Subject: Saved Visualizations
Posted by David Fanning on Thu, 03 Feb 2011 17:14:17 GMT
View Forum Message <> Reply to Message

Folks,

I had *no* idea how nice it is to save data visualizations.
It opens up a whole new world of possibilities! I just
saved a visualization on my Windows machine, e-mailed it
to my Linux machine and opened it up. Now I'm looking
at exactly the same visualization on both machines!

Wow! E-mail one of these babies to your colleagues and
they can be looking at *exactly* what you are looking
at. And for teaching purposes! My goodness... My whole
perspective on IDL has changed in the past couple of
weeks. I don't think I will ever use a normal IDL
graphics window again! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Saved Visualizations
Posted by SonicKenking on Fri, 04 Feb 2011 04:05:35 GMT
View Forum Message <> Reply to Message

On Feb 4, 4:14 am, David Fanning <n...@dfanning.com> wrote:
> Folks,
>
> I had *no* idea how nice it is to save data visualizations.
> It opens up a whole new world of possibilities! I just
> saved a visualization on my Windows machine, e-mailed it
> to my Linux machine and opened it up. Now I'm looking
> at exactly the same visualization on both machines!
>
> Wow! E-mail one of these babies to your colleagues and
> they can be looking at *exactly* what you are looking
> at. And for teaching purposes! My goodness... My whole
> perspective on IDL has changed in the past couple of

> weeks. I don't think I will ever use a normal IDL
> graphics window again! :-)
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.idlcoyote.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

WoW!! This is awesome!

I often do quite a lot ad-hoc checks on my data, which are mainly just noodling around the data and plot them with improvised ideas. These kinda works are often difficult to track back. But this save/restore window helps a ton. It saves not only the plot but also the data with it.

Now the question I have is how I can easily get the data out of the save files? So I can do some further noodling based on the save files.

Subject: Re: Saved Visualizations
Posted by David Fanning on Fri, 04 Feb 2011 04:21:26 GMT
View Forum Message <> Reply to Message

SonicKenking writes:

> Now the question I have is how I can easily get the data out of the
> save files? So I can do some further noodling based on the save files.

Would you like to work on Coyote Graphics Development?
Coyote already spends half the day grumbling about
"slave labor for less than slave wages." We could probably
use some help. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Saved Visualizations
Posted by SonicKenking on Fri, 04 Feb 2011 09:54:23 GMT
View Forum Message <> Reply to Message

> Would you like to work on Coyote Graphics Development?
> Coyote already spends half the day grumbling about
> "slave labor for less than slave wages." We could probably
> use some help. :-)
>

I'd like to help out if possible. Let me know what I can do.

Subject: Re: Saved Visualizations
Posted by David Fanning on Fri, 04 Feb 2011 12:56:44 GMT
View Forum Message <> Reply to Message

SonicKenking writes:

> I'd like to help out if possible. Let me know what I can do.

Well, since you are interested in noodling around with
save files, why don't you write a routine that can
allow the user to view and unpack the commands in the
save file?

There is already a preliminary "viewer" in the commands
LIST method. But I can see a program that "unpacks" a
command to produce variables at the main IDL level containing
all the data. Perhaps each command could be put into a
structure variable that was returned to the main IDL level.

```
cmd -> {p1:cmd.p1, p1:cmd.p2, ..., NLevels:cmd.extra.nlevels}
```

Such a thing could be very useful because it would
allow users to "recover" data from previous visualizations.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Saved Visualizations
Posted by SonicKenking on Sat, 05 Feb 2011 07:22:10 GMT
View Forum Message <> Reply to Message

> Well, since you are interested in noodling around with
> save files, why don't you write a routine that can
> allow the user to view and unpack the commands in the
> save file?
>
> There is already a preliminary "viewer" in the commands
> LIST method. But I can see a program that "unpacks" a
> command to produce variables at the main IDL level containing
> all the data. Perhaps each command could be put into a
> structure variable that was returned to the main IDL level.
>
>   cmd -> {p1:cmd.p1, p1:cmd.p2, ..., NLevels:cmd.extra.nlevels}
>
> Such a thing could be very useful because it would
> allow users to "recover" data from previous visualizations.
>

Hi David,

Following your suggestions, I came up with a new method for the FSC_Window_Command class for creating the command struct variable in IDL main level. It is written using the LIST method as the template.

I also added an keyword option to FSC_CmdWindow::ListCommand. The method will also create the struct variable when the option is turned on.

The command struct variable has the data structure as follows:
cmdStruct = {command: 'cgContour', nparams: 1, type: 0, p1: data, keywords: {nlevel: 10, fill: 1}}

I also wrote a small procedure, which takes the cmdStruct as input parameter and execute it. The struct variable can be saved as IDL source files (*.pro files) with proper output format. So it almost sounds like another way (ASCII files) for saving the visualization. It is quite interesting.

