Subject: Re: Saving & replaying direct graphics commands (was Re: Top 10 IDL Requests)
Posted by davidf on Mon, 31 Jul 2000 07:00:00 GMT

View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.cri.nz) writes:

> And then when you think about it, there already is a command editor in IDL,
> it's the one you use to write functions, procedures & scripts, so why not
> wrap up all your plotting commands in a single procedure a la XWINDOW.

Mark, just between you and me (please don't breathe a word
of this to the newsgroup), this seems a hell of a lot
simpler to me too. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Saving & replaying direct graphics commands (was Re: Top 10 IDL Requests)
Posted by davidf on Mon, 31 Jul 2000 07:00:00 GMT

View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.cri.nz) writes:

> A month or two ago I looked at David Fanning's XWINDOW procedure, as you
> have presumably done, and considered how to get past its major limitation:
> the graphics command must be a single procedure. I came up with the idea of
> a "window" that can accept & store a series of commands. This is what a
> typical creation/calling sequence would look like:
>
> owin = obj_new('xzwindow')
> owin->plot, 'plot', findgen(11)
> owin->plot, 'oplot', 0.1*findgen(11)^2, LINESTYLE=1
> owin->plot, 'oplot', sqrt(10.*findgen(11)), LINESTYLE=2, COLOR=3

I've written exactly this kind of thing (on more than
one occasion, I think). But I always get half finished
and then I'm distracted by something seemingly more

important. But I use my LinkedList object to store
the commands. I even had a "command editor" working,
where you could add or delete commands, move them
around, etc.

Perhaps between us we have more or less a complete
solution. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Saving & replaying direct graphics commands (was Re: Top 10 IDL Requests)
Posted by Mark Hadfield on Tue, 01 Aug 2000 07:00:00 GMT

View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
news:on7la3kkg0.fsf@cow.physics.wisc.edu...
>
> "Mark Hadfield" <m.hadfield@niwa.cri.nz> writes:
>
>> Once a direct graphics command has sent output to an output device, the
only
>> "memory" IDL has of that command is the changed state of the output
device.
>> At that point the system (or the user) has two ways of recreating the
output
>> to a different device:
>>
>> 1. Switch devices & re-issue the same commands
>> 2. Read the output back off the device and send it to the new device.
>> ...
>
> 3. Have the direct graphics window itself store the required data to
>   reproduce the output, and the ability to redirect to a new device.
>
> And I am totally serious; this is what I hacked up with XFWINDOW,
> which puts a "print" button on any direct graphics window under Unix.
> It's a hack because IDL doesn't provide enough documented

> functionality to achieve the full effect. I had to go stealth. :-)

That's amazing, Craig. I have checked out the code for XFWINDOW and XFILTER (I can't test them because I don't have access to any of the supported Unices) and I am very impressed. I didn't know it was possible to do stuff like that!

A month or two ago I looked at David Fanning's XWINDOW procedure, as you have presumably done, and considered how to get past its major limitation: the graphics command must be a single procedure. I came up with the idea of a "window" that can accept & store a series of commands. This is what a typical creation/calling sequence would look like:

```
owin = obj_new('xzwindow')
owin->plot, 'plot', findgen(11)
owin->plot, 'oplot', 0.1*findgen(11)^2, LINESTYLE=1
owin->plot, 'oplot', sqrt(10.*findgen(11)), LINESTYLE=2, COLOR=3
```

Here an xzwindow is an object that includes a direct graphics window and a container for "command objects". Each command object stores enough information so it can replay itself. Each time the plot method is called it constructs a command object from its arguments, plays it once and stores it in the command container. At any time--e.g. when the "print" button is pressed--the xzwindow object can replay all its command objects.

Of course for the old farts we could hide the xzwindow and the method calls behind a set of procedure calls thus:

```
xzwindow_init
xzwindow_plot, 'plot', findgen(11)
xzwindow_plot, 'oplot', 0.1*findgen(11)^2, LINESTYLE=1
xzwindow_plot, 'oplot', sqrt(10.*findgen(11)), LINESTYLE=2, COLOR=3
```

But then we could only have active xzwindow object at a time.

This is obviously not as transparent as your XFWINDOW, but it avoids one of the major limitations of the latter, i.e. the failure to record operations that affect the state of the graphics system but do not draw to a window. It's up to the user to decide which commands should be recorded and which not. E.g. consider the following

```
owin = obj_new('xzwindow')
owin->plot, '!p.linestyle = 0', /EXECUTE
owin->plot, 'plot', findgen(11)
owin->plot, '!p.linestyle = 1', /EXECUTE
owin->plot, 'oplot', 0.1*findgen(11)^2
owin->plot, '!p.linestyle = 2 & !p.color = 3', /EXECUTE
owin->plot, 'oplot', sqrt(10.*findgen(11))
```

Here the EXECUTE keyword directs the plot method it to create an "execute" command object rather than a "call_procedure" one.

I haven't fully implemented this--I got as far as writing the command-object class and building a single-command window, but the latter is still less complete than David's XWINDOW. I'm not sure how useful the multi-command version would be because it's not all that difficult to bundle a series of graphics commands into a single subroutine. Anyway, any and all opinions on this approach are welcome.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield/
National Institute for Water and Atmospheric Research
PO Box 14-901, Wellington, New Zealand