AccuRev

AccuRev Plug-In for IntelliJ® IDEA

User's Guide

Version 2014.2

Revised 17-December-2014

Copyright

This product incorporates technology that may be covered by one or more of the following patents: U.S. Patent Numbers: 7,437,722; 7,614,038; 8,341,590; 8,473,893; 8,548,967.

AccuRev, **AgileCycle**, and **TimeSafe** are registered trademarks of AccuRev, Inc.

AccuBridge, **AccuReplica**, **AccuSync**, **AccuWork**, **Kando**, and **StreamBrowser** are trademarks of AccuRev, Inc.

Table of Contents

1. Introduction 1

AccuRev Enhancements to the IDE 1

AccuRev Status Indicators 1

AccuRev Commands 1

AccuRev Console Subwindow 2

AccuRev Searches/Status Subwindow 3

AccuRev Element History 3

Checkout from Version Control for AccuRev 3

2. Getting Started 5

Establishing Your Identity 5

Placing Projects Under Version Control 6

Making AccuRev the Default Version Control Provider for New IDE Projects 6

Making AccuRev the Version Control Provider for an Existing IDE Project 7

Creating a New Workspace for Sources that are Already Under AccuRev Version Control 7

Creating a New IDE Project in a Workspace 10

Converting (external)-Status Files to Version-Controlled Elements 11

3. Executing AccuRev Commands 13

AccuRev Command Reference 13

Login 13

Add to AccuRev Depot 13

Keep 14

Anchor 14

Promote 15

Implicit Keep and Promote 15

Merge 15

Version Browser 15

Annotate 16

Defunct 16

Populate 17

Revert To 17

Diff Against 17

Synchronize Time 18

AccuRev Workspace Information 18

Update AccuRev Workspace 18

AccuRev History 19

AccuRev Statuses 21

Icon Decorations in the Project Tool 21

Searches 22

Refresh 23

Web UI 23

Send To Issue 24

Remove From Issue 24

Add To Ignore 24
Running Commands in the Background 25
IDE 'Namespace' Commands and AccuRev 25
Using the IDE's Delete Command 26

4. Day-to-Day Usage of AccuRev® 27
The AccuRev® Usage Model 27
Change and Synchronization: The Four Basic Commands 28
Keep: Preserving Changes in Your Private Workspace 28
More About Keep 29
Promote: Making Your Private Changes Public 30
Streams 31
Promotion and Parallel Development 31
Active and Inactive Files 31
More About Promote 32
Update: Incorporating Others' Changes into Your Workspace 32
More About Update 33
Merge: When Changes Would Collide 33
More About Merge 35

1. Introduction

This document describes AccuRev Plug-In for IntelliJ® IDEA, the integration of AccuRev with JetBrains's IntelliJ IDEA Integrated Development Environment (IDE). The integration is implemented as a standard plug-in, enabling IntelliJ users to access AccuRev version-control facilities using IntelliJ IDEA's own menu structure.

> **Note:** Prior to Release 2014.1, AccuRev Plug-In for IntelliJ IDEA was called AccuBridge for IntelliJ IDEA.

AccuRev Enhancements to the IDE

AccuRev Plug-In for IntelliJ IDEA extends the IDE in many ways, as described in the sections that follow.

AccuRev Status Indicators

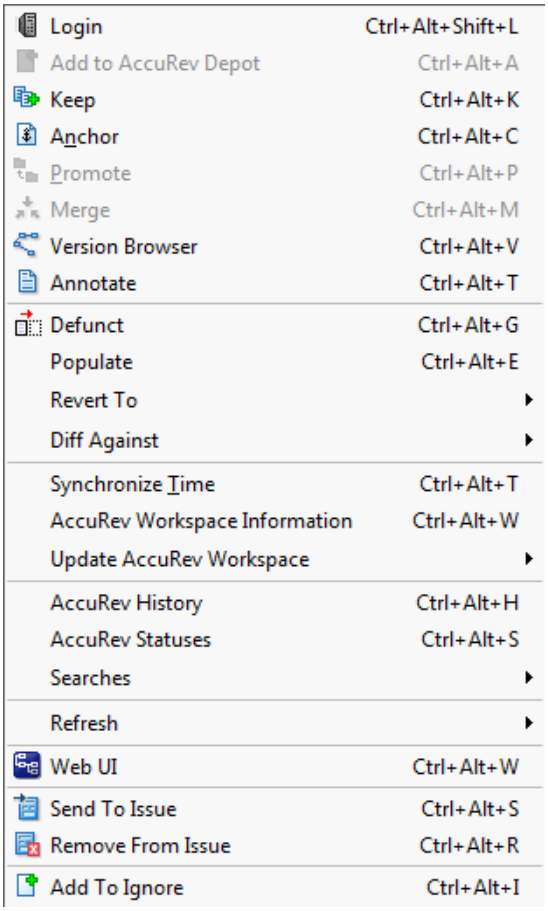
Files managed by AccuRev acquire icon decorations in the Project tool subwindow list showing their AccuRev status. Icons for directories are not decorated, even though they are under AccuRev version control, as well.

For a listing of icon decorations, see *Icon Decorations in the Project Tool* on page 21.

AccuRev Commands

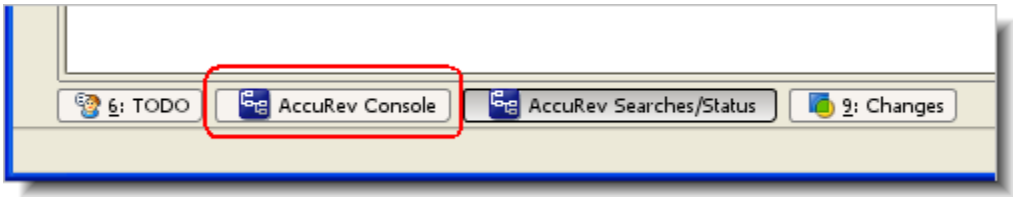
You can invoke a substantial number of AccuRev's version-control commands from context (right-click) menus and from the **Version Control > AccuRev** menu. For a complete reference to the commands available in Accurev Plug-In for IntelliJ IDEA, see *Executing AccuRev Commands* on page 13.

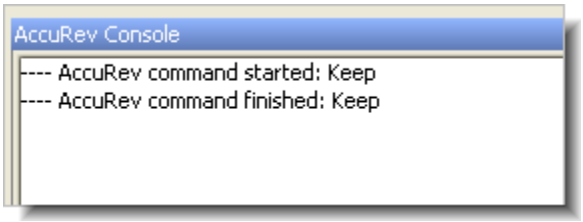
When you right click on an element and select AccuRev, the following context menu appears:



AccuRev Console Subwindow

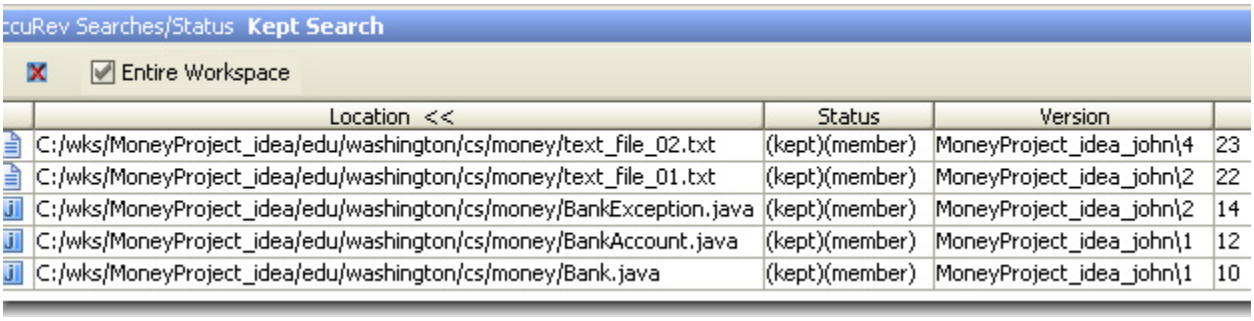
Many AccuRev commands generate messages, which are displayed in the AccuRev Console tool subwindow. This subwindow is hidden by default. To open or close it, click its button at the bottom of the IDE window, or invoke the command **Window > Tool Windows > AccuRev Console**.



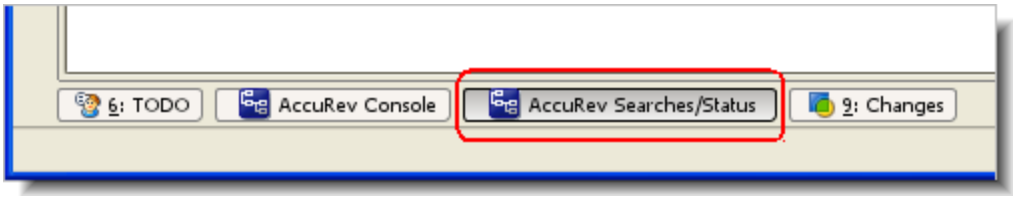


AccuRev Searches/Status Subwindow

The **AccuRev Statuses** command and the various forms of the **AccuRev Searches** command display their results in the AccuRev Searches/Status tool subwindow:

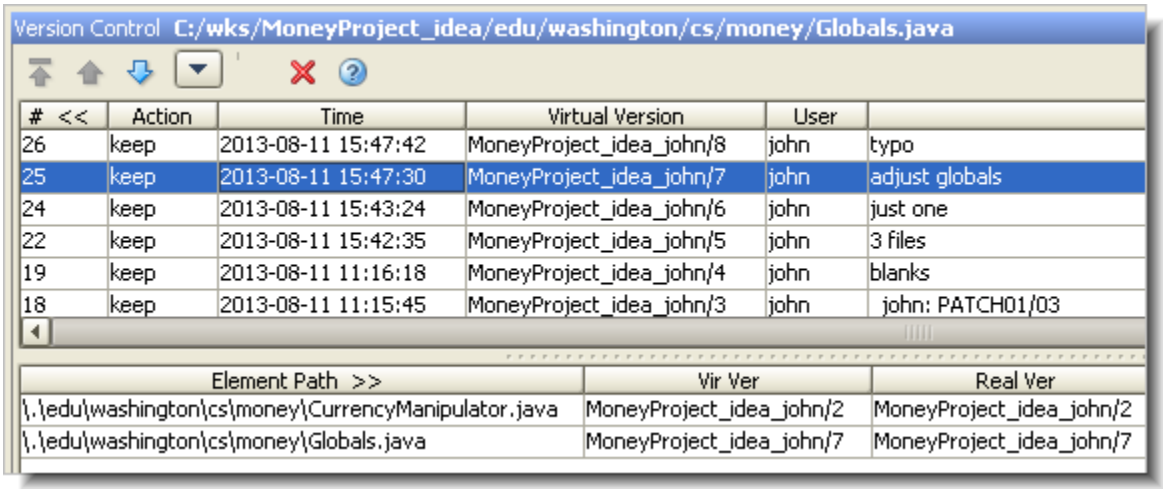


To open and close this subwindow, click its button at the bottom of the IDE window, or invoke the command **Window > Tool Windows > AccuRev Searches/Status**.



AccuRev Element History

The **AccuRev History** command displays the set of transactions involving a selected element in the Version Control subwindow.



Checkout from Version Control for AccuRev

The command **Version Control > Checkout from Version Control > AccuRev** creates a new AccuRev workspace, populating it with files already under version control in a particular AccuRev depot. Then, it converts the workspace into an IntelliJ IDEA project.

2. Getting Started

This chapter describes tasks that get you up and running with AccuRev Plug-In for IntelliJ IDEA.

Establishing Your Identity

All AccuRev commands must be executed by an AccuRev user. To use AccuRev Plug-In for IntelliJ IDEA, you must establish your identity to AccuRev. AccuRev has two schemes for authenticating users:

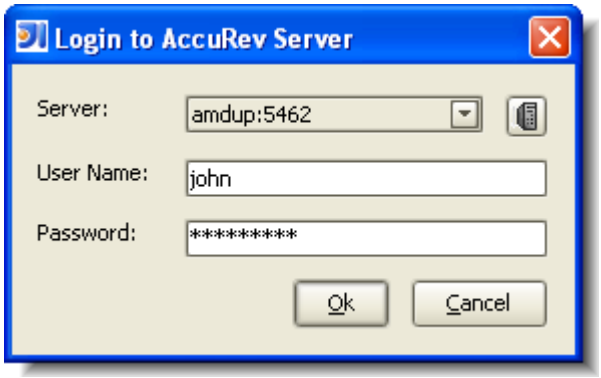
- With the *traditional* user-authentication scheme, AccuRev defaults to using your operating-system username as your AccuRev username. Your AccuRev password must be stored in file **authn**, in the **.accurev** subdirectory of your operating-system home directory. (On Windows, this is %HOMEDRIVE%\%HOMEPATH%.)