Here is the method, FSC_Window_Command::CreateCommandStruct, for
create the command struct variable in IDL main level.

```
PRO FSC_Window_Command::CreateCommandStruct, structName, Quiet=quiet

   Compile_Opt idl2

   ; Error handling.
   Catch, theError
   IF theError NE 0 THEN BEGIN
      Catch, /CANCEL
      void = Error_Message()
      RETURN
   ENDIF

   ; Struct variable name
   IF N_Elements(structName) EQ 0 THEN structName='cmd'

   cmdString = self.command
   cmdStruct = Create_Struct('Command', cmdString, 'nparams',
self.nparams, 'type', self.type)
   CASE self.nparams OF
      0:
      1: cmdStruct = Create_Struct(cmdStruct, 'p1', *self.p1)
      2: cmdStruct = Create_Struct(cmdStruct, 'p1', *self.p1, 'p2',
*self.p2)
      3: cmdStruct = Create_Struct(cmdStruct, 'p1', *self.p1, 'p2',
*self.p2, 'p3', *self.p3)
   ENDCASE
   IF Ptr_Valid(self.keywords) THEN BEGIN
      cmdStruct = Create_Struct(cmdStruct, 'keywords',
*self.keywords)
   ENDIF

   ; Copy the variable to the MAIN level
   (Scope_VarFetch(structName, /Enter, Level=1)) =
Temporary(cmdStruct)
   IF NOT Keyword_Set(quiet) THEN $
      PRINT, 'Created command struct variable ', structName, ' in
IDL $MAIN level.'

END
```

 ## ################

Here is the modified version of FSC_CmdWindow::ListCommand for adding
the option to create the command struct variable. The changes are only

adding three new lines.

```
PRO FSC_CmdWindow::ListCommand, cmdIndex,
CREATECOMMANDSTRUCT=createCommandStruct

   ; List the commands in the command window.

   Compile_Opt idl2

   ; Error handling.
   Catch, theError
   IF theError NE 0 THEN BEGIN
      Catch, /CANCEL
      void = Error_Message()
      RETURN
   ENDIF

   createCommandStruct = Keyword_Set(createCommandStruct)

   ; How many commands are there?
   count = self.cmds -> Get_Count()

   IF N_Elements(cmdIndex) EQ 0 THEN BEGIN
      FOR j = 0, count-1 DO BEGIN
         thisCmdObj = self.cmds -> Get_Item(j, /DEREFERENCE)

         ; Preface the commands with their index number.
         thisCmdObj -> List, StrTrim(j,2) + '.'

         ; Create the command struct
         IF createCommandStruct THEN thisCmdObj ->
CreateCommandStruct, 'cmd' + StrTrim(j,2)
      ENDFOR
   ENDIF ELSE BEGIN
      IF cmdIndex LT (count-1) THEN BEGIN
         thisCmdObj = self.cmds -> Get_Item(cmdIndex, /DEREFERENCE)

         ; Preface the commands with their index number.
         thisCmdObj -> List, StrTrim(cmdIndex,2) + '.'

         ; Create the command struct
         IF createCommandStruct THEN thisCmdObj ->
CreateCommandStruct, 'cmd' + StrTrim(cmdIndex,2)
      ENDIF ELSE Message, 'The command index is out of range of the
number of commands.'
   ENDELSE

END
```

################

The last one is the small procedure for execute the command in the struct variable. Note that the winid keyword does not work. The plot always goes into a new cgWindow. I am not sure why it is the case.

```
PRO cgRunCmdStruct, cmd, WinID=winid

   Compile_Opt idl2

   ; Error handling.
   Catch, theError
   IF theError NE 0 THEN BEGIN
      Catch, /CANCEL
      void = Error_Message()
      RETURN
   ENDIF

   void = Where(Tag_Names(cmd) EQ 'KEYWORDS', count)
   IF count NE 0 THEN extraKeywords = cmd.keywords

   CASE cmd.nparams OF
      0:
      1: cgWindow, cmd.command, cmd.p1, Method=cmd.type,
WinID=winid, _Extra=extraKeywords

      2: cgWindow, cmd.command, cmd.p1, cmd.p2, Method=cmd.type,
WinID=winid, _Extra=extraKeywords

      3: cgWindow, cmd.command, cmd.p1, cmd.p2, cmd.p3,
Method=cmd.type, WinID=winid, _Extra=extraKeywords
   ENDCASE

END
```

Subject: Re: Saved Visualizations
Posted by David Fanning on Sat, 05 Feb 2011 15:10:36 GMT
View Forum Message <> Reply to Message

SonicKenking writes:

> Following your suggestions, I came up with a new method for the
> FSC_Window_Command class for creating the command struct variable in
> IDL main level. It is written using the LIST method as the template.

Yes, this looks like it is going in the right direction.
You are very close to being ready to unpack those save files!

I have a full day planned away from my desk today, but
I'll study these more carefully when I get back. In the
future, feel free to e-mail the code directly to me. My
newsreader mangles long text lines. :-)

Cheers,

David

P.S. Just as a side note, the only person I know who
wrote an iTool from scratch (outside of ITTVIS employees),
spent nearly a *year* combing through the documentation
to understand the system before he wrote a single line of
code. Makes you think, doesn't it. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")
