**How to add a new application to the repository**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#Howtoaddanewapplicationtotherepository)

(*Most of the commands below were adapted from the original* [*RepoLayout*](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/RepoLayout) *page*)

If you have not already accessed the repository at least once using svn commands, See [HowToSubversion](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversion) (access from the Wiki does not count).

**Add an Existing Application to the repository**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#AddanExistingApplicationtotherepository)

This example shows the procedure for creating an initial trunk version containing the AWIPS I local application, tagging it, and then delivering the ported AWIPS II version.

Suppose you are porting a local application called coopqc and the current AWIPS I version number for coopqc is 2.5. The path in the repository we will add this to is ldadroot/apps. For a GFE SmartTool, it would be ldadroot/gfe/smarttools.

When using Subversion, changes are make in a "working copy". It is recommended to use a common naming convention for the top level directory that contains Subversion working copies. "svn\_work" will be used here. In AWIPS 2, /localapp/dev is reserved for site use for development so that will also be used here. You can also keep the working copy of your application in your home directory. You should use your home directory for the working copy if multiple user's at your site would be working on the same application.

Finally, the setting of environment variables for APP, WK, and REPO are just to make the instructions more generic. For REPO, change the last part of the path from ***apps*** to whatever path from the repository you plan on adding your app (i.e., **gfe/smarttools**).

* The following example is for a LAD application. For GFE stuff, you need to add in the path the GFE subdirectory like procedures, smartinits, smarttools, etc. So REPO might be:
* export REPO=https://collaborate.nws.noaa.gov/svn/ncladt/ladroot/gfe/smarttools

Example:

# ----- Define env vars for original AWIPS I source code, repository.

# ----- The app has already been copied from AWIPS 1 onto AWIPS 2/ADAM.

export APPNAME=coopqc

export A1APP=/localapps/dev/a1/$APPNAME

export REPO=https://collaborate.nws.noaa.gov/svn/ncladt/ladroot/apps

# ----- Make a temporary work area, Must be empty!

export WK=$HOME/svntmp

cd ~

rm -rf $WK

mkdir -p $WK

cd $WK

# ----- Get a working copy of the target directory

svn checkout -N $REPO tmp\_wcopy

The -N option in the checkout is used to only get the toplevel directory and not the contents. If you were to omit -N, then you would check out everything under apps (which is quite large now) and it would take a long time. So all that is checked out is the directory, not everything in it.

# ----- Create directory hierarchy for application within the working copy.

cd tmp\_wcopy

mkdir $APPNAME

cd $APPNAME

cp -a $A1APP trunk

mkdir branches tags

cd trunk

At this point, you have the application staged and ready to include in the repository. Please take the time now to clean up the application's directory of old backup files, tar files, data, logs, etc. You only want to commit the files that are needed to run or create the application. Do not include data, log, or config files that the program creates on its own. Delete compiled files like \*.o, \*.pyc, \*.pyo, etc. Remember, once a file is added to the repository it is in there forever. Try to get the initial import as pristine as possible.

Do not include tar files that are used to distribute the application. These can always be recreated by checking out the appropriate revision from the repository.

# ----- Tell svn to add the new application.

cd $WK/tmp\_wcopy

svn add $APPNAME

# ----- Review files that will be committed.

svn status

# ----- Use svn delete to delete any files that should not be committed.

# Sample command: svn delete coopqc.py.20100104

# ----- Commit the new application into the repository.

svn commit -m "Initial import of AWIPS I version of $APPNAME."

**Tag Original AWIPS 1 version**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#TagOriginalAWIPS1version)

If you have already started porting and the code has already diverged from the AWIPS 1 version, this step can be skipped.

# ----- Create the tag for the AWIPS I version

cd $APPNAME

# ----- This mkdir is a one time special case to define a branch for AWIPS 1 versions.

svn mkdir tags/awips1

svn cp trunk tags/awips1/2.5

svn status

# ----- svn status should show added files in tags/awips1/2.5

svn commit -m "Tagging original AWIPS I version of coopqc."