  To customize this behavior, set either or both of the following environment variables before starting the IDE (Windows users should set user-level, not system-level variables):

  - To use a different username, set ACCUREV_PRINCIPAL to that username.
  - To place your **.accurev** directory elsewhere, set ACCUREV_HOME to the new location.

- With the *accurev_login* user-authentication scheme, you must perform an explicit login at the AccuRev level. You can do this before starting IntelliJ IDEA. When you create a workspace in the IDE, you are prompted to log in if you aren't already logged in. When working within an IDE project, you can use the **AccuRev > Login** command to log in. It's available on context menus in the Project tool subwindow or under **VCS** on the main menu.

  In the Login dialog box, select an AccuRev Server, and enter a username/password pair. Then click **Ok**.



  See the *AccuRev Administrator's Guide* for complete information on AccuRev user management.

Placing Projects Under Version Control

AccuRev Plug-In for IntelliJ IDEA provides access to AccuRev version control commands within IntelliJ IDEA. The commands executed by the Integration move data between a central source-code repository (a depot) and a personal AccuRev work area (a workspace). AccuRev can manage an IDE project's files if (and only if) the project's location on disk is within an AccuRev workspace. It doesn't matter which is created first — the IDE project or the AccuRev workspace.

Also relevant is this aspect of AccuRev's data architecture: AccuRev does not automatically version-control every file within an AccuRev workspace. For example, version-controlling text-editor backup files would be a waste of resources. Files created in a workspace initially have **(external)** status. They become version-controlled elements when explicitly processed with the **Add to AccuRev Depot** command.

Given these facts, it makes sense to consider the following use cases. Before performing any of the procedures, make sure that AccuRev depots and streams have been created and properly configured. These operations cannot be performed within IntelliJ IDEA.

- **Case 1:** An IntelliJ IDEA project already exists; you want to make AccuRev its version control provider.

  Perform the procedure in section *Making AccuRev the Version Control Provider for an Existing IDE Project*.

- **Case 2:** Your sources are already under AccuRev version control; you want to use IntelliJ IDEA for development.

  If you have access to an AccuRev workspace containing the sources, perform the procedure in *Creating a New IDE Project in a Workspace*.

  If you need to create a new AccuRev workspace containing the sources, perform the procedure in *Creating a New Workspace for Sources that are Already Under AccuRev Version Control*.

- **Case 3:** Your sources are located in an AccuRev workspace; an IntelliJ IDEA project exists for the sources.

  Invoke the IDE's **File > Open Project** command, navigate to the project file (**.ipr**), and click **OK**.

Making AccuRev the Default Version Control Provider for New IDE Projects

After you perform this procedure, AccuRev version control will be enabled automatically for any project created with the **New Project** command, as long as the specified "project file location" for the new project is within an existing AccuRev workspace.

1. Close any open projects.
2. Select **File > Settings** from the IntelliJ IDEA main menu.
3. Expand the **Version Control** node.
4. In the **Version Control** list box, select **AccuRev**.

Making AccuRev the Version Control Provider for an Existing IDE Project

You might want to use AccuRev selectively — for some new projects but not others. Or you might have existing projects to be used with AccuRev. This procedure handles both of these cases:

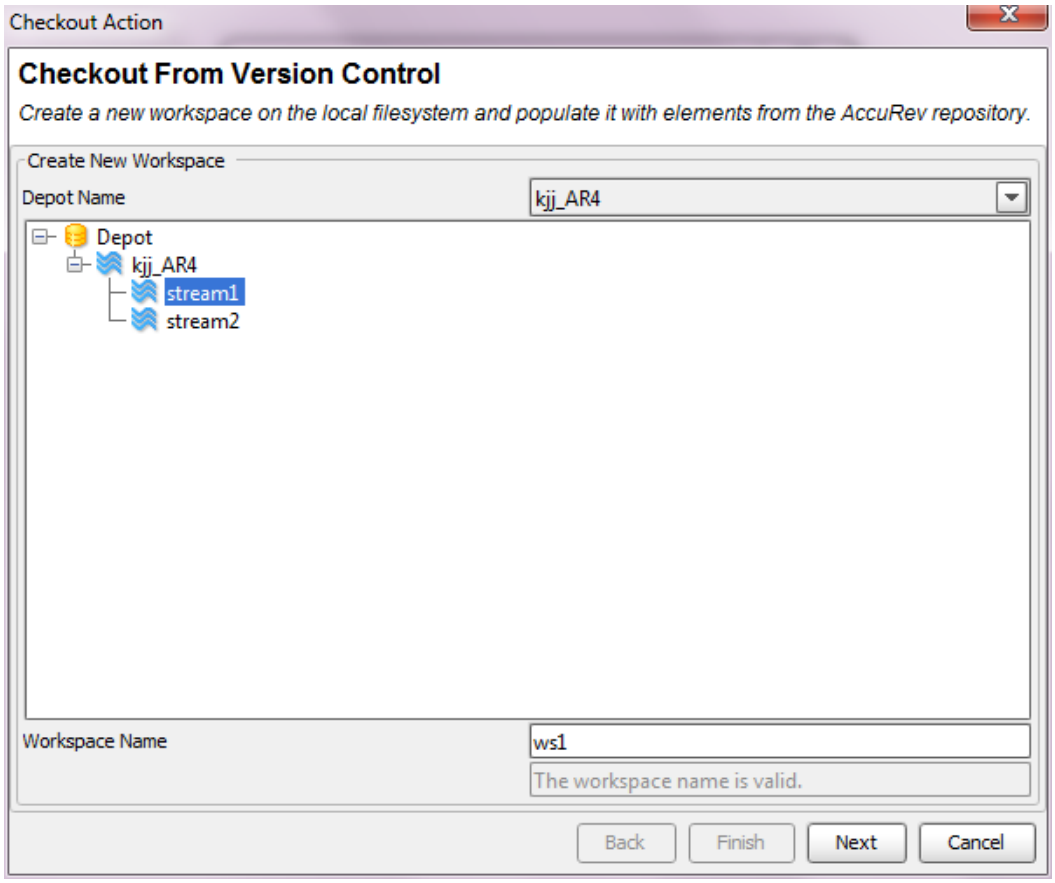
1. Arrange for the project files to be located within an existing AccuRev workspace, in either of these ways:
   - Move all the project files from their current location to a location within an existing AccuRev workspace.
   - Use the AccuRev GUI to convert the project's current disk location, or a higher-level directory, into a new AccuRev workspace (**File > New > Workspace** command).

   In either case, the project's root directory need not be the top-level directory of the workspace.
2. Open the project in IntelliJ IDEA.
3. Select **File > Settings** from the IntelliJ IDEA main menu.
4. Expand the **Version Control** node.
5. In the **Version Control** list box, select **AccuRev**.
6. If you moved data into an existing workspace or created a new workspace, as outlined in Step 1, you must place the project's files under version control. See *Converting (external)-Status Files to Version-Controlled Elements* on page 11.

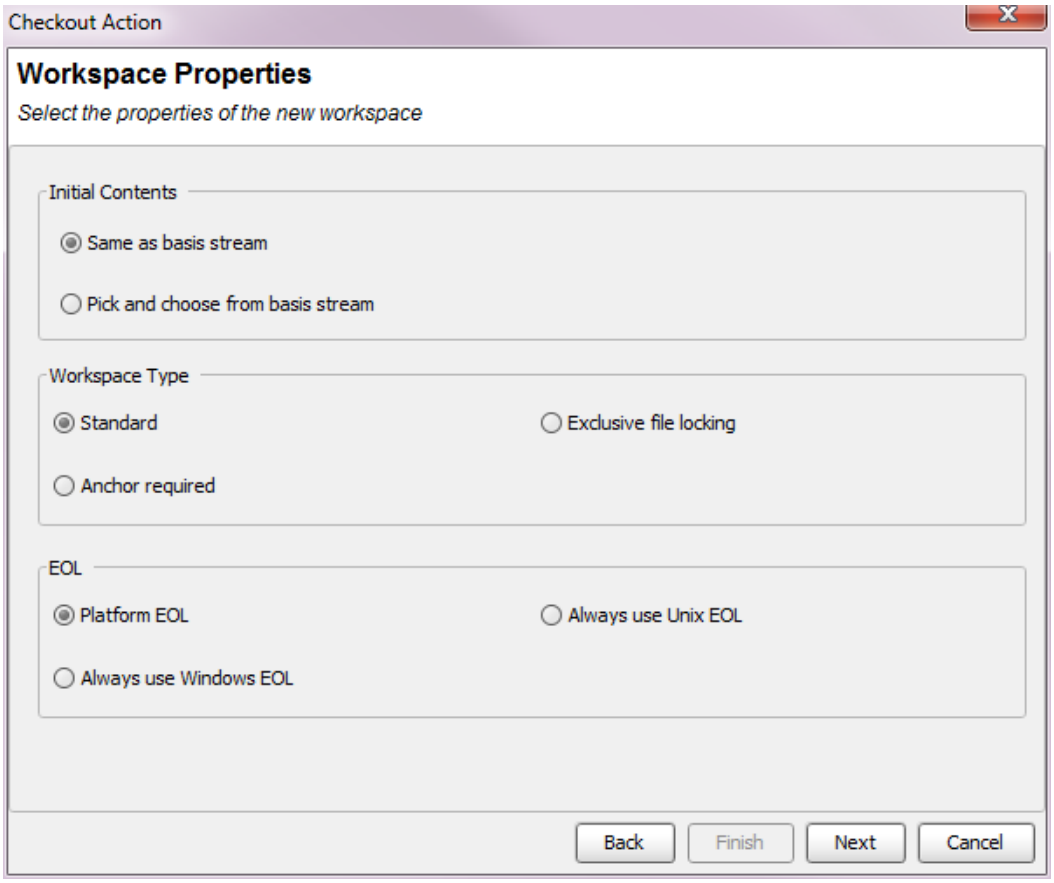
Creating a New Workspace for Sources that are Already Under AccuRev Version Control

Use this procedure if someone has already placed a set of development files under AccuRev version control on his or her own machine. This procedure creates a new workspace for your personal use, and "updates" the new workspace with the latest versions of the development files.

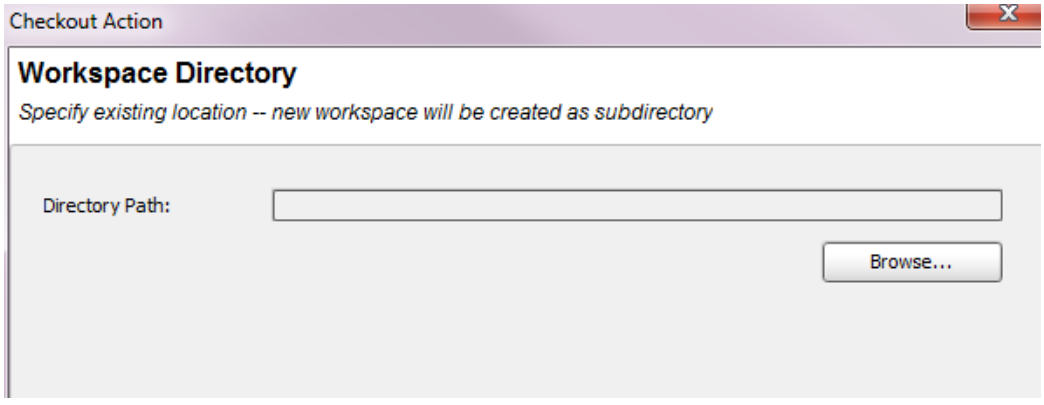
1. Invoke the command **Version Control > Checkout from Version Control > AccuRev** or select **Checkout from Version Control > AccuRev** from the IntelliJ IDEA Quick Start menu. If you are using the accurev_login authentication mode, you might be required to log in at this time.
2. On the first wizard screen, specify an AccuRev depot, a stream within that depot, and a name for the new workspace. AccuRev automatically adds your username as a suffix to the name you enter (unless you type the suffix yourself).



3. On the second wizard screen, accepting the defaults is often most appropriate. See the description of the *Anchor* command for a discussion of the exclusive file locking and anchor required features.

