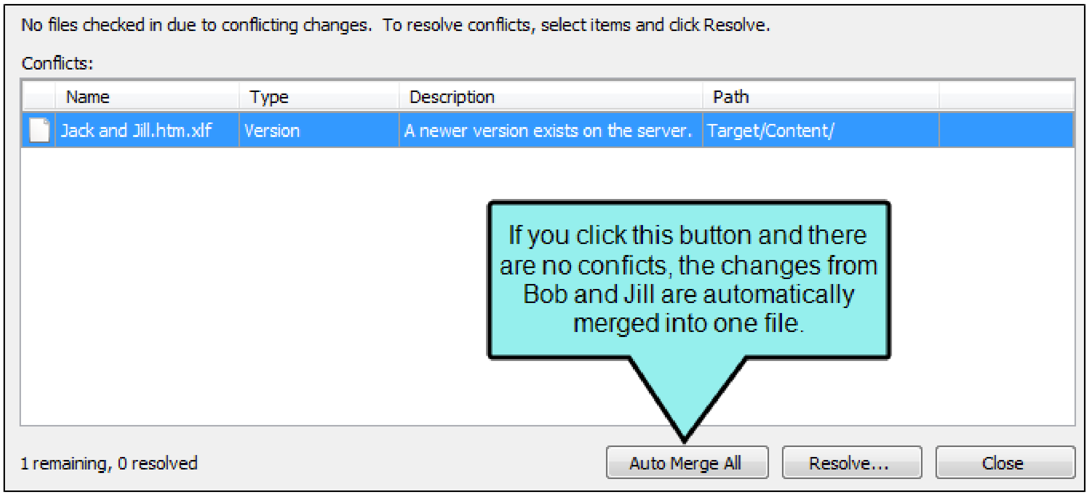
Let's say two translators—Bob and Jill—are working on the same project, using source control to manage the files.

Bob checks out the glossary and starts making changes to it.

Jill also checks out the glossary and makes some changes to it. Jill's changes are in a different segment in the glossary than Bob's changes; there are no conflicts. She finishes before Bob and submits the file to source control.

Bob finishes his changes and tries to submit the file. Before the file can be submitted, Bob is prompted with a dialog, indicating that changes from another user have already been submitted.

Bob selects **Auto Merge All**. The changes from Bob and Jill are now both shown in the merged topic.



No files checked in due to conflicting changes. To resolve conflicts, select items and click Resolve.

Conflicts:

Name	Type	Description	Path
Jack and Jill.htm.xlf	Version	A newer version exists on the server.	Target/Content/

If you click this button and there are no conficts, the changes from Bob and Jill are automatically merged into one file.

1 remaining, 0 resolved

CHAPTER 4

Other Activities for Subversion

In addition to the main activities, there are some other tasks you might perform regarding this feature.

This chapter discusses the following:

Deleting Source Control Files

You can delete a file that is bound to source control. This also removes the file from Subversion.

How to Delete a File

1. In the File List window pane, select the relevant file(s).
2. On your keyboard press **DELETE**.
3. The Delete dialog opens. Select the bound files you want to delete.
4. Click **Delete**. The files are removed from your project and from the source control repository. Files that are deleted from projects bound to Subversion cannot be undeleted.

Disabling the Get Latest Prompt for Source Control

By default, when you open a project that is bound to source control, a message automatically asks if you want to get the latest version of files. However, you can disable this prompt in the Source Control tab of the Options dialog (**File > Options**). Therefore, in the future when you open the project you will no longer see the message, and the project will open without replacing any local files with the latest ones from source control.

How to Disable the Get Latest Prompt for Source Control

1. Select **File > Options**. The Options dialog opens.
2. Select the **Source Control** tab.

   > **NOTE** This tab will not be visible if your project is not yet bound to source control. See "Binding a Project to Subversion" on page 14.

3. Click the check box **Do not prompt to get latest when opening source control bound projects**..
4. Click **OK**.

Disabling a Subversion Provider

By default, when a project is bound to source control, the provider (Git, Perforce Helix Core, Subversion, or Team Foundation Server) is enabled. This means that the source control interface elements in Lingo are visible, and you can use them to perform various tasks (e.g., commits, synchronize changes).

Disabling a provider means that the source control interface elements are no longer shown. This does not mean you cannot use source control. As long as the provider is still *bound* to the project, you can perform source control tasks in a third-party tool outside of Lingo.

How to Disable a Provider

Use this method if you want to disable a provider in just one project, rather than many projects.

1. Do one of the following, depending on the part of the user interface you are using:
    - **Project Properties** Select **File > Project Properties**.
    - **Source Control Explorer** Select **View > Source Control Explorer**. Then, in the window pane, click **Settings**.
2. Click **Enabled** to remove the check mark.
3. If you used the Project Properties dialog, click **OK**.

NOTE If you disable a Git provider, the local repository will continue to track your changes in case you later decide to enable the provider once again.

If you disable one of the other providers (Perforce Helix Core, Subversion, Team Foundation Server), your changes after that point will not be tracked. Therefore, if you later enable the provider again, it will not have recorded any changes made since the time that you disabled it.

