Switching a Working Copy

The `svn switch` command transforms an existing working copy into a different branch. While this command isn't strictly necessary for working with branches, it provides a nice shortcut to users. In our earlier example, after creating your private branch, you checked out a fresh working copy of the new repository directory. Instead, you can simply ask Subversion to change your working copy of */calc/trunk* to mirror the new branch location:

```
$ cd calc

$ svn info | grep URL
URL: http://svn.example.com/repos/calc/trunk

$ svn switch http://svn.example.com/repos/calc/branches/my-calc-branch
U   integer.c
U   button.c
U   Makefile
Updated to revision 341.

$ svn info | grep URL
URL: http://svn.example.com/repos/calc/branches/my-calc-branch
```

After “switching” to the branch, your working copy is no different than what you would get from doing a fresh checkout of the directory. And it's usually more efficient to use this command, because often branches only differ by a small degree. The server sends only the minimal set of changes necessary to make your working copy reflect the branch directory.

The `svn switch` command also takes a `--revision` (`-r`) option, so you need not always move your working copy to the “tip” of the branch.

Of course, most projects are more complicated than our *calc* example, containing multiple subdirectories. Subversion users often follow a specific algorithm when using branches:

1. Copy the project's entire “trunk” to a new branch directory.

2. Switch only *part* of the trunk working copy to mirror the branch.

In other words, if a user knows that the branch-work only needs to happen on a specific subdirectory, they use `svn switch` to move only that subdirectory to the branch. (Or sometimes users will switch just a single working file to the branch!) That way, they can continue to receive normal "trunk" updates to most of their working copy, but the switched portions will remain immune (unless someone commits a change to their branch). This feature adds a whole new dimension to the concept of a "mixed working copy"—not only can working copies contain a mixture of working revisions, but a mixture of repository locations as well.

If your working copy contains a number of switched subtrees from different repository locations, it continues to function as normal. When you update, you'll receive patches to each subtree as appropriate. When you commit, your local changes will still be applied as a single, atomic change to the repository.

Note that while it's okay for your working copy to reflect a mixture of repository locations, these locations must all be within the *same* repository. Subversion repositories aren't yet able to communicate with one another; that's a feature planned beyond Subversion 1.0.[12]

Switches and Updates

Have you noticed that the output of `svn switch` and `svn update` look the same? The `switch` command is actually a superset of the update command.

When you run `svn update`, you're asking the repository to compare two trees. The repository does so, and then sends a description of the differences back to the client. The only difference between `svn switch` and `svn update` is that the `update` command always compares two identical paths.

That is, if your working copy is a mirror of */calc/trunk*, then `svn update` will automatically compare your working copy of */calc/trunk* to */calc/trunk* in the `HEAD` revision. If you're switching your working copy to a branch, then `svn switch` will compare your working copy of */calc/trunk* to some *other* branch-directory in the `HEAD` revision.

In other words, an update moves your working copy through time. A switch moves your working copy through time *and* space.

Because `svn switch` is essentially a variant of `svn update`, it shares the same behaviors; any local modifications in your working copy are preserved when new data arrives from the repository. This allows you to perform all sorts of clever tricks.

For example, suppose you have a working copy of */calc/trunk* and make a number of changes to it. Then you suddenly realize that you meant to make the changes to a branch instead. No problem! When you `svn switch` your working copy to the branch, the local changes will remain. You can then test and commit them to the branch.

---

[12] You *can*, however, use `svn switch` with the `--relocate` switch if the URL of your server changes and you don't want to abandon an existing working copy. See the `svn switch` section in *Chapter 9, Subversion Complete Reference* for more information and an example.