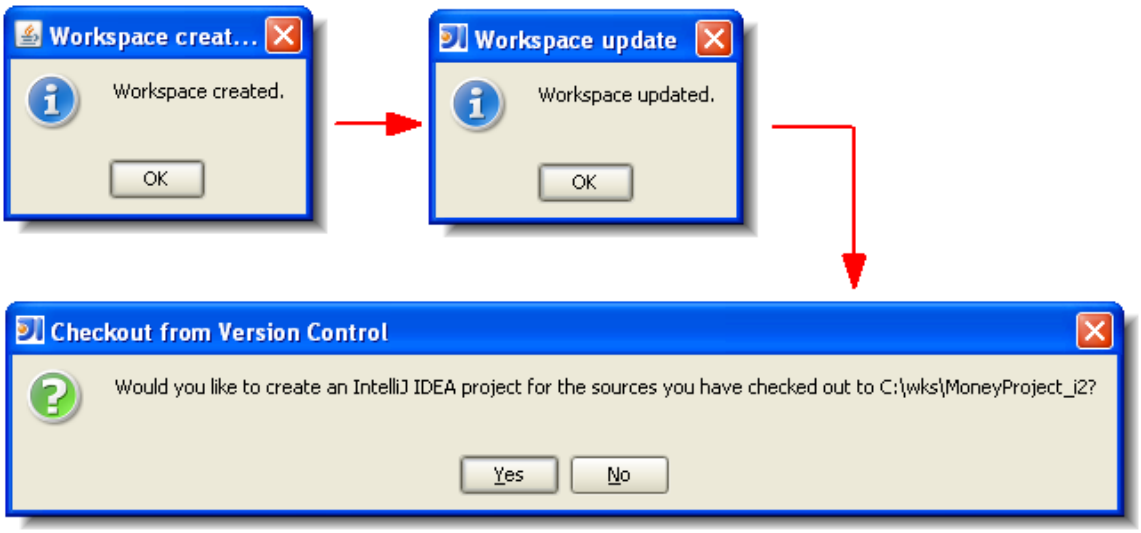


4. On the third wizard screen, click the **Browse** button and navigate to an existing directory.



The new workspace will be created as a subdirectory at the location you specify. The subdirectory name is the same as the workspace name you specified on the first wizard screen.

5. IntelliJ IDEA automatically proceeds to create an IDE project in the new AccuRev workspace:



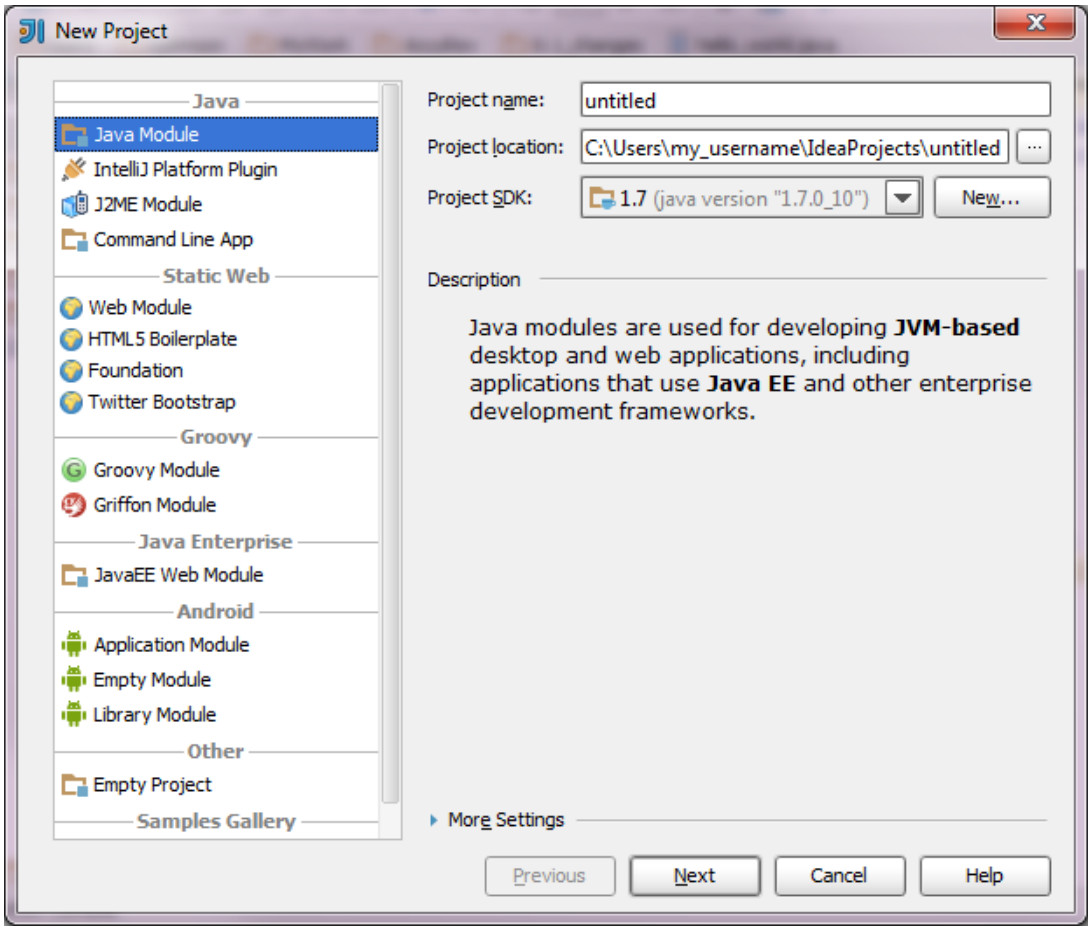
6. Click **Yes** to complete the process.

Creating a New IDE Project in a Workspace

This is essentially the standard Intellij IDEA procedure for creating a new project, with a little extra work at the end.

1. Choose **Create New Project** from the Quick Start menu, or invoke the command **File > New Project** on the IDE's main menu.

   The New Project dialog box appears:



2. Enter a **Project name**, and specify a **Project file location** that is within an existing AccuRev workspace. You can create the project in the workspace's top-level directory, or in a subdirectory.

3. Continue through the wizard's screens, ending with **Finish**.

4. Make sure that AccuRev is the version control provider for the new project — see *Making AccuRev the Version Control Provider for an Existing IDE Project* on page 7.

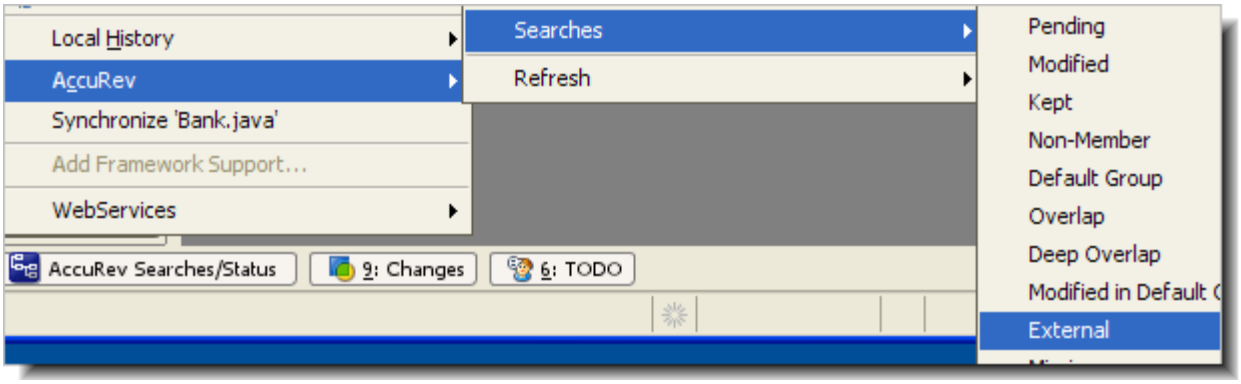
5. Make sure that the files in the project are version-controlled elements — see *Converting (external)-Status Files to Version-Controlled Elements* on page 11.

Converting (external)-Status Files to Version-Controlled Elements

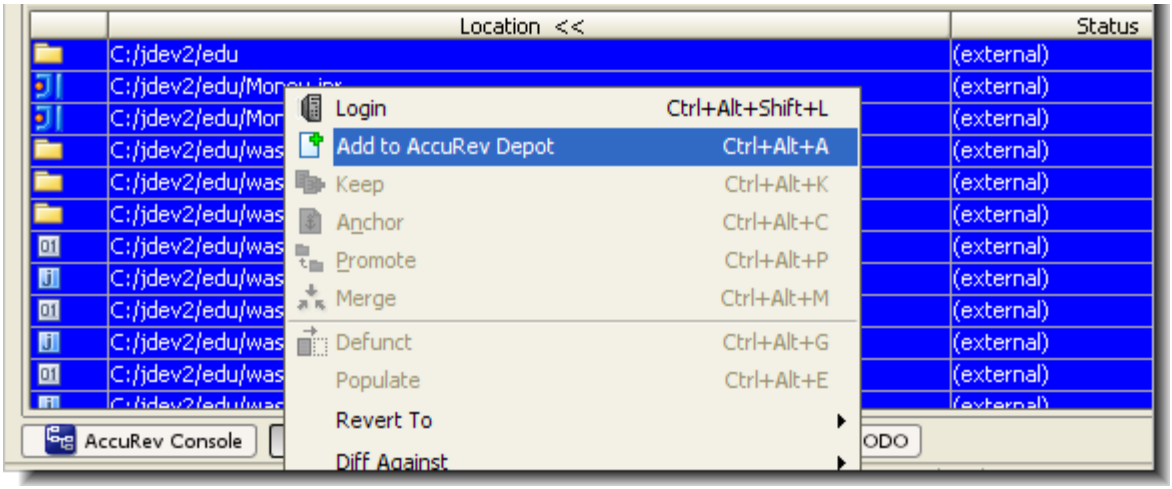
AccuRev does not automatically version-control every file within an AccuRev workspace. For example, version-controlling text-editor backup files would be a waste of resources. Initially, files created in, or copied into, a workspace have **(external)** status. You can convert such files to version-controlled elements with the Integration's **Add to AccuRev Depot** command.

Use the following procedure when you've copied a set of files into a workspace (or, perhaps, unpacked a ZIP archive containing development data).

1. Invoke the command **AccuRev > Searches > External** to list the workspace's external files at the bottom of the IDE window.



2. Select the **(external)**-status files that you want to convert to version-controlled elements. (The **Edit > Select All** command might be useful.)
3. Invoke the command **Add to AccuRev Depot**.



The newly created elements have **(kept)** status.

3. Executing AccuRev Commands

You invoke AccuRev commands in the IDE from the AccuRev menu. This menu is available under Version Control on the IDE's main menu, or from context menus in several places within the IDE, including the Project, Version Control, and AccuRev Searches/Status subwindows.

To execute an AccuRev command:

1. Right-click a file or directory; or select multiple items, then right-click the selection.
2. Choose **AccuRev** from the context menu that appears.
3. Select an AccuRev command from the context menu, or from a cascading submenu. (Commands are enabled only if they are valid for your selection.)

The AccuRev commands are described in the following section, *AccuRev Command Reference*. Certain of the IDE's own commands (that is, commands that are not on the AccuRev menu) automatically invoke AccuRev commands, in order to keep the IDE synchronized with AccuRev. This is discussed in section *IDE 'Namespace' Commands and AccuRev*.

AccuRev Command Reference

The following commands are available in the IDE on the AccuRev menu, and are described in the sections below:

Login
Add to AccuRev Depot
Keep
Anchor
Promote
Merge
Version Browser
Annotate
Defunct
Populate
Revert To
Diff Against
Synchronize Time
AccuRev Workspace Information
Update AccuRev Workspace
AccuRev History
AccuRev Statuses
Searches
Refresh
Web UI
Send To Issue
Defunct
Add To Ignore

Login

If AccuRev is configured to use the accurev_login user-authentication scheme, you must log in before performing any AccuRev operation on a file in an AccuRev workspace. If the login is invalid or fails, then no AccuRev operation can be performed.

Add to AccuRev Depot

Converts the selected file(s) in the IDE project into AccuRev version-controlled elements. The directory containing the files is also converted to an element, if necessary.

