
Subject: Re: removing old IDL versions

Posted by [John-David T. Smith](#) on Tue, 27 Nov 2001 16:45:14 GMT

[View Forum Message](#) <> [Reply to Message](#)

Harald von der Osten-Woldenburg wrote:

>
> Thanks a lot for your hints. So I will not remove older versions...
>

I've attached a bash shell script I use extensively to manage all my version of IDL. It makes a series of links for running older versions, e.g.:

```
idl (the latest)
idl_5.5
idl_5.4
...
```

and similarly with:

```
idlhelp
idlhelp_5.5
idlhelp_5.4
...
```

```
idlde
idlde_5.5
idlde_5.4
...
```

etc. The links and main script are generated automatically. If you install IDL in, e.g., /usr/local/rsi, just run the script from there, whenever a new version is in place. It's very useful for pulling up an old version of help, or testing fatal bugs in prior version of IDL ;)

Another very nice feature: the correct IDL library is inserted on your path, automatically, for whichever version you invoke. An even better way to use IDL paths is to avoid the environment variable IDL_PATH altogether, and use instead IDL_EXTRA (special to this script), which will be appended to the path after the IDL libraries are included. So, for a blanket installation, run makelinks, then the user need not setup any IDL_PATH, etc., unless he wants additional libraries, in which case he specifies them with IDL_EXTRA (and IDL_DLM_EXTRA if necessary).

Tested under Linux with bash, but should be fairly portable.

JD
#!/bin/bash

```

# makelinks: script for generating the links to run multiple versions
# of installed IDL, including the plain command line version, the
# development environment, and the online help, separately. Run this
# script after installing/deleting a new version of idl.

# NOTE: WHEN INSTALLING A NEW VERISON OF IDL, DO NOT SAY "Y" TO THE
# "MAKE LINKS" OPTION OF THE INSTALL SCRIPT WHICH RUNS AT THE END OF
# INSTALLATION (YOU CAN OPT NOT TO RUN IT, AS THIS IS ALL IT DOES).
# INSTEAD, RUN THIS SCRIPT AFTER INSTALLATION COMPLETES.

# THIS SCRIPT MUST BE AT THE MAIN RSI INSTALL LEVEL, E.G. /usr/local/rsi

# J.D. Smith 2000-11-5

base=$(dirname $0)
# Edit this to make the links elsewhere, but this is usually good.
destdir="$base/../bin"

[ -x $destdir/idl ] || {
    echo "Generating $destdir/idl script"
    cat <<\EOF > $destdir/idl
#!/bin/bash
# idl: A script to run multiple versions of RSI's IDL, including
# separately the command line, the development environment, or the
# online help. Each version is linked to this script (named idl),
# with names of type "prog_vers", where prog is one of:
# idl
# idlde
# idlhelp
# and vers is the version number, like 5.2. An example is idlde_5.4.

# You can them simply say, e.g., idlhelp_5.1 to get version 5.1's
# help, or idl_5.4 to run the command-line idl version 5.4, etc. If
# you find a version available which does not work, alert the
# sysadmin.

# The full set of links for all installed version of IDL are created
# by the script "makelinks" available in the RSI install directory.
# See that script for more info. Run it after installing or deleting
# an installation of idl in the RSI install directory (often
# /usr/local/rsi, or /export/local/rsi for exported shares). The
# installation directories are by default named idl_vers
# (e.g. /export/local/rsi/idl_5.4), and this naming convention must be
# retained. If this script is run via one of its links which has no
# version number (idl, idlde, or idlhelp), the latest version will be
# run. All IDL_PATH entries which reference a preset $IDL_DIR will be
# updated. This will ensure the correct set of IDL libraries are