NOTE When you disable a provider, that information is written to the registry on your computer.

NOTE If you disable a provider, but then perform one of the following actions in the Lingo interface, the provider will automatically become enabled once again.

- Bind an existing project
- Bind a new project
- Import a project from source control

NOTE Having a provider enabled in Lingo does not interfere with your workflow if you are performing source control actions exclusively outside of Lingo. Even if a provider is enabled in the project and the source control user interface elements are visible, this does not mean Lingo is automatically performing any source control actions with your files. It simply means Lingo is recognizing the binding, so it reflects your activities (e.g., the Pending Changes window is populated when you make edits in topics). However, if you prefer not to see any of this in Lingo, you can disable the provider.

Disconnecting From Source Control

There may be times that you need to disconnect from source control to work offline. You can disconnect from Apache Subversion and reconnect at any time.

Disconnecting from source control is beneficial because it lets you modify files when you would otherwise not have access to the source control system (e.g., you are out of the office with your laptop or you do not want to access source control over VPN). It also provides a fallback offline status in the event that the network is disconnected while you are working, so you are able to continue working on the files you have checked out until the network connection is restored.

How to Disconnect From Subversion

When you want to work offline, you can disconnect from source control.

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Source Control > Disconnect**.
   - **Right-Click** If you have the File List open, right-click on any file and select **Source Control > Project > Disconnect**.
2. A confirmation dialog appears. Click **Yes**. You will be disconnected from source control.

NOTE Because Subversion does not have a "checked out” status, disconnecting from source control lets you work as if your project is unbound. You do not need to check out files before disconnecting.

NOTE When you disconnect from source control, you are not able to see the source control status of files or access source control functions.

NOTE If you make a change to a file’s properties (e.g., delete) while disconnected from source control, your changes may not be preserved when you reconnect to the network.

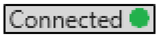
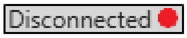
To prevent errors, it is recommended that you do not make these kinds of changes until you reconnect to source control.

How to Reconnect to Subversion

When you are finished working offline, you can reconnect to source control.

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Source Control > Reconnect**.
   - **Right-Click** If you have the File List open, right-click on any file and select **Source Control > Project > Reconnect**.
2. Commit the files you modified. See "Committing Source Control Files" on page 19.

What’s Noteworthy?

NOTE Your current network connection status is indicated in the lower right corner of the Lingo interface. If you are connected you will see  Connected 🟢; if you are disconnected you will see  Disconnected 🔴.

Enabling Source Control Status Checks

If you are using source control integration in Lingo, you can check for frequent status changes automatically. You can specify the number of minutes and seconds when you want Lingo to ping the source control repository and get status changes for files that have been committed, moved, deleted, etc. The upside of this feature is that you can ensure that the source control status information is always up to date. The downside is that you may experience slower performance due to this constant communication over the network.

How to Enable Source Control Status Checks

1. Select **File > Options**. The Options dialog opens.
2. Select the **Source Control** tab.

   > NOTE This tab will not be visible if your project is not yet bound to source control. See "Binding a Project to Subversion" on page 14.

3. Click the check box **Enable background status checks**. A check mark in the box indicates that the feature is enabled.
4. Enter the number of minutes and or seconds between each status update.
5. Click **OK**.

> NOTE See "Viewing Modified Files" on page 57.

Locking a File

When you are working, you may want to lock the files you have modified. Locking a file does not prevent other users from modifying the file. However, no one else can commit a file that you have locked unless they steal the lock from you or until you remove the lock. See "Unlocking a File" on page 50.

You can steal a lock from another user if you need to commit the locked file while they are working on it. Likewise, another user can steal a lock on a file you have locked.

How to Lock a File

1. In the Source Control Explorer or File List, select the relevant file(s).

   OR

   Open a file.
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Lock**.
   - **Right-Click** In the File List, right-click the files you want to lock and select **Source Control > Lock**.
   - **Source Control Explorer** Right-click the files you want to lock and select **Lock**.

   The Lock dialog opens. The selected files are listed with check boxes next to them.
3. (Optional) In the **Comment** field, enter an optional comment tied to the submit. This enables you to keep an audit trail for a file. The comment can then be viewed from the History dialog, which can be accessed from the Source Control Explorer, the Source Control ribbon, or the Source Control button .
4. (Optional) If you want to steal another user's lock so you can commit the file while they are working on it, select **Steal the Lock**. This will give you the lock so you can commit the file.
5. Make sure to select the check box next to each file you want to lock so it contains a check mark.
6. Click **Lock**..

NOTE You cannot lock files with Rename status.

NOTE Subversion will automatically lock a modified file when saving changes (if it is not already locked) if you selected the option to automatically check out files from source control when saving them. Because there is not a "checked out" status for Subversion files, these files will be marked as modified ✓ and can be committed.

Modifying Network Settings

You can view and change various source control network settings while working in Lingo.