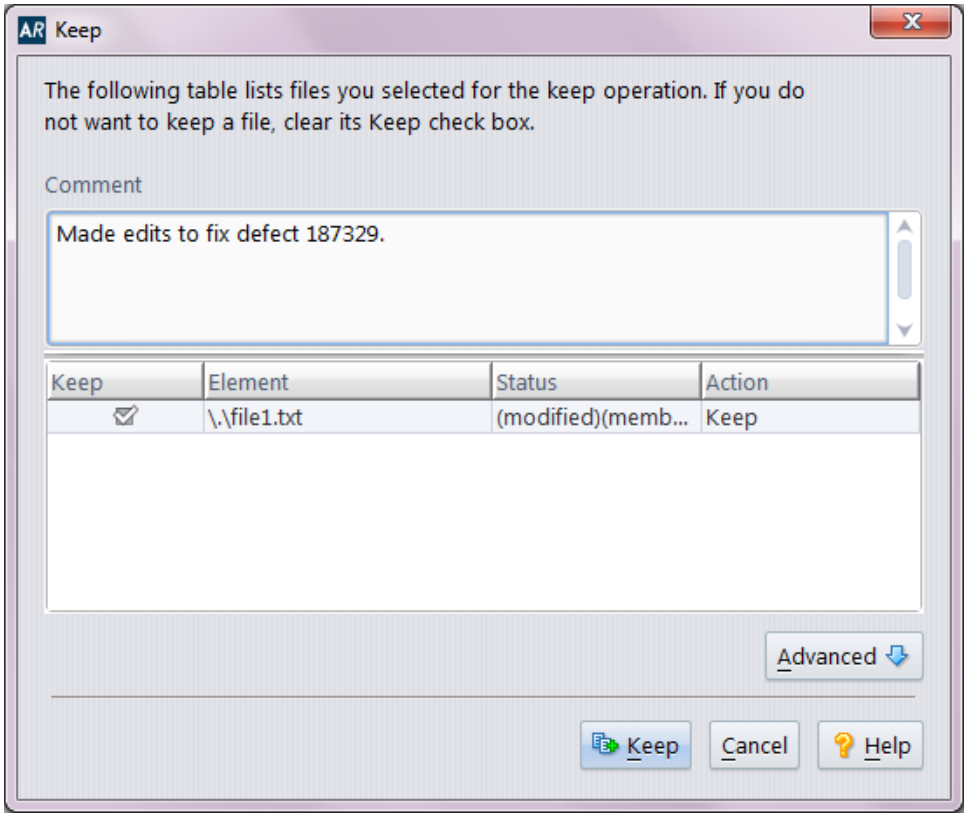
You can undo the effect of this command with the **Revert to > Backed Version** command. (See *Revert To*.) This removes the selected files from the IDE project, if you have not yet Promoted them.

Note: It is not necessary to place the IDE project files (**.iws**, .**ipr** and **.iml**) under version control.

Keep

Saves the changes you've made to one or more files as "private" versions in the AccuRev repository. These versions are visible only in your workspace — not in the backing stream or in other users' workspaces.

A dialog box appears, in which you can enter a comment string to be stored as part of the Keep transaction. The dialog makes it easy to reuse comment strings that you previously specified.



Do not confuse the versions of a file created by **Keep** with the "local history" copies of the file created when you close the Editor. Local history copies are maintained by the IDE in the local file system; versions created with **Keep** exist permanently within the AccuRev repository.

Anchor

Activates the selected file(s) in your workspace, by adding them to the workspace's default group. This ensures that the *Update AccuRev Workspace* command does not overwrite the file with a version created in another workspace.

If the AccuRev workspace uses the AccuRev exclusive file locking or anchor-required feature, files are initially read-only. (Exclusive file locking can also be enabled for individual files.) In this situation, you *must* Anchor a file before editing it, in order to make it writable. These features can be enabled during workspace creation — see *Creating a New Workspace for Sources that are Already Under AccuRev Version Control* on page 7.

If you use a standard AccuRev workspace, in which files are always writable, you'll rarely need to invoke **Anchor**.

Promote

Converts one or more "private" versions into "public" versions. That is, it takes versions that you previously created in your workspace with **Keep**, and sends them to the backing stream shared by you and other members of your development team.

Implicit Keep and Promote

If you execute a **Promote** command on files which have changed since they were last preserved in the repository with **Keep**, meaning they have **(modified)** status, the Integration invokes a sequence of AccuRev commands on the selected files:

1. Performs a **Keep** on all the files in the selection with **(modified)** status.
2. Performs a **Promote** on all the files in the selection that now have **(kept)** status.

Merge

Combines two versions of an element: your version and the version in the workspace's backing stream. This command is the equivalent of "Resolve" in other source code management terminologies.

If the element type is **binary**, you can elect to **Keep** your version or the backed version. AccuRev does not support merging binary files.

If the element type is **text**, executing **Merge** brings up a merge tool. By default, the merging of text-file contents is performed by AccuRev's own Merge tool, which uses a 3-way merge algorithm.

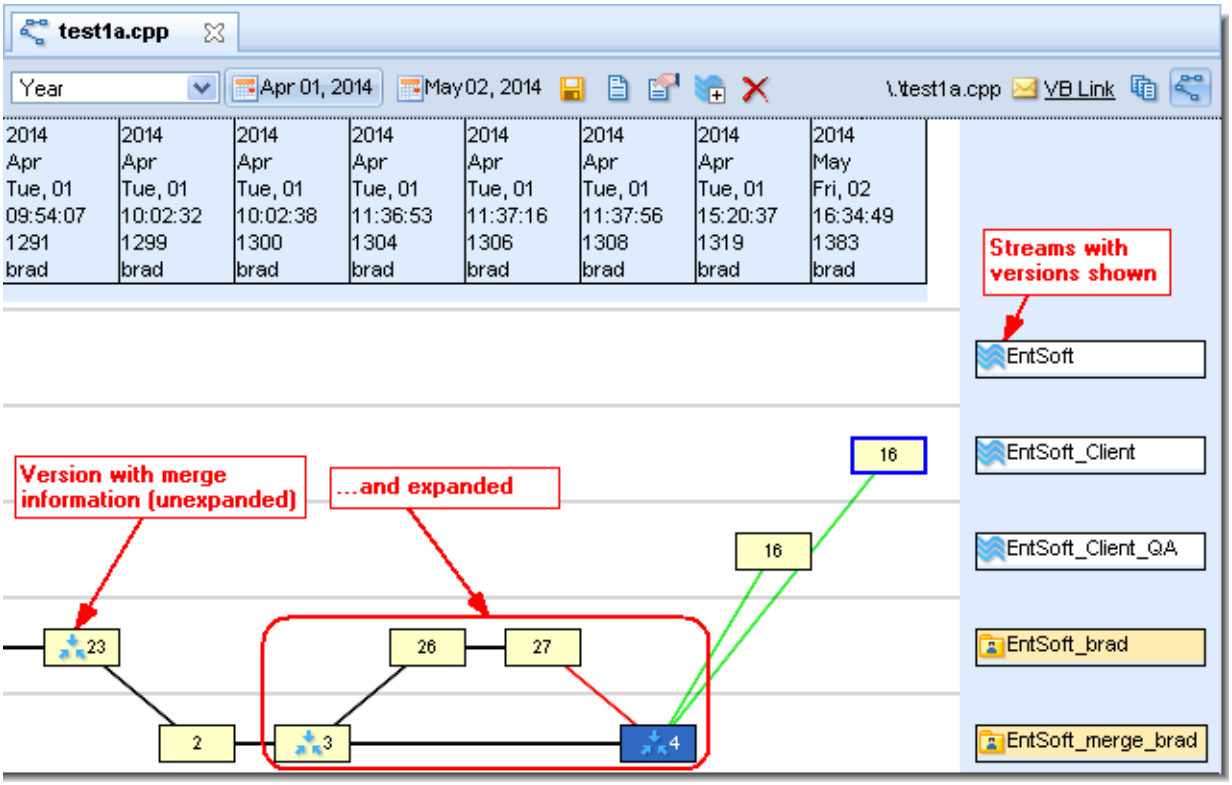
A successful **Merge** ends with a **Keep** command, creating a new version of a file in your workspace.

> **Note:** AccuRev uses a separate executable to perform Merge operations. Check that your environment variables and system path are correctly defined so this executable is available to the Integration.

Version Browser

Shows the ancestry for each element you select in the Version Browser, a graphical view that displays the current version of the selected element, the real ancestors of that version, and the virtual versions that have one of those ancestors as a parent. The **Show** field in the Version Browser toolbar specifies the number of transactions displayed in the Version Browser.

The Version Browser time line shows when each transaction took place, as well as the number and user associated with the transaction. Note that the time line proceeds left-to-right, therefore the most recent versions will be visible on the right side of the Version Browser.



Note: The Version Browser is available only if AccuRev Plug-In has been configured to provide access to the AccuRev Web Interface (Web UI). See the *AccuRev Web Interface User's Guide* for more information on the Version Browser.

Annotate

Invokes the AccuRev Web Interface to display a read-only copy of the selected file, prefixing each line with one or more of the following: the user who created the line, the transaction in which the line was added or most recently modified, the timestamp of that transaction, the version-ID of the version created in that transaction. Also noted are lines that have been modified in the workspace but not yet kept or promoted.

Defunct

Deletes a file from your disk, and marks it as having **(defunct)** status in the AccuRev workspace. A comment dialog enables you to enter comments for the defuncted element.

Other users will continue to see the file in their workspaces. When you **Promote** a defunct element to the backing stream, it disappears entirely from the workspace stream, and from the File Browser display. The element then becomes **(defunct)**, hence also active, in the backing stream.

> **Note:** The IDE's **Delete** command does not invoke AccuRev's **Defunct** command. See *Using the IDE's Delete Command* on page 26.

Populate

Restores a file with **(missing)** status to your project. To locate such files, use the Missing element search — see *Searches* on page 22.

Invoking the **Populate** command copies the version currently in the workspace stream (in the AccuRev repository) to the workspace.

> **Note:** Do not invoke Populate on a file that is not missing. This command overwrites the file, in the same way that the **Revert To > Most Recent Version** command does.

Revert To

Discards the changes you've made to a file and restores another version to the workspace.

- **Revert to > Backed Version**

  For each selected element, discards all changes since the last time you promoted the element, or since the last Update AccuRev Workspace command, whichever is more recent.

- **Revert To > Most Recent Version**

  Discards content changes for each selected element since the last private version you created with the **Keep** command. This command does not discard changes to an element's name.

Diff Against

Compares your file with another version. Only text files can be compared.

- **Diff Against Most Recent Version**

  Compares your file with the last private version you created with **Keep**.

- **Diff Against Backed Version**

  Compares your file with the version currently in the workspace's backing stream.

- **Diff Against Basis Version**

  Compares your file with the nearest ancestor version that either was created in another workspace or was promoted from this workspace to the backing stream.

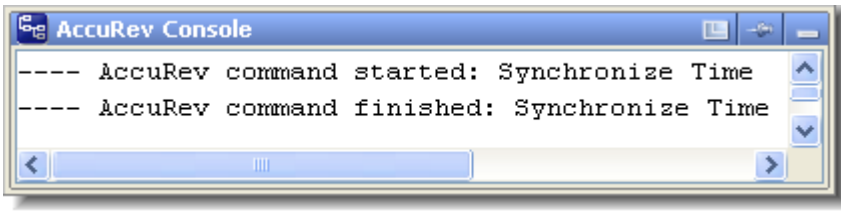
- **Diff Against File on Disk**

  Compares your file with an arbitrary file on your machine. A standard operating system "open file" dialog box appears which you can use to locate and select the file.

  Note: AccuRev uses a separate executable to perform Diff operations. Check that your environment variables and system path are correctly defined so this executable is available to the Integration.

Synchronize Time

Changes the system clock on the local machine to match the clock on the AccuRev server machine. A success (or failure) message appears in the AccuRev Console window.

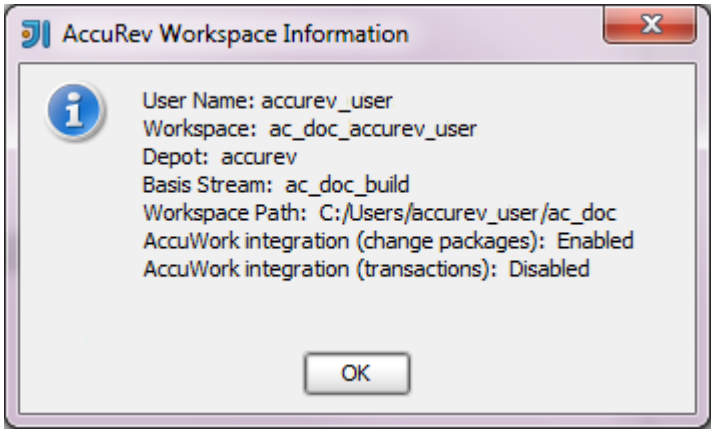


Note: You must have **root** (on Linux/Unix) or **Administrator** (on Windows) privileges to change the system time.