```

```
# being run.
```

```
# The user need no longer set IDL_DIR, it will be set by this script.  
# It's not a problem if they do. They can also use IDL_EXTRA to  
# specify other paths they'd like to search, which will be appended  
# beyond the distributed IDL library's path. If they'd like to  
# override the library, for instance, they can specify IDL_PATH  
# directly, being sure to include $IDL_DIR/lib. This is less portable  
# and not always a good idea, unless you really need to specify the  
# ordering of the paths.
```

```
# N.B.: This all means a new user need set up *nothing* in their  
# .cshrc, .login, etc., to run any version of IDL immediately, and  
# should simply specify a colon-separated lists of paths in  
# "IDL_EXTRA" for their local programs (with a + prefix for recursion,  
# if desired).
```

```
# N.B.: THIS FILE GOES IN THE DIRECTORY LABELED "destdir" IN THE  
# SCRIPT makelinks (see the RSI install directory),
```

```
# J.D. Smith 2000-11-5
```

```
# Set this to the RSI install base, to run this script  
rsi_base=/usr/local/rsi
```

```
base=$(dirname $0)
```

```
# Save the (possible) user-set idl dir  
[ -n "$IDL_DIR" ] && old_dir=$IDL_DIR
```

```
# See how we were called.
```

```
vers=${0##*/}  
case $vers in  
  *idlde*)  
    vers=${vers#idlde_};  
    prog="idlde";  
    ;;  
  *idlhelp*)  
    vers=${vers#idlhelp_};  
    prog="idlhelp";  
    ;;  
  *idl*)  
    vers=${vers#idl_};  
    prog="idl";  
    ;;  
  *)  
    echo "Improperly formatted command name: $0";  
    exit 1;
```

```

;;
esac

# Only dots and numbers
vers=$(echo $vers | tr -cd '[0-9].')

# No version? No problem. Find the latest version.
[ -z "$vers" ] && {
    latest="0.0"
    for scr in $base/idl_*; do
        [ -L $scr ] || continue # Only links please
        dvers=${scr##*/}
        dvers=${dvers#idl_}

# Only dots and numbers, we need to compare
dvers=$(echo $dvers | tr -cd '[0-9].')
[ -z "$dvers" ] && continue

        major=${dvers%%.*}
        minor=${dvers#*.}
        minor=$(echo $minor | tr -d '.')
        fvers="$major.$minor"
        [ -n $(echo "if (${fvers}>${latest}) 1" | bc) ] && {
            vers=$dvers;
            latest=$fvers;
        }
    done
}

[ -z "$vers" ] && {
    echo "Cannot find any validly named version of idl in $base." 1>&2;
    exit 1;
}

export IDL_DIR="$rsi_base/idl_$vers"

# Update any relevant parts of the IDL_PATH to point to the new directory
if [ -n "$old_dir" -a -n "$IDL_PATH" ]; then
    export IDL_PATH=$(echo $IDL_PATH | sed "s:$old_dir:$IDL_DIR:g")
fi

# If there wasn't one, make sure the lib directory is included
[ -z "$IDL_PATH" ] && export IDL_PATH="+$IDL_DIR/lib${IDL_EXTRA:+:$IDL_EXTRA}"
export IDL_DLM_PATH="+$IDL_DIR/bin${IDL_DLM_EXTRA:+:$IDL_DLM_EXTRA} "

# Run the correct version
exec $IDL_DIR/bin/$prog $@

```

```

EOF
chmod +x $destdir/idl
}

# Kill the old links, in case we removed a directory
rm -f $destdir/idl{,de,help}_*

# Go through the installed idl directories, making links
latest="0.0"
for dir in $base/idl_*; do
  [ -d $dir -a ! -L $dir -a -d $dir/bin ] || continue
  vers=${dir##*/}
  vers=${vers#idl_}

  # Only dots and numbers, we need to compare
  vers=$(echo $vers | tr -cd '[0-9].')
  [ -z "$vers" ] && continue

  echo VERSION $vers
  for end in "" de help; do
    echo "Making ${destdir}/idl${end}_${vers}";
    ln -s idl ${destdir}/idl${end}_${vers};
  done

  # Find the latest version
  major=${vers%%.*}
  minor=${vers#*.}
  minor=$(echo $minor | tr -d '.')
  fvers="$major.$minor"
  [ -n $(echo "if (${fvers}>${latest}) 1" | bc) ] && {
    latest_vers=$vers;
    latest=$fvers;
  }
done

# Set up links for running the latest version
[ -L $destdir/idlde ] || {
  echo "Making $destdir/idlde";
  ln -s idl $destdir/idlde;
}
[ -L $destdir/idlhelp ] || {
  echo "Making $destdir/idlhelp";
  ln -s idl $destdir/idlhelp;
}

[ -L $base/idl ] && {
  echo "LATEST VERSION";
  echo "Making idl->idl_${latest_vers}";
}

```

```
rm -f idl;  
ln -s idl_${latest_vers} idl;  
}
```

File Attachments

1) [makelinks](#), downloaded 122 times
