Subject: Re: Tip: how to mix object gui with command line
Posted by Mirko Vukovic on Thu, 23 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.12c81f5e6aecc9239899c7@news.frii.com>,
davidf@dfanning.com (David Fanning) wrote:
> Karri Kaksonen (karri.kaksonen@picker.fi) writes:
>
>> I just thought of dropping a line about a discovery I made.
>> This may be old news for all old timers but as it was new for
>> me I thought of sharing this idea.
>>
>> When you build an application as an object like:
>> o=obj_new('MyApplication')
>>
>> and you start up the graphical user interface with buttons etc.
>> o->draw
>>
>> then I suddenly notice that there is something weird going on
>> and want to have a look at my data from the command line.
>>
>> In a widget-program I would have to quit the program and start
>> debugging. But in an object program I can leave the program
>> running and just fetch the data.
>>
>> The key is in coding in methods for accessing private stuff like:
>>
>> function MyApplication::getdata
>>   return *self->data
>> end
>>
>> then I can just click on the IDL> command line and write:
>> a=o->getdata()
>> and continue to work on it on the command line.
>
> I think you meant "RETURN, *self.data".
>
> But in any case, this is certainly possible to do. I would
> be just a bit careful with it, however, since it is quite easy
> to completely defeat the whole purpose of objects, which
> is to encapsulate the data and the methods that work on
> the data inside the object, out of the view of the rest
> of the world.
>
> For example, you can easily write a GetDataPointer method:
>
>   FUNCTION JUNKER::GetDataPointer
>   RETURN, self.data

>   END
>
> Now the outside world can muck around with the data *inside*
> the object. Oh, dear! Keep in mind that just because something
> is *possible* doesn't mean it's always a good idea. :-)
>
> Cheers,
>
> David
>
> --

I have an obj object that is inherited by pretty much any object that is
not meant to be inherited. This object has the Debug method that
consists of a single ``stop'' statement. Calling it stops the execution
with self ready to be examined.

Mirko