How to Modify Network Settings

1. Do one of the following, depending on the part of the user interface you are using:

    - **Ribbon** Select **Source Control > Network Settings**.
    - **Right-Click** If you have the File List open, right-click on any file and select **Source Control > Project > Network Settings**.
    - **Local Toolbar** In the local toolbar of the File List, click ▣, then select **Project > Network Settings**.
    - **Source Control Explorer** From the **View** ribbon, open the Source Control Explorer. From the drop-down, select the **Home Page** view. Click **Network Settings**.

    The Network Settings dialog opens.

2. In the **Group** field, select the group for which you want to change the settings.

    OR

    Do one of the following:

    - (Optional) If you want to add a custom group, type its name in the **Group** field, then click **Add**.
    - (Optional) If you want to remove a group, select it from the **Group** field, then click **Remove**.

3. (Optional) If you entered a custom group name, in the **Remote Host/Pattern** field, enter the name of the domain to which the network settings should apply.

    NOTE If you are using Subversion, you can enter wildcards in this field.

4. In the grid, make changes to the network settings as necessary. Click ☰ to sort the settings by category, or click AZ↓ to sort them alphabetically.

SUBVERSION NETWORK SETTINGS

- **Advanced**
  - Chunk Requests
  - HTTP Bulk Update
  - HTTP Compression
  - Maximum Connections
- **Caching Options**
  - Store Authentication Info
  - Store Passwords
  - Store Plaintext Passwords
  - Store Plaintext SLL Client Cert Passphrase
  - Store SSL Client Cert Passphrase
- **Global Options**
  - Default User Name
- **HTTP Proxy Options**
  - Proxy Host
  - Proxy Host Port
  - Proxy Password
  - Proxy Timeout
  - Proxy User Name
  - Site Exceptions

- **SSL Options**
    - SSL Authority Files
    - SSL Client Certificate File
    - SSL Client Certificate Password
    - SSL Trust Default CAs

NOTE For more information about each of these settings, refer to:

```
http://svnbook.red-bean.com/en/1.7/svn.advanced.confarea.html
```

5. Click **Save**.

Reverting Modified Source Control Files

If you have modified files from source control but do not want to keep your modifications, you can use the "Revert" option instead of committing the files. While committing the file would save your changes to source control, reverting a file returns it to its previously committed state and does not commit any of your new changes to source control.

How to Revert a Source Control File

1. In the Source Control Explorer or File List, select the relevant file(s).
2. Do one of the following, depending on the part of the user interface you are using:

    - **Ribbon** Select **Source Control > Revert** (for selected files) or **Source Control > Revert All** (for all files in the project).
    - **Right-Click** In the File List, right-click the files you want to revert and select **Source Control > Revert** (for selected files) or **Source Control > Project > Revert All** (for all files in the project).

    The Revert dialog opens. The selected files are listed with check boxes next to them.
3. (Optional) If you want to see all files with pending changes (rather than only those you selected), click 🗐.
4. Make sure to click the check box next to each appropriate file so that it contains a check mark.
5. Click **Revert**.

Rolling Back to an Earlier Version of a File

One of the benefits of Lingo's integrated source control is that you can view the history and differences for a particular file. You can view code and content differences between two source control versions of the same file. This is useful if you need to roll back to an earlier version of a file.

See "Viewing Differences in Source Control Files" on page 51 and "Viewing the History of Source Control Files" on page 55.

☆ **EXAMPLE** You have been working on translating a particular topic for a few days. Each day you modify that topic file, make your changes, and commit the file back to the server at the end of the day. At a certain point, you determine that you need to "roll back" to an earlier version of the file, using it to replace the latest version. Therefore, you use this feature to view the highlighted differences between the current version and an older version of the file. Once you have identified the older version that you want to use, you can retrieve that version.

How to Roll Back to an Earlier Version of a File

1. In the Source Control Explorer or File List, select the relevant file(s).

   OR

   Open a file.

2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > View History**.
   - **Right-Click** If you have the File List open, right-click the file you want to roll back and select **Source Control > View History**.
   - **Source Control Explorer** With the Pending Changes pane open, right-click the file you want to roll back and select **View History**.

   The History dialog opens.

3. (Optional) View the differences between two versions of the file. This may help you decide which version of the file you want to retrieve. (Another way is to look at the dates for each version in the History dialog.)

   To view the differences, follow these steps.

   a. Select the first file version from the list.

   b. Hold the **CTRL** key and select the second file version from the list.

