Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by Pepijn Kenter on Thu, 11 Mar 2004 15:36:16 GMT
View Forum Message <> Reply to Message

> Is this a Windows thing? Collecting data for a call to RSI. :-)
>

Apperently, it works fine under my linux:

```
IDL> Print, !Version
{ x86 linux unix linux 6.0 Jun 27 2003      32      64}
IDL> !P.Multi=[0,2,3]
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
     0.0468800     0.705734     0.485943     0.980474
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
      0.546880     0.705734     0.985942     0.980474
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
     0.0468800     0.372401     0.485943     0.647140
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
      0.546880     0.372401     0.985942     0.647140
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
     0.0468800    0.0390675     0.485943     0.313807
IDL> plot, findgen(11)
IDL> print, !X.Window[0], !Y.Window[0], !X.Window[1], !Y.Window[1]
      0.546880    0.0390675     0.985942     0.313807
```

By the way, what is the easiest way to switch between IDL versions under
linux? What I do now is using a soft link: idl -> idl_6.0 and renaming
it when I need to run 5.5. Aren't there any command line options?
Couldn't find them

Pepijn.

Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by Michael Wallace on Thu, 11 Mar 2004 16:54:27 GMT
View Forum Message <> Reply to Message

> By the way, what is the easiest way to switch between IDL versions under
> linux? What I do now is using a soft link: idl -> idl_6.0 and renaming
> it when I need to run 5.5. Aren't there any command line options?
> Couldn't find them

There aren't any command line options because IDL 5.5 and IDL 6.0 are
two separate programs and they aren't aware that the other exists.
While soft-linking works, you should also make sure that your IDL_PATH
is updated to include the correct directories. You wouldn't want IDL
5.5 trying to use commands built for 6.0 or vice versa.

A better solution would be to package the soft link command and update
of IDL_PATH in a script. Then when you want to switch, just call that
command. Maybe if I have time later today, I'll provide an example of
what I'm talking about.

--Mike

Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by Foldy Lajos on Thu, 11 Mar 2004 17:14:08 GMT

View Forum Message <> Reply to Message

Hi,

On Thu, 11 Mar 2004, Michael Wallace wrote:

>> By the way, what is the easiest way to switch between IDL versions under
>> linux? What I do now is using a soft link: idl -> idl_6.0 and renaming
>> it when I need to run 5.5. Aren't there any command line options?
>> Couldn't find them
>
> There aren't any command line options because IDL 5.5 and IDL 6.0 are
> two separate programs and they aren't aware that the other exists.
> While soft-linking works, you should also make sure that your IDL_PATH
> is updated to include the correct directories. You wouldn't want IDL
> 5.5 trying to use commands built for 6.0 or vice versa.
>
> A better solution would be to package the soft link command and update
> of IDL_PATH in a script. Then when you want to switch, just call that
> command. Maybe if I have time later today, I'll provide an example of
> what I'm talking about.
>
> --Mike
>

just create two soft links:

```
 ln -s /usr/local/rsi/idl_5.5/bin/idl /usr/local/bin/idl55
 ln -s /usr/local/rsi/idl_6.0/bin/idl /usr/local/bin/idl60
```

and use the idl55 and idl60 commands (idl is a script itself, which sets
the appropriate env. variables).

regards,
lajos

Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by Michael Wallace on Thu, 11 Mar 2004 21:07:04 GMT
View Forum Message <> Reply to Message

> just create two soft links:
>
> ln -s /usr/local/rsi/idl_5.5/bin/idl /usr/local/bin/idl55
> ln -s /usr/local/rsi/idl_6.0/bin/idl /usr/local/bin/idl60
>
> and use the idl55 and idl60 commands (idl is a script itself, which sets
> the appropriate env. variables).

In this case, you still have the issue of what IDL_PATH is set to. If left unset, the soft links solution you suggest will work. IDL will automagically determine the correct lib directory to use.

However, almost all of us have to explicitly set IDL_PATH in order to include our own libraries and other procedures. Because of this we are forced to choose between the $RSI_DIR/idl_5.5/lib directory and the $RSI_DIR/idl_6.0/lib directories. And you can't set IDL_PATH to include both directories since one will clobber the other.

So, with that in mind, the better solution is to symlink the IDL directories rather than the executables. By using a symlink that points to idl_5.5 or idl_6.0 or other directory, we don't have the IDL_PATH issue because we just have to only include $RSI_DIR/idl/lib in IDL_PATH. So, all we have to do is change the symlink and we're ready to go.

