Subject: Re: How do I save parameters for next run?
Posted by Rick Towler on Fri, 02 Apr 2004 00:09:06 GMT
View Forum Message <> Reply to Message

"Yunxiang Zhang" wrote...

> Sorry for the confusion. I just want my stand alone application to have a
> small "memory" interactively. For example, I have a program, say,
> 'mypro.sav' and needs user to input a series of parameters, say p1, p2
> ...pn through an input gui. Then the program runs perfect. But then if
> the next run the user only need to change a small fraction of the
> parameters, it would be nice that my program can have a small memory so
> that my gui displays the parameters input last time and the user need
> only change maybe one parameter and press 'enter' to continue the
> rest of the calculations.
>
> The idea comes from my colleagues and I still can't think of a way without
> creating some additional log file. Maybe somebody can tell me how to hack
> the .sav file so that i can dock some data into a specific region without
> breaking the .sav file.

I am assuming you mean between IDL VM sessions. For instance, you run the application today, close it, then next week you run it again and you want those previous parameters to be "sticky".

You could hack the .sav file but why waste the time? I mean, I am all about hacks like this but... You could have the same functionality using a simple log file in minutes.

But if you are dead set against a second file, then I would try adding a function that returns your parameters which you would use to set the defaults when the application launches. Something simple like:

```
function setParms

  p1 = 0.1
  p2 = 0.2
  p3 = 4.5

  RETURN, [p1,p2,p3]

end
```

Then I would add an updateParms procedure that opens your .sav file and given an array of offsets writes values into that file. Since you won't know the offsets yet just make them up.

Compile your .sav file and with your favorite hex editor go looking for your

parameters. Note the offsets, then go back to your IDL code and change the offsets in your updateParms pro.

Then compile it and hope that the offsets haven't changed.

A slightly more brutish approach would be to simply create a dummy function that explicitly defines a large array. Then just find the offset to the beginning of the data and read/write your parameters in there.

This of course will probably not work if you have compressed your .sav file. You'll also have to figure out path and filename details. And then there are endianess issues. I don't know how the .sav files deal with this.

Have fun.

-Rick