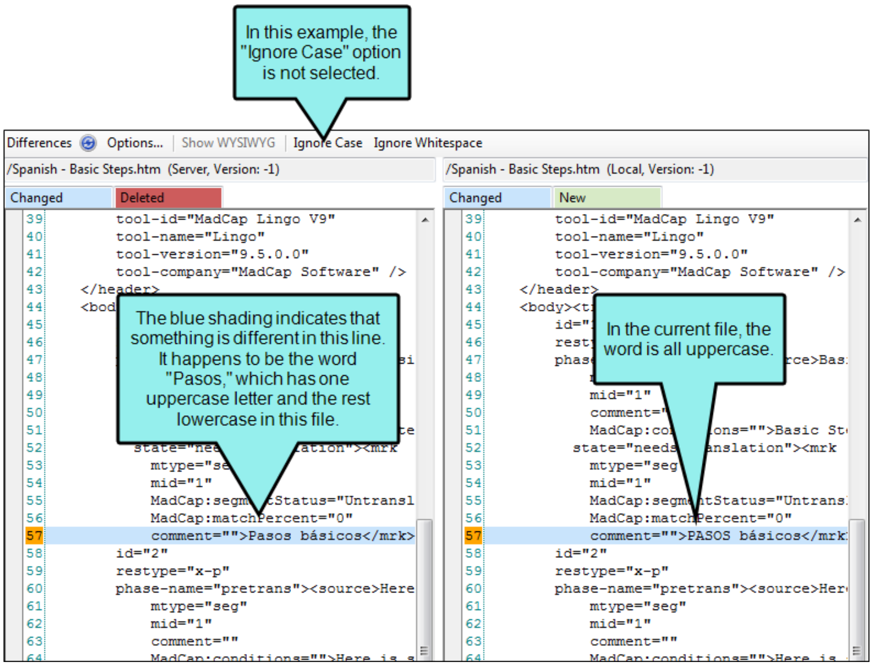
   c. Click **Show Differences**. The Differences Editor opens to the right, showing content from the backup file on the left and the current version of the file on the right.

   d. In the local toolbar of the Differences Editor, you can click any of the following to make adjustments.

      - **Options** Click this to open the File Differences dialog, which lets you change the colors used to display content differences between the files.

- **Ignore Case** Click this to ignore changes in case when viewing differences.

EXAMPLE If a word is not capitalized in the original file but it is capitalized in the current file, this option does not highlight those differences.



In this example, the "Ignore Case" option is not selected.

/Spanish - Basic Steps.htm (Server, Version: -1)

```
        tool-id="MadCap Lingo V9"
        tool-name="Lingo"
        tool-version="9.5.0.0"
        tool-company="MadCap Software" />
    </header>
    <bod

                                      si



                                      te
          state="nee          ation"><mrk
            mtype="se
            mid="1"
            MadCap:segm  tStatus="Untransl
            MadCap:matchPercent="0"
            comment="">Pasos básicos</mrk>
        id="2"
        restype="x-p"
        phase-name="pretrans"><source>Here
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Here is s
```

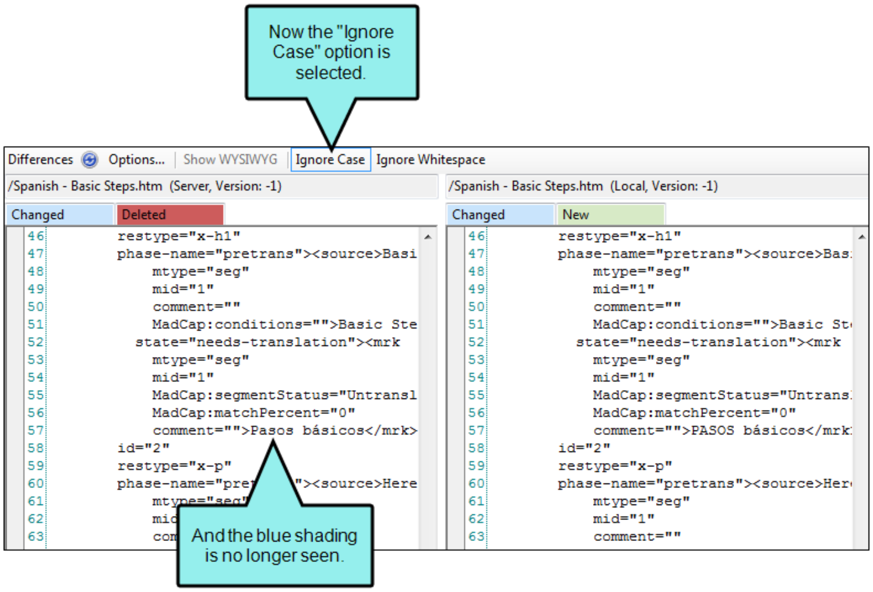
The blue shading indicates that something is different in this line. It happens to be the word "Pasos," which has one uppercase letter and the rest lowercase in this file.

/Spanish - Basic Steps.htm (Local, Version: -1)

```
        tool-id="MadCap Lingo V9"
        tool-name="Lingo"
        tool-version="9.5.0.0"
        tool-company="MadCap Software" />
    </header>
    <body><t
        id="
        rest
        phas                         rce>Bas

            mid="1"
            comment="
            MadCap:co    ons="">Basic St
          state="needs  anslation"><mrk
            mtype="seg
            mid="1"
            MadCap:segm  tStatus="Untrans
            MadCap:matchPercent="0"
            comment="">PASOS básicos</mrk
        id="2"
        restype="x-p"
        phase-name="pretrans"><source>Her
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Here is
```

In the current file, the word is all uppercase.



Now the "Ignore Case" option is selected.