I hacked up a quick little script in the last few minutes to do the recreation of the symlink for us. I make no guarantees about the quality. I show it here as an example only.

```
#!/bin/bash

#

# Usage: switchidl version_number
#
# Example: switchidl 5.6
#
# This script will recreate the idl symlink on systems where multiple
# versions of IDL are installed.  Two assumptions are made:
```

```
#   1.  All IDL installations are in the $RSI_DIR directory
#   2.  All IDL installations are in directories of the form idl_xxx
#       where xxx is the version number
#


# If RSI_DIR is not set, use the IDL default
if [ ! -n "$RSI_DIR" ]; then
    RSI_DIR=/usr/local/rsi
fi


# Check for the version number argument
if [ ! -n "$1" ]; then
    echo Usage: switchidl version_number
    exit 1
fi


# Check if requested IDL directory exists
if [ ! -e $RSI_DIR/idl_$1 ]; then
    echo IDL $1 could not be found
    exit 2
fi


# Check if IDL symlink already exists
if [ -e $RSI_DIR/idl ]; then


    # Delete the symlink, if possible
    if [ -w $RSI_DIR/idl ]; then
       rm $RSI_DIR/idl
    else
       echo Incorrect permissions -- could not alter symlink
       exit 3
    fi
else


    # Check if it's possible to write to $RSI_DIR
    if [ ! -w $RSI_DIR ]; then
       echo Incorrect permissions -- could not alter symlink
       exit 3
    fi
fi

# All checks are satisfied
# Move to $RSI_DIR, create the link and return
owd=`pwd`
cd $RSI_DIR
ln -s idl_$1 idl
cd $owd
```

Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by mmiller3 on Fri, 12 Mar 2004 21:40:42 GMT

View Forum Message <> Reply to Message

>>>> > "Pepijn" == Pepijn Kenter <kenter_remove_spam@tpd.tno.nl> writes:

    > By the way, what is the easiest way to switch between IDL
    > versions under linux? What I do now is using a soft link:
    > idl -> idl_6.0 and renaming it when I need to run
    > 5.5. Aren't there any command line options? Couldn't find
    > them

I have this function in my .bashrc:

```
idl_version () {
    source /usr/local/rsi/idl_${1}/bin/idl_setup.bash
}
```

Use it like this:

```
  idl_version 6.0   # now the idl command is for version 6.0
  idl_version 5.5   # now the idl command is for version 5.5
```

Mike

Subject: Re: System Variable Behavior Changed in IDL 6.0
Posted by JD Smith on Fri, 12 Mar 2004 23:24:09 GMT

View Forum Message <> Reply to Message

On Thu, 11 Mar 2004 10:54:27 -0600, Michael Wallace wrote:

>> By the way, what is the easiest way to switch between IDL versions under
>> linux? What I do now is using a soft link: idl -> idl_6.0 and renaming
>> it when I need to run 5.5. Aren't there any command line options?
>> Couldn't find them
>
> There aren't any command line options because IDL 5.5 and IDL 6.0 are
> two separate programs and they aren't aware that the other exists.

> While soft-linking works, you should also make sure that your IDL_PATH
> is updated to include the correct directories. You wouldn't want IDL
> 5.5 trying to use commands built for 6.0 or vice versa.
>
> A better solution would be to package the soft link command and update
> of IDL_PATH in a script. Then when you want to switch, just call that
> command. Maybe if I have time later today, I'll provide an example of
> what I'm talking about.

Attached are two scripts which work together. The first, "makelinks", automagically makes links to run all your installed versions of idl, including nice things like idlhelp_5.6. Just plop it in /usr/local/rsi/ (or wherever your idl directory lives), and run from there. I usually keep about three versions around, but any number are supported.

The other script "idl" is a front end to all idl* commands (idlhelp, idl, idlde). It sets IDL_DIR for you, sets up the PATH for you to include the correct version of the IDL-shipped libraries, and lets you add extra libraries to your path with an environment variable IDL_EXTRA. It's great for a system-wide install because then users don't need to bother with any environment variables. You can of course include more system libraries in the script than just the IDL default $IDL_DIR/lib.

Hope they help.

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
# DON'T RUN makelinks UNTIL YOU'VE VERIFIED IDL IS WORKING WITH THE NEW
# LICENSE SERVER.  RUN IT DIRECTLY TO CHECK.

# 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 "Cannot find the master script $destdir/idl" 1>&2;
    exit 1;
}

# 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;
}


#-------------------------- Cut Here ---------------------------------
#     This is the idl script: put in /usr/local/bin/

#!/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.  The 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, as seen by the computer 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}"

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