

# The Benefits of Managing the Build Process with Surround SCM Snapshot Branches

Many source code management solutions rely on labels and tags to manage the build and release processes. While Surround SCM contains labeling functionality, snapshot branches provide several useful features that can improve these processes. This article discusses the benefits of using snapshot branches to manage builds and releases in Surround SCM.

## What snapshot branches offer

Snapshot branches record the state of source code at a given point in time so it can be reproduced in the future. As part of the build process, users can continually check in changes to the source code and decide which of the previous builds will be used for testing and eventually packaged into a release. Once the source code is isolated in the preferred state, creating a snapshot branch allows users to check in additional changes to the source code without changing the build configuration contained in the snapshot. This helps track changes over time and provides build managers with a way to manage the changes that are included in a release.

## Snapshots cannot be modified

Snapshot branches are read-only so their contents cannot be modified, but snapshots act like all other Surround SCM branches in a branch structure. You can create a baseline or workspace branch from the snapshot and use the Surround SCM promote command to merge changes into the snapshot's read-write parent branch. This allows you to skip over the read-only branch during the merge operation.

## Branches offer flexible security

Surround SCM contains many additional security options that control who can rename, remove, and hide branches. You can set up security groups that provide for more flexible control of snapshot (or other) branches.

## Snapshots can be visible or hidden

Snapshot branches appear in the Branch tree making them easier to identify than labels, which are only displayed in certain dialog boxes. However, creating builds on a regular basis can add a large number of branches to the Branch tree. You can use the User Options to hide snapshot branches, and the administrator can also hide older, unused branches.

## Using snapshots for builds

You can use both labels and snapshots for builds, but snapshots are recommended for guaranteed repeatable builds. For example, you support multiple versions of a product and must maintain separate patches and fixes for each product version. After you complete a build you can create a snapshot branch that includes all the build files and historical information. To create a snapshot branch, select the baseline branch you want to archive and the corresponding repository, and choose Branch > Create Branch. After the snapshot branch is created it can be viewed in the Branch tree.

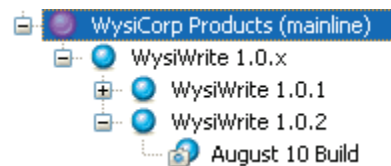


Figure 1. A typical branch structure including a snapshot branch associated with a build date.

A couple of weeks later you may be ready to create another build that includes changes made to the source code. You can create another snapshot branch for that build. Creating snapshots at each project milestone is a helpful way to see progress over time.

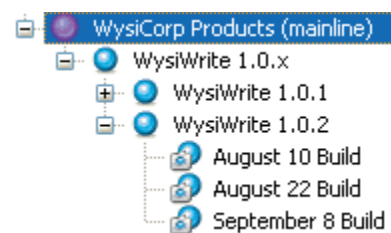


Figure 2. Snapshots to show different project milestones.

Suppose you discover a major issue with your most recent build so you need to re-create the previous build. When using snapshot branches, it is simply a matter of getting a copy of the files included in the snapshot branch for that previous build.

### File name and location information is saved

Moving and renaming files does not cause issues when you use snapshot branches like it can when using labels. When you use labels, only the file content is saved when the label is added. If you need to re-create an old build, using labels creates the possibility for old build scripts not to work properly if files have been renamed or moved. When you use snapshot branches, you will notice that files have the same names and are in the same locations they were in when the snapshot was created. This guarantees that old builds can be reproduced exactly.

### Branch differences

Snapshot branches allow you to compare two branches and see how the contents are different as well as to see variances between the individual files. You can see which files changed from one build to another and which were renamed, moved, and removed.

## Conclusion

While labels can be used similarly, snapshot branches offer more benefits for managing build and release processes. Snapshot branches are historically precise when tracking renames, moves, and removes in Surround SCM. Snapshots can be created faster on a larger number of files and include more features and security options than labels. Snapshot branches also allow you to quickly see the components of a given build and to easily re-create the build without reassembling the files by hand. When snapshot branches are used in into the build process, build and releases managers, developers, and users benefit from these added features and capabilities.

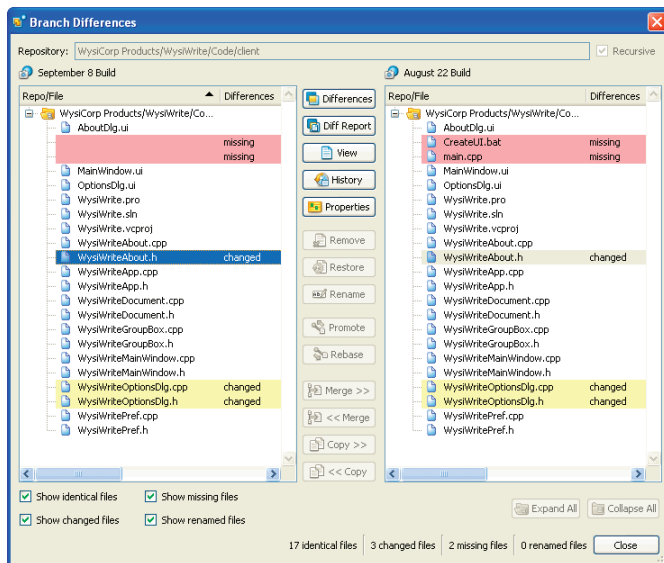


Figure 3. Branch differences between snapshots.