/Spanish - Basic Steps.htm (Server, Version: -1)

```
        restype="x-h1"
        phase-name="pretrans"><source>Basi
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic Ste
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untransl
            MadCap:matchPercent="0"
            comment="">Pasos básicos</mrk>
        id="2"
        restype="x-p"
        phase-name="pre        "><source>Here
            mtype="seg"
            mic
            com
```

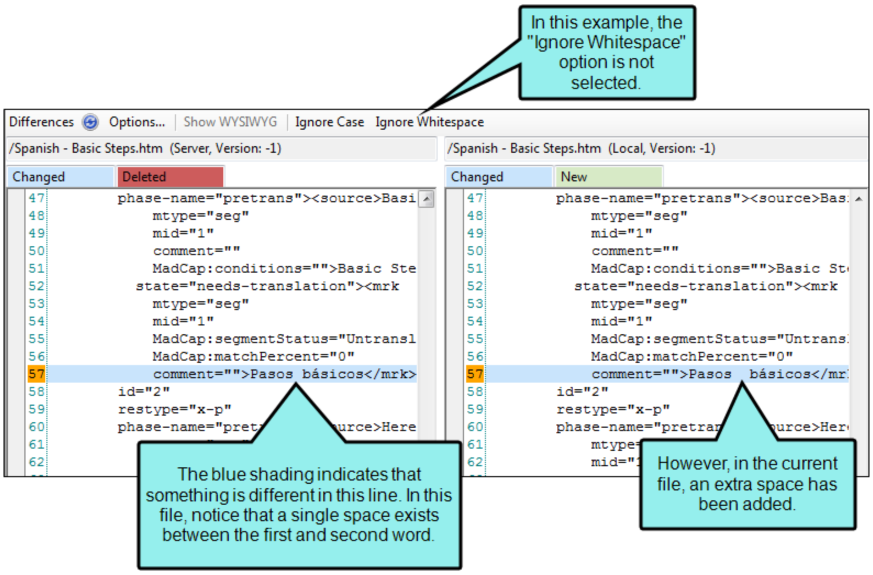
/Spanish - Basic Steps.htm (Local, Version: -1)

```
        restype="x-h1"
        phase-name="pretrans"><source>Bas
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic St
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untrans
            MadCap:matchPercent="0"
            comment="">PASOS básicos</mrk
        id="2"
        restype="x-p"
        phase-name="pretrans"><source>Her
            mtype="seg"
            mid="1"
            comment=""
```

And the blue shading is no longer seen.

- **Ignore Whitespace** Click this to ignore whitespace when viewing differences.

EXAMPLE A segment is identical in both files, except for an extra space that was added within the segment in one of those files. If you click this option, that difference is not highlighted.



In this example, the "Ignore Whitespace" option is not selected.

/Spanish - Basic Steps.htm (Server, Version: -1)

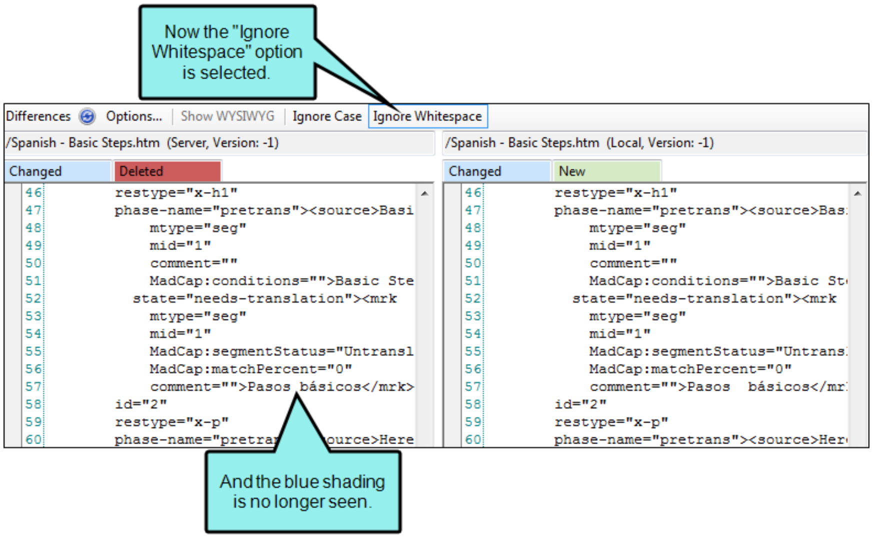
```
        phase-name="pretrans"><source>Basi
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic Ste
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untransl
            MadCap:matchPercent="0"
            comment="">Pasos básicos</mrk>
        id="2"
        restype="x-p"
        phase-name="pret
```

/Spanish - Basic Steps.htm (Local, Version: -1)

```
        phase-name="pretrans"><source>Bas
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic St
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untrans
            MadCap:matchPercent="0"
            comment="">Pasos  básicos</mr
        id="2"
        restype="x-p"
        phase-name="pretran
            mtype=
            mid="1
```

The blue shading indicates that something is different in this line. In this file, notice that a single space exists between the first and second word.

However, in the current file, an extra space has been added.



Now the "Ignore Whitespace" option is selected.