Some AccuRev commands are disabled if the client's time does not match the server's time within a preset tolerance.

AccuRev Workspace Information

Displays information regarding your current AccuRev work environment in a dialog box:



Update AccuRev Workspace

Copies versions from your workspace's backing stream into your workspace. This has the effect of incorporating other people's changes, which they have promoted to the backing stream, into your workspace.

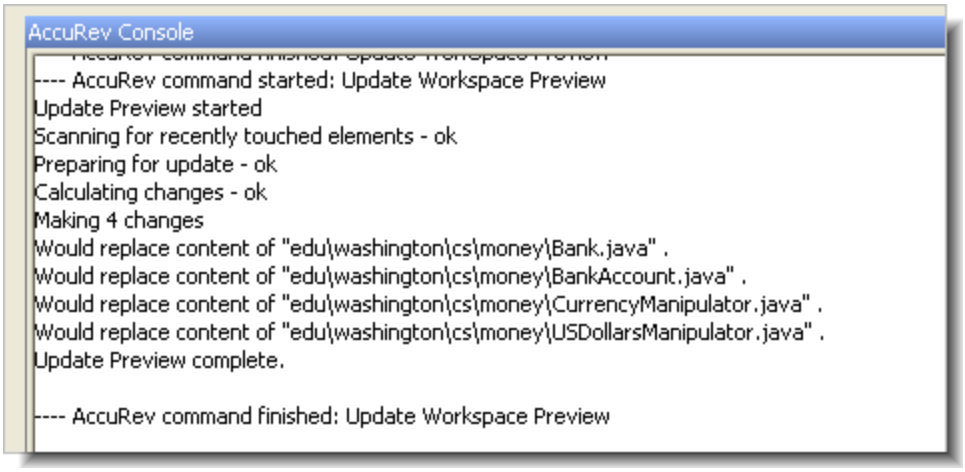
- **Update AccuRev Workspace > Entire Workspace**

  Updates the entire workspace. This command writes "started" and "finished" messages to the AccuRev Console subwindow, but does not include element-by-element details.

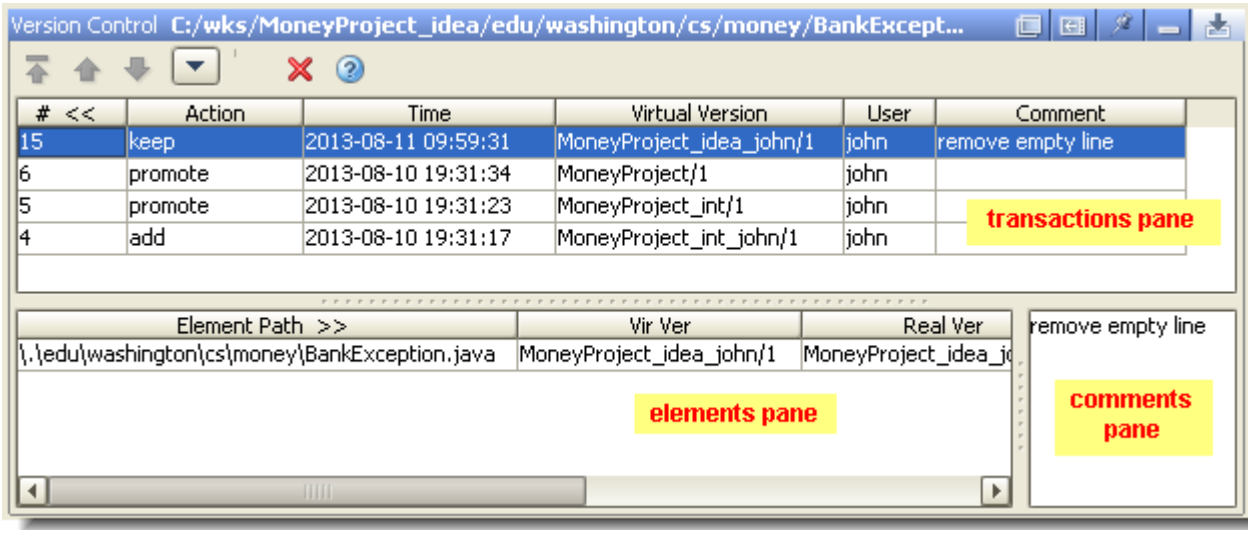
- **Update AccuRev Workspace > Preview**

  Reports to the AccuRev Console subwindow the element-by-element changes that would be made to the workspace during an actual update.

AccuRev History



Displays the transactions for the selected file or directory element in the Version Control tool window:



AccuRev tracks the complete history of each version-controlled element (file, directory, or link). This history consists of the set of transactions involving that element. Most changes to a depot are logged by transactions.

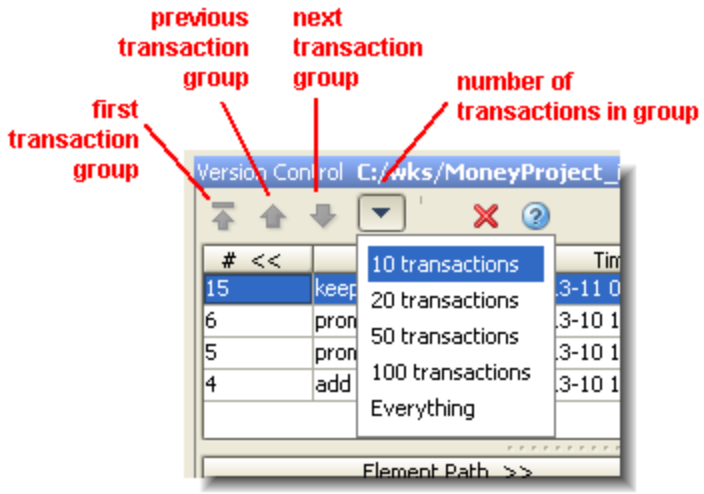
> **Note:** This command applies to only one element at a time. It is disabled in the AccuRev menu if you select more than one item.

The top pane of the Version Control tool window **(transactions pane)** shows the transactions that this element was part of. If you select a transaction, a list of all elements involved in that transaction appears in the bottom pane **(elements pane)**. The comment for the selected transaction is displayed separately in a **comments pane** to the right of the elements pane.

Typically, most of an element's transactions are created by **Keep** and **Promote** commands, as shown in the Action column in the transactions pane. Transactions are also logged in several other situations — for example:

- When an element is first added to an AccuRev depot (**add**).
- When you rename it or move it to a different directory **(move)**. This action occurs when you use the IDE to move or rename a file, for example in a refactoring operation.
- When you incorporate someone else's changes into your work (merge), and some others.

The toolbar at the top of the Version Control tool window contains navigation buttons to move among groups of transactions: first group, previous group, and next group, and a drop-down list that controls how many transactions are displayed at a time.



Right-clicking an item in the transactions pane or elements pane displays a context menu with the following commands. In the elements pane, the command operates on the selected version of the element. In the transactions pane, the command operates on the selected transaction's version of the element whose history is being displayed.

- **Open**

  Opens the selected element version in the IDE Editor.

- **Export**

  Copies the selected version to a location you specify.

- **Diff Against > Other Version**

  (*Transactions pane only*) Compares the selected version to another version that you select. Click on another transaction to specify the version created by that transaction.

- **Diff Against > Previous Transaction**

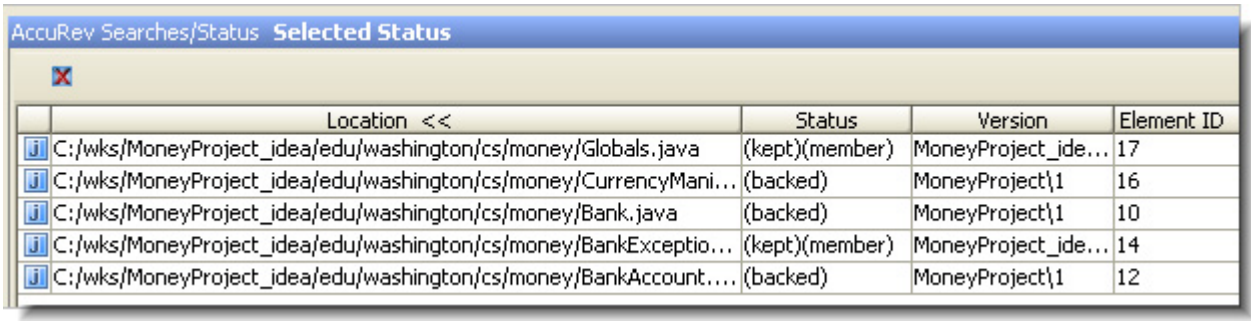
  (*Elements pane only*) Compares the selected version with the version from the previous transaction.

- **Properties**

  (*Elements pane only*) Shows the element type, element ID, and pathname within the workspace of the selected element.

AccuRev Statuses

Determines the current AccuRev status of the selected item(s), and displays the results in the AccuRev Search/Statuses subwindow.



Location <<	Status	Version	Element ID
C:/wks/MoneyProject_idea/edu/washington/cs/money/Globals.java	(kept)(member)	MoneyProject_ide...	17
C:/wks/MoneyProject_idea/edu/washington/cs/money/CurrencyMani...	(backed)	MoneyProject\1	16
C:/wks/MoneyProject_idea/edu/washington/cs/money/Bank.java	(backed)	MoneyProject\1	10
C:/wks/MoneyProject_idea/edu/washington/cs/money/BankExceptio...	(kept)(member)	MoneyProject_ide...	14
C:/wks/MoneyProject_idea/edu/washington/cs/money/BankAccount....	(backed)	MoneyProject\1	12

You can select one or more elements in this window and right-click to display the **AccuRev** context menu and perform any valid AccuRev command.

You can click on a column header to sort the listing on that column. Click again to reverse the sort order.

Icon Decorations in the Project Tool

Each file or directory object displayed in the Project tool subwindow has an AccuRev-level status, expressed as a set of status indicators – for example, **(kept)(member)**. The Project tool does not display these status indicators, but it does display corresponding icon decorations:

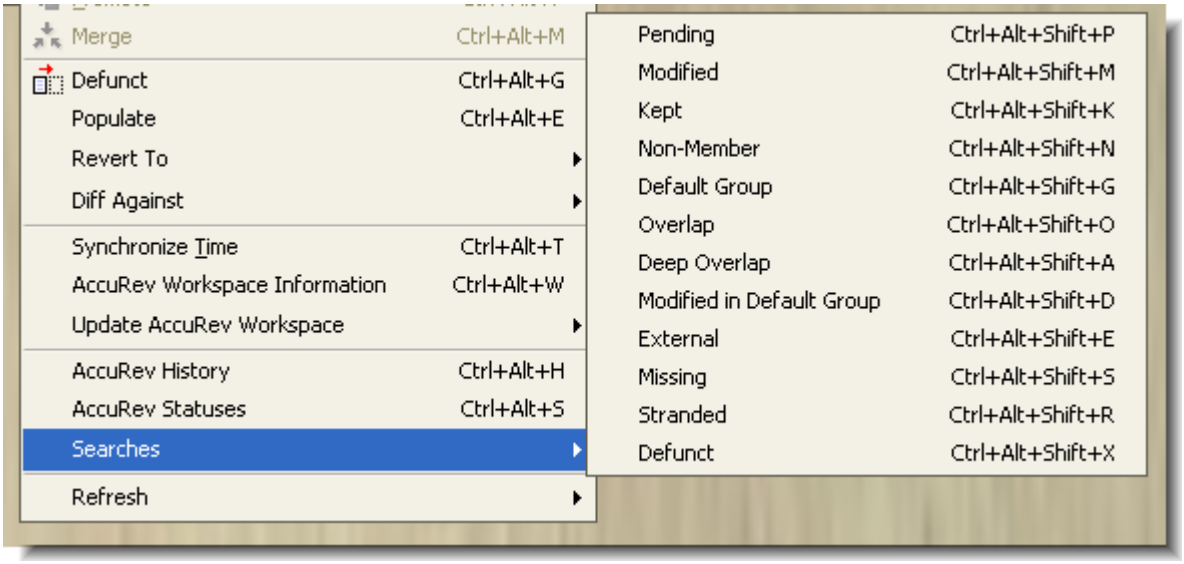
Status	Icon Decoration
(backed)	
(stale)	
(modified)	
(missing)	
(overlap)	
(member)	

The AccuRev status indicators are not mutually exclusive — for example, a file's status can include both the **(modified)** and **(member)** indicators. But the IDE uses only one decoration on each icon or each filename. The following precedence order of the AccuRev statuses determines which decoration appears:

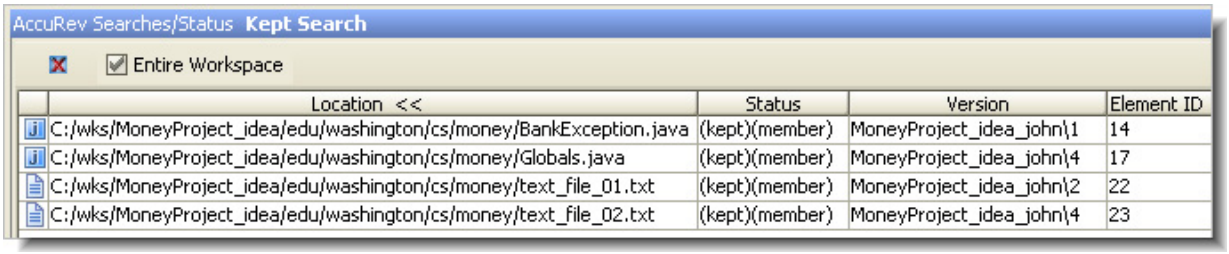
(overlap) *highest precedence*
(stale)
(missing)
(modified)
(member)
(backed) *lowest precedence*

Searches

Searches your workspace for files whose AccuRev status matches the criterion you specify — **Pending**, **Modified**, and so on:



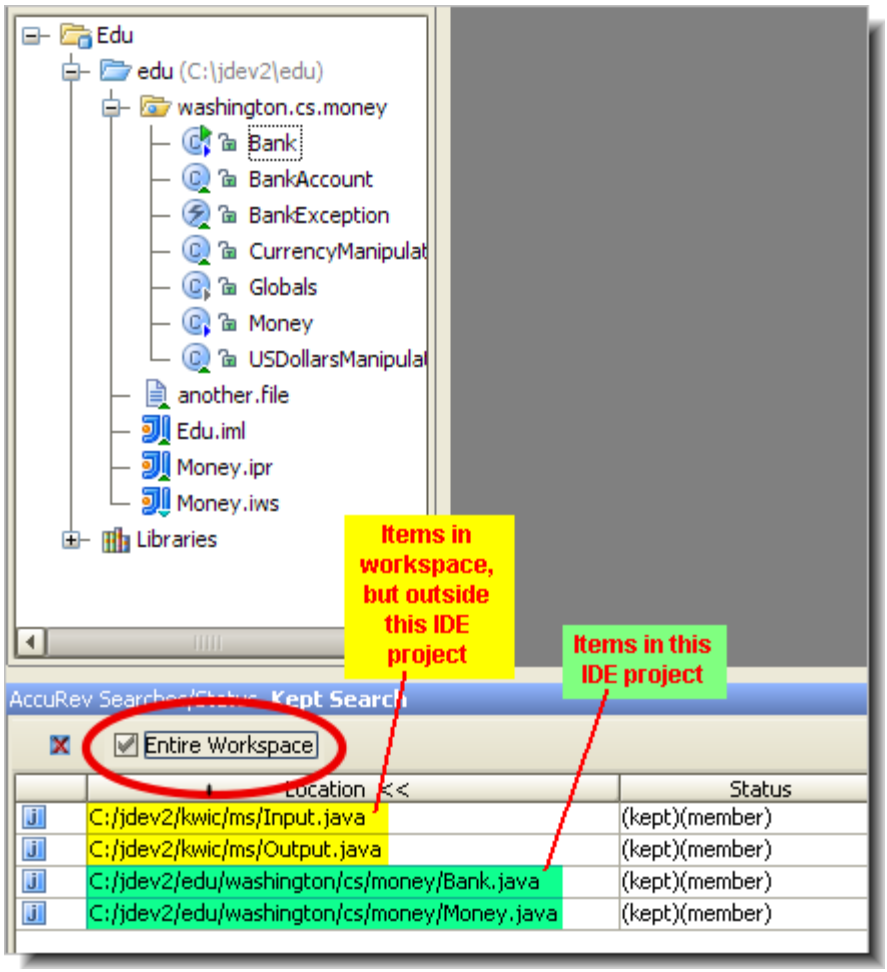
The results appear in the AccuRev Searches/Status window, replacing the current contents (if any):



You can select one or more elements in this window and right-click to display the AccuRev context menu and perform any valid AccuRev command. This makes it easy to perform such operations as:

- Promoting elements that are pending promotion.
- Keeping and Promoting **(modified)** elements.
- Promoting elements in the workspace's default group.
- Placing **(external)** files under version control.

By default, a search is conducted through the entire workspace containing the IDE project. If the project occupies a subtree of the workspace, the search results might contain files that are not in the IDE project. Clear the **Entire Workspace** checkbox to restrict the search scope to the IDE project only.



Refresh

Makes a call to the AccuRev Server and refreshes the statuses of the selected files and directories. Two options are available.

- **Refresh > Selected Content Roots**

  Refreshes the selection and (if it includes a directory) the subdirectories.

- **Refresh > Project**

  Refreshes the status of all AccuRev-controlled elements in the project.

The AccuRev icon decorations of the selected items are refreshed, too. If you have version-controlled items that are modified by the IDE or an external process, this command can help you to detect such changes to those items.

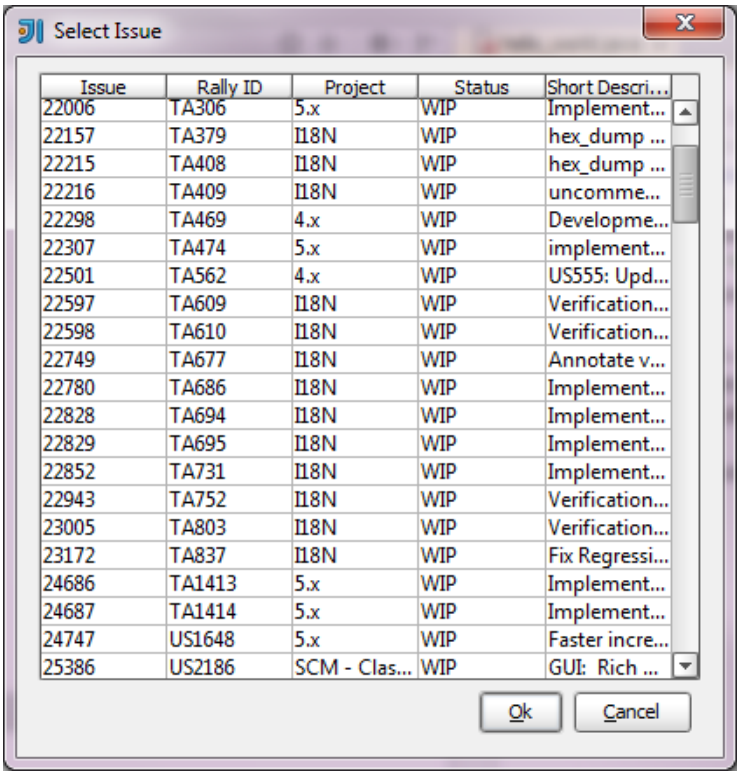
Web UI

Attempts to open the AccuRev Web Interface (Web UI).

Send To Issue

Adds the selected element to the change package of an AccuWork issue that you specify.

When you right-click a selected element and choose **AccuRev > Send To Issue** from the context menu, a list of issues returned by your AccuWork default query appears in the Select Issue dialog box:



You can select an issue and click **Ok** to add the element to the issue's change package.

See also *Defunct*.

Remove From Issue

Removes the selected element from the change package of an AccuWork issue that you specify.

When you right-click on a selected element and choose **AccuRev > Remove From Issue** from the context menu, a list of issues returned by your AccuWork default query appears in the Select Issue dialog box. Select an issue and click **Ok** to remove that element from the change package associated with the issue.

See also *Send To Issue*.

Add To Ignore

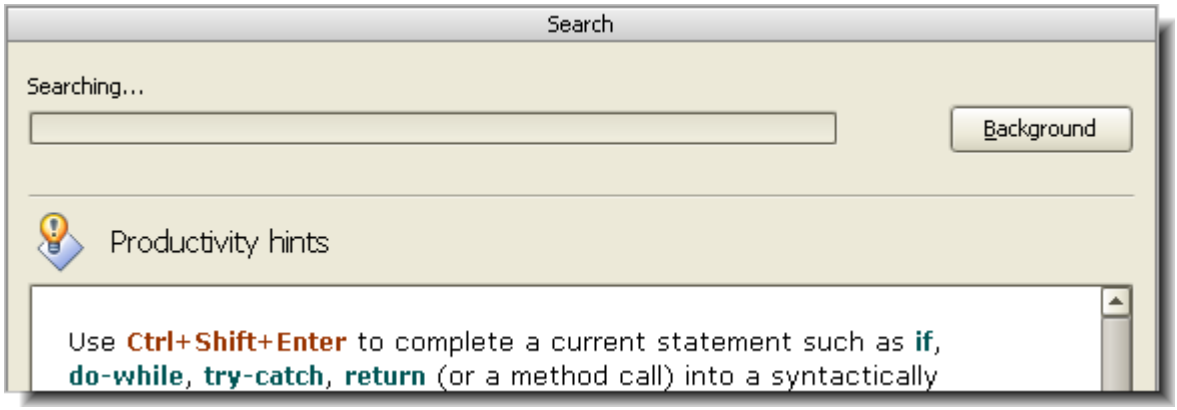
AccuRev's pathname optimization facility provides for faster performance by allowing certain objects to be ignored during various commands — notably **(external)**-status objects during whole-workspace searches. AccuRev supports specification of objects to be ignored on a per-directory basis so that one or more directories in a workspace can contain a text file named **.acignore**.

If you right-click on selected files that you want to ignore and choose **AccuRev > Add To Ignore** from the context menu, AccuRev creates a **.acignore** file in that directory configured with the files you selected. If a **.acignore** file already exists in that directory, AccuRev Plug-In updates it with the names of the files.

A **.acignore** file applies only to its own directory. In particular, it does not apply recursively to lower-level directories.

Running Commands in the Background

Only long-running commands — **Searches**, **Refresh**, and **Diff Against** — can be run in the background. When the command begins execution, a pop-up window appears:



Click **Background** to close the pop-up window and displays a command-specific message and progress indicator at the bottom of the IntelliJ IDEA window:

IDE 'Namespace' Commands and AccuRev

IntelliJ IDEA provides a method for changing the pathname of a file or directory:

- To rename an object, right-click it and select **Refactor > Rename** from the object's context menu.
- To move an object to a different directory, right-click it and select **Refactor > Move** from the object's context menu.

After either of the above operations completes, the Integration automatically invokes the AccuRev **Rename** command, to record the pathname change in the repository. As usual, the results are displayed in the AccuRev Console tool window.

If you are not logged in, the IDE's **Refactor** command proceeds, but no AccuRev **Rename** command can be executed. In this case, a warning box appears, explaining that the old name will have **(missing)** status and the new name will have **(external)** status.

A similar result occurs if you use the **Move** command to move an object from one IDE project to another project that is part of the same AccuRev workspace. The object will be moved, but no AccuRev **Rename** command will be invoked. The object will have **(missing)** status in the source location and **(external)** status in the destination location.

Using the IDE's Delete Command

Executing the IDE's **Edit > Delete** command on an element does not automatically invoke the AccuRev **Defunct** command. The element is removed from the IDE project, but gets the AccuRev status **(missing).**

To resolve a **(missing)** status after using the **Delete** command, take the following steps:

1. Invoke the command **AccuRev > Searches > Missing** to list the deleted element.
2. Invoke one of these AccuRev commands from the context menu of the deleted element:
    - The **Defunct** command removes the element at the AccuRev level, as well. (You need to Promote the element to make the removal public. You can do this after listing the element with a Defunct, Pending, or Default Group search.)
    - The **Populate** command undoes the **Delete** command, restoring the element to the IDE project. (But any recent changes that you did not preserve with **Keep** will be lost.)

4. Day-to-Day Usage of AccuRev®

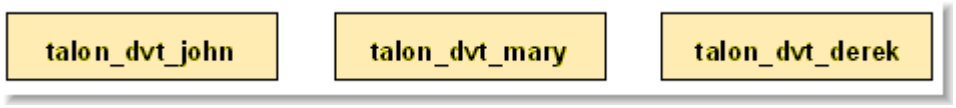
This chapter introduces the basic AccuRev concepts and procedures that will enable you to be comfortable and productive using AccuRev on day one. More detailed information is always just a click away: pressing F1 or clicking the help button ( ) anywhere in the AccuRev GUI provides you with access to the entire AccuRev documentation set in both HTML and PDF formats.