**Clean up temporary working copy**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#Cleanuptemporaryworkingcopy)

At this point, the files are all in the repository, clean up the temporary working copy in tmp\_wcopy. Next step will check out a working copy of just your application.

cd

rm -rf $WK/tmp\_wcopy

**Check out an Application to do modifications**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#CheckoutanApplicationtodomodifications)

Simple and generic for any directory you want to check out from the repository.

If checking out to a shared location outside your home directory, setting umask 002 is recommended before performing these steps.

mkdir -p /localapps/dev/svn\_work

cd /localapps/dev/svn\_work

svn checkout $REPO/$APPNAME

You can optionally supply a destination directory at the end of the checkout command, typically used when checking out trunk (I always check out to a directory with "\_svn" appended so I know it is a working copy):

cd /localapps/dev/svn\_work

svn checkout $REPO/$APPNAME/trunk ${APPNAME}\_svn

**Modify and commit changes**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#Modifyandcommitchanges)

Now the code is ready for editing in /localapps/dev/svn\_work/$APPNAME/trunk. To submit the changes back to the repository, use svn commit

1. Always do svn update before starting to make changes.
2. Edit files as you normally would.
3. Must use svn commands to rename, copy or delete files and directories.
4. Must use svn add to add new files.
5. Directories can be added with svn add or svn mkdir.
6. Help: svn help and svn help cmd
7. Put revision number and date in your code: [SubversionTips](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/SubversionTips)
8. Always use svn status immediately before commit to double check what will be uploaded.
9. svn commit -m "A descriptive message of the change, Please include the app name in the message."

**Tagging a New Version**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#TaggingaNewVersion)

Tagging is used to capture a specific state of the files and simply give that state a name.

1. You must have a working copy of the application including the tags and trunk directories.
2. cd into the application's top level directory.
3. <tagname> can be anything but is normally some sort of a version number. The tags/<tagname> directory must not exist.
4. svn cp trunk tags/<tagname>
5. svn status
6. # ----- svn status should show added files in tags/<tagname>
7. svn commit -m "Tagging version <tagname> of <appname>."

**Starting a new application from scratch**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#Startinganewapplicationfromscratch)

This is the same as everything above, just don't have any code to initially copy in or tag.

Set environment variables WK, APPNAME, and REPO as above.

rm -rf $WK

mkdir -p $WK

cd $WK

svn checkout -N $REPO tmp\_wcopy

cd tmp\_wcopy

mkdir $APPNAME

cd $APPNAME

mkdir trunk tags branches

cd trunk

# ---- Set up recommended application directory structure

# ---- do only what your app needs, but doc required!

mkdir bin doc config etc data logs lib src

cd $WK/tmp\_wcopy

svn add $APPNAME

svn status

svn commit -m "Initial setup for new application $APPNAME"

cd

rm -rf $WK/tmp\_wcopy

cd /localapps/dev/svn\_work

svn checkout $REPO/$APPNAME

**Short version of adding existing app**[**¶**](https://collaborate.nws.noaa.gov/trac/ncladt/wiki/HowToSubversionNewApp#Shortversionofaddingexistingapp)

Just the commands, no comments!

Set environment variables WK, APPNAME, A1APP and REPO as above.

***Clean out all the crap that doesn't belong in the repository from the $A1APP directory before starting*.**

rm -rf $WK

mkdir -p $WK

cd $WK

svn checkout -N $REPO tmp\_wcopy

cd tmp\_wcopy

mkdir $APPNAME

cd $APPNAME

cp -a $A1APP trunk

mkdir branches tags

cd $WK/tmp\_wcopy

svn add $APPNAME

svn status

svn commit -m "Initial import of application $APPNAME"

svn mkdir tags/awips1

svn cp trunk tags/awips1/<tagname>

svn status

svn commit -m "Tagging original AWIPS I version of $APPNAME."

cd

rm -rf $WK/tmp\_wcopy

cd /localapps/dev/svn\_work

svn checkout $REPO/$APPNAME ${APPNAME}\_svn