/Spanish - Basic Steps.htm (Server, Version: -1)

```
        restype="x-h1"
        phase-name="pretrans"><source>Basi
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic Ste
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untransl
            MadCap:matchPercent="0"
            comment="">Pasos básicos</mrk>
        id="2"
        restype="x-p"
        phase-name="pretra          ource>Here
```

/Spanish - Basic Steps.htm (Local, Version: -1)

```
        restype="x-h1"
        phase-name="pretrans"><source>Bas
            mtype="seg"
            mid="1"
            comment=""
            MadCap:conditions="">Basic St
          state="needs-translation"><mrk
            mtype="seg"
            mid="1"
            MadCap:segmentStatus="Untrans
            MadCap:matchPercent="0"
            comment="">Pasos  básicos</mr
        id="2"
        restype="x-p"
        phase-name="pretrans"><source>Her
```

And the blue shading is no longer seen.

e. When you are finished viewing the differences, close the window pane.

4. In the History dialog, select the version of the file to which you want to roll back.
5. Click **Get Selected Version**. That file is retrieved from the server and replaces the local copy of the file in your project.
6. In the History dialog, click **Close**.

Setting Color Options for Project File Differences

If you are using Lingo's integrated source control features, you can view differences between files in various ways. One way is to view file differences between a local version of a Lingo project and the source control version.

When viewing file differences between a local version of a Lingo project and the source control version, you can select color options to display the files. Color coding makes it easier to discern where differences between files occur.

For more information see "Viewing Differences in Source Control Files" on page 51.

☆ **EXAMPLE** By default the files that are included only in your local copy are displayed as green in the Differences Editor, and the files that are included only in source control are displayed in red. You can use this dialog to change the local-only files to blue and the source control-only files to yellow.

How to Set Color Options for Project File Differences

1. In the Source Control Explorer or File List, select the relevant file(s).
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Show Differences**.
   - **Right-Click** If you have the Source Control Explorer or File List open, right-click the file you want to view and select **Source Control > Show Differences**.

   The Differences Editor opens.
3. In the local toolbar of the Differences Editor, click **Options**. The File Differences dialog opens.

4. Change the text or background color for any of the difference types. To do this, simply click in the appropriate **Text** or **Background** cell and select **Pick Color**. In the Color Picker dialog, choose the new color.

5. Click **OK**.

Unbinding a Subversion Provider

When you unbind a provider, it means you are removing the connection altogether between the Lingo project and the local repository.

How to Unbind Using the Project Properties Dialog

1. Open the project.
2. Select **File > Project Properties**.
3. Select the **Source Control** tab.
4. Click **Unbind Provider**. (If your project is dual-bound, you will also see a section for the other binding.)
5. Click **OK**.

How to Unbind Using the Source Control Explorer

1. Open the project.
2. Select **View > Source Control Explorer**.
3. From the drop-down or the Home pane, select **Settings**. The Settings pane opens.
4. Click **Unbind Provider**. (If your project is dual-bound, you will also see a section for the other binding.)
5. Click **Yes**.

What’s Noteworthy?

NOTE You can also disable a provider, which retains the binding but hides source control elements from the user interface.

Unlocking a File

If you have locked a file, you should unlock it when you are done modifying it. Other users can modify the file while you have it locked, but they cannot commit a locked file until you unlock the file or unless they steal the lock from you. To help prevent file conflicts and make sure that everyone on your team has the most current version of the file, you should unlock and commit the file when you are finished working on it.

See "Locking a File" on page 34 for more information on locking files and stealing locks.

How to Unlock a File

1. In the Source Control Explorer or File List, select the relevant file(s).

   OR

   Open a file.

2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Unlock**.
   - **Right-Click** In the File List, right-click the files you want to unlock and select **Source Control > Unlock**.
   - **Source Control Explorer** Right-click the files you want to unlock and select **Unlock**.

   The Unlock dialog opens. The selected files are listed with check boxes next to them.

3. Make sure to select the check box next to each file you want to unlock so it contains a check mark.

4. Click **Unlock**.

Viewing Differences in Source Control Files

One of the benefits of Lingo's integrated source control is that you can view the history and differences for a particular file.

Ways to View Differences Between Files

You can view differences between files in the following ways:

- **Two Versions of Same Source Control File (History/Roll Back)** You can view code and content differences between two source control versions of the same file. This is useful if you need to roll back to an earlier version of a file. See "Rolling Back to an Earlier Version of a File" on page 40 and "Updating Source Control Files" on page 18.

  ☆ EXAMPLE You have been working on translating a particular topic for a few days. Each day you modify that topic file, make your changes, and commit the file back to the server at the end of the day. At a certain point, you determine that you need to "roll back" to an earlier version of the file, using it to replace the latest version. Therefore, you use this feature to view the highlighted differences between the current version and an older version of the file. Once you have identified the older version that you want to use, you can retrieve that version.

- **Local Versus Source Control Version of a File** You can view code and content differences between the local version of a file and the source control version of that file.

  ☆ EXAMPLE You modify a topic from source control and then make changes to some of the translations in your local copy of that file. You save your changes. Later that day, you want to revisit the new translations, but you cannot remember exactly which translations you added and which were there before. Therefore, you use this feature to highlight the text differences between your local version of the file and the version stored in the source control application. The new text is highlighted on the side displaying the local version of the file.