The AccuRev® Usage Model

AccuRev's flexibility makes it easy to use for a variety of development scenarios. But like every software system, AccuRev has usage models that were foremost in the minds of its architects. This section describes the most common usage model.

AccuRev is a *software configuration management* (SCM) system, designed for use by a team of people who are developing a set of files. This set of files might contain source code, images, technical and marketing documents, audio/video tracks, or any other digital content the user puts into the system. The files and directories in AccuRev are said to be "version-controlled" or "under source control".

For maximum productivity, the team's members must be able to work independently of each other -- sometimes for just a few hours or days, other times for many weeks. Accordingly, each user has his own private copy of the version-controlled files. The private copies are stored on the user's own machine (or perhaps in the user's private area on a public machine), in a directory tree called a *workspace*. We can picture the independent workspaces for a three-user team comprised of John, Mary, and Derek as follows:

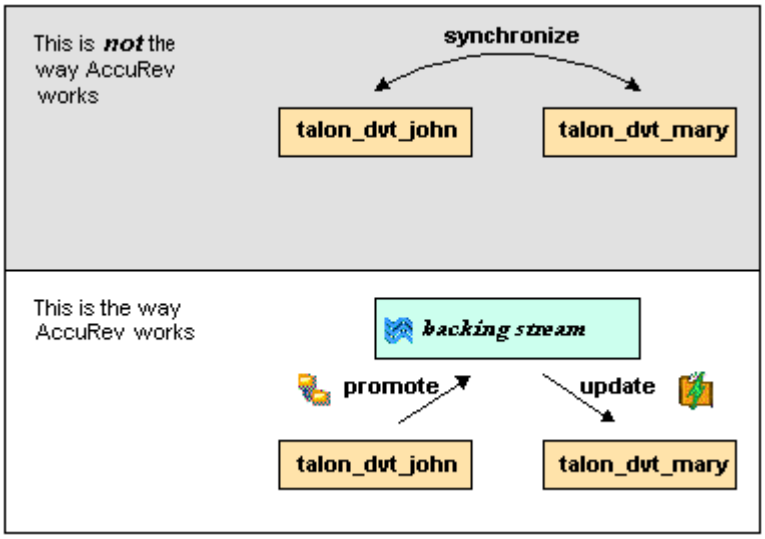


This set of users' workspaces uses the convention of having like names, suffixed with the individual usernames. AccuRev enforces this username-suffix convention. In this example, **talon_dvt** might mean "development work on the Talon product"; **john**, **mary**, and **derek** would be the users' login names.

From AccuRev's perspective, development work in this set of workspaces is a continual back-and-forth between getting "in sync" and "out of sync":

- Initially, the workspaces are completely synchronized: each workspace has copies of the same set of version-controlled files.
- The workspaces become unsynchronized as each user makes changes to some of the files.
- Periodically, users share their changes with each other. When **john** incorporates some or all of **mary**'s changes into his workspace, their two workspaces become more closely (perhaps completely) synchronized.

You might assume that the workspace synchronization process involves the direct transfer of data from one workspace to another. But this is not the way AccuRev organizes the work environment. Instead of transferring data directly between private areas (that is, between users' workspaces), AccuRev organizes the data transfer into two steps:



1. One user makes his changes public -- available to all the other members of his team. This step is called *promoting*.
2. Whenever they wish, other team members incorporate the public changes into their own workspaces. This step is called *updating*.

The first step involves a public data area, called a stream. A *stream* is an AccuRev structure with two very important features that support and simplify parallel development. First, as implied by the preceding illustration, numerous individual contributors can create workspaces on a single stream, allowing them to easily share their work with others, and to get changes from them when they wish. As you will see later, AccuRev has numerous safeguards that prevent one user from overwriting another user's changes, and tools that help you resolve conflicting changes when they occur. Second, in a stream hierarchy, *inheritance* lets child streams automatically inherit changes from the parent streams above them, reducing the need for tedious merges that might otherwise be required when several developers are working on the same code.

AccuRev has several kinds of streams, but the most common one is the backing stream. Later, we will show you how the data in this public stream "is in back of" or "provides a backstop for" all the private workspaces of the team members.

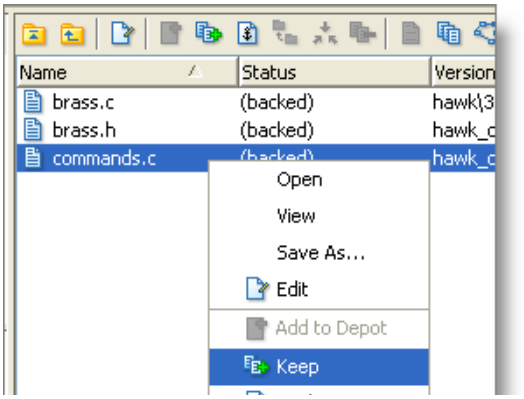
Change and Synchronization: The Four Basic Commands

With the usage model described above, you will be able to accomplish most of your AccuRev work with four simple commands: **Keep**, **Promote**, **Update**, and **Merge**. These commands are described in the following sections.

Keep: Preserving Changes in Your Private Workspace

An AccuRev workspace is just a normal directory tree, in which you make changes to version-controlled files. You can work with the files using text editors, build and test tools, IDEs, and so on, just as if the files were not version-controlled at all. For example, you might edit a source file and invoke the editor's **Save** command a dozen times over the course of an hour or two. These operations do not involve AccuRev at all -- the operating system changes the contents and the timestamp of the file in your workspace.

You do not need to perform a "check out" operation or otherwise get permission from AccuRev before editing a file in your workspace. (Some legacy SCM systems do impose such a regimen, and AccuRev can be configured to require checkouts, if your organization requires them.)



Every so often, you want AccuRev to preserve the current contents of the file as an official new *version* of the file. You accomplish this using AccuRev's **Keep** command. This figure shows how to invoke the **Keep** command from the File Browser toolbar; **Keep** is also available from the **Actions** and context (right-click) menus. Note that you can also keep new files, like **requirements.txt** in the preceding illustration -- its status is **(external)** because it has not yet been added to AccuRev.

Tip: The File Browser helps organize your workspace using Outgoing Changes (your work), Incoming Changes (others' work), and Conflicts modes (conflicting changes in your work and that of a teammate).

You can continue modifying the file, using **Keep** to preserve the latest changes, as often as you like. Other team members will not complain about "thrashing" because these new versions stay within your workspace and do not affect any other user's workspace.

AccuRev retains all the versions that you **Keep**. This makes it possible for you to roll back to any previous version you created.

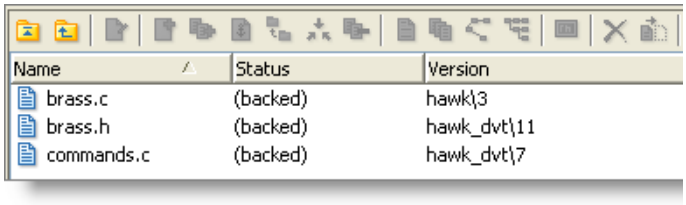
Several other operations are similar to **Keep**, in that they create a new version of a file in your workspace, without affecting any other user's workspace. The most important of these are:

- **Rename** and **Move**: You can rename a file or move it to a different directory (or both), using AccuRev commands. Other users will continue to see the file at its original pathname in their workspaces.
- **Defunct**: You can remove a file from your workspace with the AccuRev command **Defunct**. When you promote a defuncted file, other users will continue to see the file until they update their workspaces, at which point the file will be removed. (**Defunct** differs from a simple **Delete** in that **Defunct** removes the file from your workspace, while **Delete** removes the physical file from your local directory.)

More About Keep

We said earlier that AccuRev "retains all the versions that you **Keep**". But where? Each time you **Keep** a file, its current contents are copied to the AccuRev repository, located on the machine where the AccuRev Server runs. You do not need to care about the name and precise location of this copy. Each version you create has a *version-ID*, such as **velo_dfoster\2** (which translates as "the **2**nd version of this file created in workspace **velo_dfoster**").

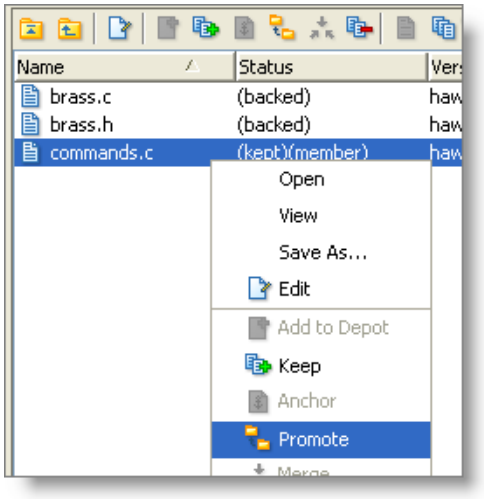
AccuRev keeps track of the *status* of each file in a workspace. After you **Keep** a file, the Status column in the AccuRev File Browser contains the indicator **(kept)**. It also contains the indicator **(member)**, meaning that the file belongs to the set of files you are actively working on. (See *Active and Inactive Files* for more information.) The Version column displays the version-ID.



A change to the data within a file, recorded by **Keep**, is termed a *content change*; the change made by **Rename**, **Move** or **Defunct** is termed a *namespace change*. (Many SCM systems do not handle namespace changes at all, or have very limited capabilities in this area.) As noted previously, AccuRev saves a new copy of the file in the repository whenever you make a content change. But it does not need to copy the file when you make a namespace change; rather, the AccuRev Server just records the change in its database.

To perform version control on directories, AccuRev only needs to keep track of namespace changes -- renaming, moving, or deleting a directory. Unlike some legacy SCM systems, AccuRev does not need to record a new directory version when you make a content change -- for example, adding a new file to the directory.

Promote: Making Your Private Changes Public

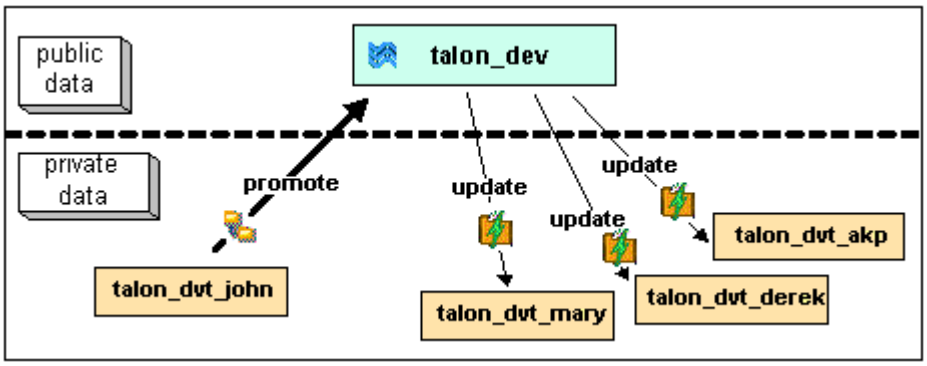


At some point, after you have used **Keep** to create one or more new, private versions of a file in your workspace, you typically want to share the changes you have made with the other team members. To make your (most recent) new version "available to the public", you *promote* it. This figure shows how to invoke the **Promote** command from the File Browser toolbar; **Promote** is also available from the **Actions** and context (right-click) menus.

Tip: You can **Promote** files that you have not yet kept -- AccuRev performs the necessary **Keep** action for you. In the case of new files, AccuRev first adds the file to the AccuRev repository.

Promoting your new version of a file does not automatically "push" it into the workspaces of the other team members. When a user decides that she is ready to incorporate versions of files that other team members have promoted, she "pulls" them into her workspace with the **Update** command. This process is described in the following section, *Update: Incorporating Others' Changes into Your Workspace* on page 32.

Streams



The **Promote** command sends data to -- and the **Update** command gets data from -- an AccuRev data structure called a *stream*. The stream (named **talon_dev** in this illustration) acts as a central data exchange for the set of workspaces used by a development team. A stream also has a bit of "traffic controller" built in, preventing team members' efforts from colliding and providing other mechanisms to control the flow of data.

A stream is not, as you might initially suppose, a set of copies of promoted files. Rather, it is more like a list of version-IDs.

- The 4th version created in workspace **talon_dvt_akp** of file **logic.xml**
- The 7th version created in workspace **talon_dvt_mary** of file **matrix.ini**
- and so on...

In SCM vernacular, a stream is a *configuration* of a collection of version-controlled files. The term "stream" is apt, because it implies the ongoing changes happening in a development project. Each time a user promotes a version of file **logic.xml**, the stream configuration changes for that file -- for example, from "the 5th version created in workspace **talon_dvt_derek**" to "the 7th version created in workspace **talon_dvt_mary**".