- **Local Versus Source Control Version of All Files in a Folder** You can view file differences between the local version of the files in a folder and the source control version. Most likely, you will use this to view all of the difference between the local files and source control files in a specific content folder.

  > ☆ **EXAMPLE** You are working on a large Lingo project. During the course of the day, you edit several files in the project. At the end of the day, you commit most of your files, but forget to commit a few of them. Later, you realize that you missed a few files, and now those files are out-of-date. Therefore, you use this feature to see the file-level differences between your local copy of the folder in the File List and the source control copy. The differences are color coded, so you can easily identify the files in question. (By default, the files that are included only in your local copy are green, and the files that are included only in source control are red.)

How to View Differences Between Two Versions of the Same Source Control File

1. In the Source Control Explorer or File List, select the relevant file(s).
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > View History**.
   - **Right-Click** If you have the Source Control Explorer or File List open, right-click the file (s) you want to view and select **Source Control > View History**.

   The History dialog opens.
3. From the list, select the first file version that you want to compare.
4. Hold the **CTRL** key and select the second file version from the list.
5. Select **Show Differences**.The Differences Editor opens.
6. (Optional) In the Differences Editor, use the buttons in the local toolbar to customize the information shown in the editor.
7. When you are finished viewing the differences, close the window.

How to View Differences Between the Local Version of a File and the Source Control Version

1. In the Source Control Explorer or File List, select the relevant file(s).
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Show Differences**.
   - **Right-Click** If you have the Source Control Explorer or File List open, right-click the file (s) you want to view and select **Source Control > Show Differences**.

   The Differences Editor opens.
3. (Optional) In the Differences Editor, use the buttons in the local toolbar to customize the information shown in the editor.
4. When you are finished viewing the differences, close the window.

How to View Differences Between the Local Version of All Files in a Folder and the Source Control Version

1. In the File List, select the relevant folder.
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > Show Differences**.
   - **Right-Click** Right-click the folder you want to view and select **Source Control > Show Differences**.

   The Differences Options dialog opens.
3. (Optional) Use this dialog to specify the type of information that you want to see in the Differences Editor. (You can also choose these options from the local toolbar of the Differences Editor.)

   - **Show files that are only in the Left folder** Displays the files on the left side of the Differences Editor. The left side is used to show the local copies of your project files.
   - **Show files that are only in the Right folder** Displays the files on the right side of the Differences Editor. The right side is used to show the source control copies of your project files.
   - **Show files that are different in both folders** Displays the files where differences occur between the local copy and source control copy of the project. For example, the left side might display files that you have created in your local copy but have not yet been added to source control.
   - **Show files that are the same in both folders** Displays the files that are the same in the local copy as they are in the source control copy.
   - **Recursive** Displays files recursively. In other words, if you have files contained within folders, selecting this button will ensure that you see all of the files, not just the folders.
4. Click **OK**. The Differences Editor opens.
5. (Optional) In the Differences Editor, use the buttons in the local toolbar to customize the information shown in the editor.

Viewing the History of Source Control Files

One of the benefits of Lingo's integrated source control is that you can view the history for a particular file, including who committed the file and when it was committed. You can also view differences between different versions of the file and roll back to an older version if necessary.

For more information see "Viewing Differences in Source Control Files" on page 51 and "Rolling Back to an Earlier Version of a File" on page 40.

How to View the History of a Source Control File

1. In the Source Control Explorer or File List, select the relevant file(s).

   OR

   Open a file.
2. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Source Control > View History**.
   - **Right-Click** If you have the Source Control Explorer or File List open, right-click the file you want to view and select **Source Control > View History**.
3. The History dialog opens. The following are explanations of the different parts of this dialog.

   - **Version** Displays a number for each version of the file. The lower the number, the older the version. The higher the number, the more recent the version.
   - **Users** Displays the name of the user who has been working on the file.
   - **Date** Displays the date and time when the action has occurred.
   - **Action** Displays the action that has taken place for the file (e.g., commit).
   - **Comment** Displays the comment (if any) associated with the file. A comment can be added to a file when you commit that file to source control. This enables you to maintain an audit trail for the file's history.

- **Get Selected Version** Retrieves a particular version of a file, thus rolling back to that version of the file. The local version of the file is replaced with the source control version that you selected.
- **Show Differences** Opens a dialog that lets you view the differences between two versions of a file. If you select one row in the History dialog and view the differences, you will see the content differences between the version that you selected and the version of the file in your local copy of the Lingo project. If you select two files in the History dialog (by holding down the CTRL key) and view the differences, you will see the content differences between those two versions of the file.

4. In the History dialog, click **Close**.

> **NOTE** When you modify a file in source control, you are actually modifying the file's corresponding XLF file. You will see the XLF file if you open the Source Control Explorer. This is because you need the XLF file available in order to make changes, view the file's history, or view differences.

Viewing Modified Files

You can use the Source Control Explorer to view all of the files that you have modified and need to commit.

How to View Files That You and Others Have Modified—Pending Changes Window Pane

1. Select **Source Control > Pending Changes**. The Pending Changes window pane opens.
2. (Optional) You can use the **Filter** field to limit the files that are displayed.
    - **All Files** Displays all translatable files.
    - **Topic Files** Displays only the topic (HTM and HTML) files.
    - **Page Layout Files** Displays only the page layout (FLPGL) files.
    - **Snippet Files** Displays only the snippet (FLSNP) files.
    - **Stylesheet Files** Displays only the stylesheet (CSS) files.
    - **Template Page Files** Displays only the template page (FLMSP) files.
    - **Image Files** Displays only the image files.
3. (Optional) You can use the following toggle buttons in the local toolbar to limit the files that are displayed.

	This filters the Pending Changes window pane to show or not show files that other users have modified.
	This filters the Pending Changes window pane to show or not show files that are out of date.
	This filters the Pending Changes window pane to show or not show files that have been deleted.

This filters the Pending Changes window pane to show or not show files that are locked.

4. Take note of the **Status** and **User** columns. (You may need to use the scroll bar to view these columns.)

   - **Status** Displays the status of the file, such as whether you have modified it. You can also see if another user has modified or locked a file.

     - **Modified** This indicates that the file has been modified. You can commit the file when you are ready (if you are the user who has modified it, or if you have stolen the lock on the file from another user).

     - **Add** This indicates that you have a file in your project but have not yet added it to Subversion. This might occur, for example, if you create a new topic and do not add the file to source control during the topic creation process. To resolve this, simply right-click on the file and select **Source Control > Add**.

     - **Out of Date** This indicates that the file is not current (i.e., the local copy of the file is older than the source control copy). This might happen, for example, if another user modifies the file and commits it to source control. If this occurs, you can modify the file or update the file from source control.

     - **Locked** This indicates that the file has been locked by you or another user. Any user can modify the file even if it has been locked. However, a user cannot commit a file that another user has locked unless they steal the lock first.

     - **Renamed** This indicates that a file has been renamed, but not modified in any other way.

   > **NOTE** You can click the refresh button in the local toolbar to make sure you have the most recent status for each file. Another option is that you can use a feature to automatically ping the source control repository periodically, thus refreshing this information frequently. However, you may experience slower performance with this automatic status update option set. See "Enabling Source Control Status Checks" on page 33.

   - **User** Displays the user name. If you see the name of another user in this column, it means that the file has been modified by that user.

How to View Files That Others Have Modified–File List Window Pane

1. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **View > File List**.
   - **Keyboard Shortcut** Press **CTRL+SHIFT+J**.

   The File List window pane opens.

2. (Optional) You can use the **Filter** field to limit the files that are displayed.

   - **All Files** Displays all translatable files.
   - **Topic Files** Displays only the topic (HTM and HTML) files.
   - **Page Layout Files** Displays only the page layout (FLPGL) files.
   - **Snippet Files** Displays only the snippet (FLSNP) files.
   - **Stylesheet Files** Displays only the stylesheet (CSS) files.
   - **Template Page Files** Displays only the template page (FLMSP) files.
   - **Image Files** Displays only the image files.

3. Take note of the **Status** and **User** columns. (You may need to use the scroll bar to view these columns.)

   - **Status** Displays the status of the file, such as whether you have modified it. You can also see if another user has modified or locked a file.

     > **NOTE** You can click the refresh button in the local toolbar to make sure you have the most recent status for each file. Another option is that you can use a feature to automatically ping the source control repository periodically, thus refreshing this information frequently. However, you may experience slower performance with this automatic status update option set. See "Enabling Source Control Status Checks" on page 33.

   - **User** Displays the user name. If you see the name of another user in this column, it means that the file has been modified by that user.

How to View Files That You Have Modified–Source Control Explorer

1. Select **View > Source Control Explorer**. The Source Control Explorer opens.
2. From the drop-down or the Home pane, select **Pending Changes**.

   The Pending Changes pane opens. Files that you have changed appear in the **Included Changes** or **Excluded Changes** section (depending on whether you are going to include or exclude them in your next commit). You will not see other users' changes in the Source Control Explorer.
3. Take note of the file's status. The status is written in brackets next to the file name (e.g., modified, add).

   > NOTE You can click the refresh navigation button in the Source Control Explorer to make sure you have the most recent status for each file. Another option is that you can use a feature to automatically ping the source control repository periodically, thus refreshing this information frequently. However, you may experience slower performance with this automatic status update option set. See "Enabling Source Control Status Checks" on page 33.

> NOTE When you modify a file in source control, you are actually modifying the file's corresponding XLF file. You will see the XLF file if you open the Source Control Explorer. This is because you need the XLF file available in order to make changes, view the file's history, or view differences.

> NOTE When you modify a file in source control, you may sometimes see a SKL file alongside the XLF and original files. This skeleton file is a placeholder file. Be sure to check in all three associated files together. Checking in just the SKL file may result in errors.

PDFs

The following PDFs are available for download from the online Help.

Cheat Sheets

Shortcuts Cheat Sheet

User Guides

Alignment Guide

Getting Started Guide

Key Features Guide

Source Control Guide: Git

Source Control Guide: Perforce Helix Core

Source Control Guide: Subversion

Source Control Guide: Team Foundation Server

Termbases Guide

Touring the Workspace Guide

Translation Guide

What's New Guide