Promotion and Parallel Development

Sometimes, AccuRev does not allow you to promote a file to the development team's stream, because another team member has already promoted the same file (after modifying it in their own workspace). AccuRev prevents you from overwriting your colleague's change to the team's shared stream. This situation is called an *overlap*: two users working at the same time on the same goal, to create the stream's next version of a particular file.

Before you can promote your changes to the stream, you must first perform a *merge* on the file that has an overlap. This command is described in *Merge: When Changes Would Collide* on page 33.

Active and Inactive Files

As you work with a file using the commands described above, AccuRev considers the file to alternate between being *active* in your workspace and *inactive*:

- The file is initially *inactive*.
- When you create a new version in your workspace, using **Keep**, **Rename**, **Move**, or **Defunct**, the file becomes *active*.
- When you make your private version public, using the **Promote** command, the file becomes *inactive* again.

Later, you might restart this cycle, making the file active again by creating another new version of it. Alternatively, updating your workspace might overwrite your inactive file with a newer version that another team member promoted.

AccuRev keeps track of the set of active files in your workspace. Officially, this set is called the *default group*. You might find it easier to think of it as the workspace's "active group".

More About Promote

The **Promote** command does not copy the promoted version to the AccuRev repository. It does not need to. Promotion just gives an additional name to a version that *already exists* in the repository -- having been placed there by a previous **Keep** command (or **Rename**, **Move**, or **Defunct**). For example, promoting "the 7th version created in workspace **talon_dvt_mary**" might give that version the additional name "the 3rd version promoted to stream **talon_dvt**".

Just to emphasize the previous point: a stream does not reside in the file system, but in the database used by the AccuRev Server. Promoting a version to a stream does not create a copy of a file; it just creates an additional file-reference in the database.

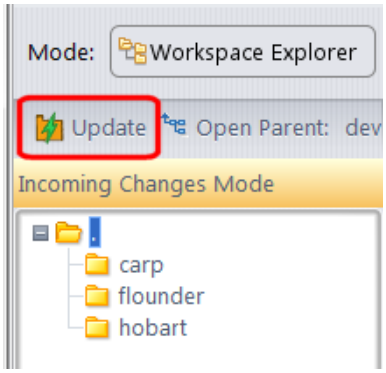
It might seem strange at first that deleting a file with the **Defunct** command makes the file *active*. The File Browser continues to list the file -- with a **(defunct)** status -- even though the file has been removed from your workspace's disk storage. This design feature enables AccuRev to implement the file-deletion operation using the same private-change/public-change scheme as all other changes.

We have discussed *the* stream that is the basis for a set of workspaces. But a typical development project has many streams, organized into a hierarchy. Promoting a version to a higher-level stream from a lower-level stream makes that version "even more public" -- for example, available to users outside your local development team.

Update: Incorporating Others' Changes into Your Workspace

As users work independently of each other, the contents of their workspaces increasingly diverge. Typically, some of the differences between workspaces are inconsistencies. For example, changes that John makes in a report-library routine might cause errors in the report program that Mary is writing. To minimize the time and effort required to resolve inconsistencies during the integration phase of a project, it makes sense to have users synchronize their workspaces on a regular basis.

With AccuRev, synchronization does not mean incorporating data into your workspace directly from one or more other workspaces. Instead, synchronization involves copying data into the workspace from the stream to which all team members **Promote** their changes. This figure shows how to invoke the **Update** command from the File Browser toolbar; **Update** is also available from the **File** and context (right-click) menus.



Tip: You can **Update** your workspace from any File Browser mode, but the Incoming Changes mode shows you exactly how your workspace will change.

> **Note***:* the stream's role as a provider of data -- through **Update** operations -- to a set of workspaces motivates the term *backing stream*. Think of restocking a store's shelves with merchandise retrieved from "the back room".

So updating your workspace copies versions of certain files from the backing stream to the workspace, overwriting/replacing the files currently in the workspace. But which files? **Update** changes a file if:

- There is a newer version in the backing stream, and
- The file is *not* currently active in your workspace.

Update will not overwrite an active file, even if there is a new version of it in the backing stream. No matter how good someone else's code is, you do not want his changes to wipe out the changes that you have been making! This situation is another instance of an *overlap*, which was introduced in *Promote: Making Your Private Changes Public* on page 30. You can encounter an overlap during a promote (if you are trying to make your private changes public), or during an update (if you are trying to bring already-public changes into your private workspace). You can use the *merge* operation to resolve all such overlap situations. See *Merge: When Changes Would Collide* on page 33.

Update handles namespace changes as well as content changes. Thus, if your colleague renamed a file and promoted the change, an update will cause the file to be renamed in your workspace. And if your colleague removed a file using the **Defunct** command, an update will cause the file to disappear from your workspace.

More About Update

Here is how AccuRev prevents an update from overwriting your changes: the first thing **Update** does is to analyze your workspace, determining whether each version-controlled file is "active" or "inactive". Initially, all the files in a workspace are inactive -- each one is a copy of some version in the repository. (For each version-controlled file, AccuRev keeps track of *which* particular version.)

A file is deemed to be active in your workspace if you have created a new version of it, using the **Keep**, **Rename**, **Move**, or **Defunct** command. (There are other AccuRev commands that "activate" a file.) When **Update** copies versions from the repository into your workspace, it skips over all active files.

> **Note***:* **Update** can tell if you have modified a file but have not yet stored the changes in the repository as a new **Keep** version. It uses timestamps and checksums to determine this. The presence of such files prevents the update from proceeding if updating would overwrite one or more of them with the backing-stream version. You can use the **Anchor** command to activate such files, enabling **Update** to do its work.

Merge: When Changes Would Collide

The preceding sections on the **Promote** and **Update** commands both discuss the situation in which two users concurrently work on the same file. Their changes to the file are said to *overlap*. Both **Promote** and **Update** decline to process a file with overlap status, because doing so would cause one user's changes to overwrite another's changes.

For example:

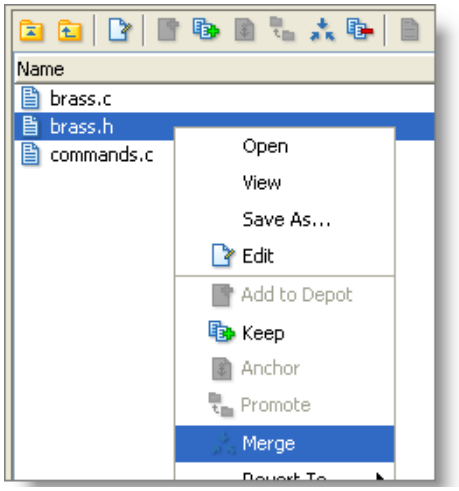
- Team members John and Mary both Keep one or more new private versions of **logic.xml** in their respective workspaces.
- Mary Promotes her latest new version of **logic.xml** to the backing stream.
- At this point, AccuRev:
  - Will not allow John to **Promote** his version of **logic.xml** to the backing stream (there is no reason to assume that John's changes should take precedence over the changes Mary has already made and promoted).
  - Will not overwrite John's copy of **logic.xml** when he updates his workspace (again, AccuRev does not assume that a version in the backing stream should take precedence over changes made to the

version in the workspace). The **Update** command skips over this file, but continues its work on other files.

- "Flags" the overlap condition by highlighting **logic.xml** in yellow.

Before John can either promote or update **logic.xml**, he must incorporate, or merge, the version in the backing stream -- which contains Mary's changes -- into his own copy of the file. The **Merge** command is essentially a text editor, which combines the contents of two versions of a text file. The resulting "merged version" replaces the file in John's workspace.

This figure shows how to invoke the **Merge** command from the File Browser toolbar; **Merge** is also available from the **Actions** and context (right-click) menus.

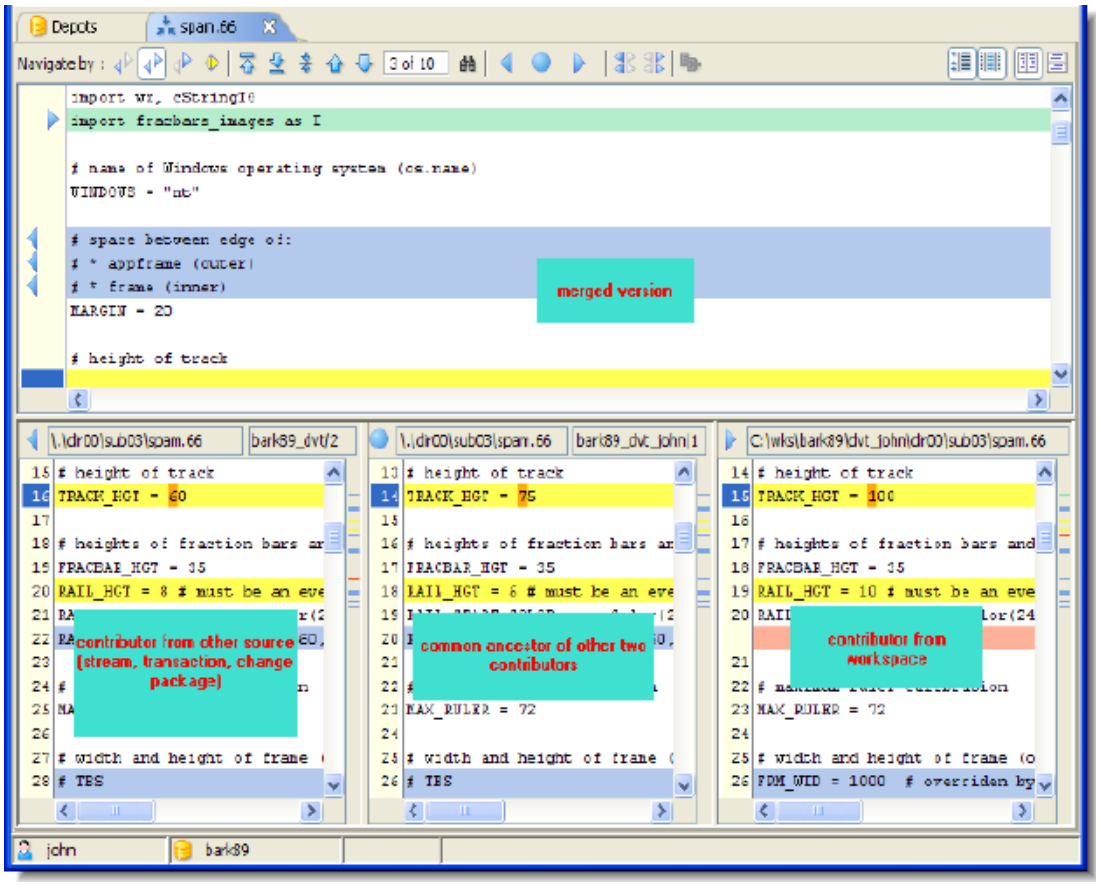


Tip: Use the File Browser Conflicts mode to view overlap and other conditions that prevent files from being updated and promoted.

Often, a merge operation is unambiguous, and so can be performed automatically. For example, suppose Mary's changes to file **logic.xml** all occur in lines 30–50, and all of John's changes occur in lines 125–140. In this case, merging the two versions involves replacing some or all of John's 20 lines with Mary's. Now, the edited version of **logic.xml** in John's workspace contains both users' changes.

> **Note***:* We do not claim that the two sets of changes are semantically consistent with each other. That is what the build-and-test cycle is for!

If both John and Mary have made changes to the same part of the file -- say, lines 2-10 -- then John must decide how to resolve this *conflict*. The graphical **Merge** tool makes this easy:



After performing a merge, AccuRev automatically Keeps the merged version to preserve the results of the merge operation. You can then Promote the merged version to the backing stream. After that, other team members can use Update -- perhaps in conjunction with Merge -- to bring all the changes into their workspaces.

More About Merge

The graphical Merge tool performs a "three-way merge", which uses the common ancestor of the two versions being merged. This algorithm helps to automate the merge operation, often completely eliminating the need for human intervention. AccuRev performs merge operations on text files only. Binary files are "merged" by choosing which version to take.

AccuRev keeps track of all merge operations. This greatly simplifies subsequent merge operations on files that have been merged previously: you do not need to resolve the same conflicts over and over again.

The most common overlap situation happens when AccuRev prevents you from promoting a file, because someone else "got there first" in creating a version in the backing stream. AccuRev can also detect *deep overlaps*, in which another user "got there first" in creating a version in the *parent* of the backing stream, or in other higher-level streams.