Developer

Zach Forsyth

COLLABORATORS			
	TITLE : Developer		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Zach Forsyth	October 23, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Developer

1.1 Imp Professional Development Guide

```
                    ************************************************************ ↩

   *           Imp Professional Module Development Docs           *
   ************************************************************
```

Okay, first off this is very very preliminary documentation. This file will attempt to explain how to write a module for my Imp Professional DM's assistant. Good luck and thanks for the help!

ImpPro~shared~library

Sample~ImpPro~modules

Other~misc~junk

1.2 Imp Professional Shared Library

The shared library idea arose from the fact that most of the ↩ modules need similar functions (like random # generation). The library also allows for easy access to the ImpPro shared semaphore, which contains information that can be shared between modules.

The library contains the following functions:

Version 1

impSeedRand()

impRand()

impInterpretAndRoll()

```
impComputeDHTRSS()

impSetGameTimeRaw()

impGetGameTimeRaw()

impSetPartyXP()

impGetPartyXP()
    Version 2


impLogEvent()
    Version 3


impReadSnapshot()

impWriteSnapshot()
    Version 4


impArexxSelector()

impLaunchCommand()

impLaunchCommandWait()
    Version 5


impReturnStrSTEM()

impReturnNumSTEM()

impGotoFileLabel()
```

1.3 Seed the random # generator

```
int  impSeedRand(long);
```

This function seeds the random # generator. If a non-NULL value is passed to impSeedRand, it will use that value as a seed. If NULL is passed as an argument, the current time will be used as the seed. The return value is undefined.

Examples:

```
   impSeedRand(NULL);   // This seeds the rnd # generator based on time
   impSeedRand(3234);        // This uses 3234 as a seed
```

See Also:

impRand()

1.4 Return random number

```
int  impRand(int);
```

This function returns a number between 1 and the value passed into it.

Example:

```
value = impRand(4);   // This will return a value between 1 and 4
```

Be sure to seed the random # generator using impSeedRand before using this function for the first time.

See also:

impSeedRand()

1.5 Roll a die string

```
int  impInterpretAndRoll(char *);
```

This function takes a standard die string, and returns the result of rolling that string. The string must be null terminated.

Examples:

```
value = impInterpretAndRoll("1d4");
value = impInterpretAndRoll("3d100+3");
value = impInterpretAndRoll("5d100-30");
value = impInterpretAndRoll("3d6*2");
value = impInterpretAndRoll("d6");
value = impInterpretAndRoll("6");        // This will return 6
```

See also:

impRand()
,
impSeedRand()

1.6 Convert raw to to Days, turns, etc

```
int  impComputeDHTRSS(struct TimeStruct *);
```

This function fills in a TimeStruct based on the value of the raw field of the structure:

```
struct TimeStruct {
        int raw;
        int value[TS_PARTS]; };

Example:

   struct TimeStruct temp;

   temp.raw = impGetGameTimeRaw();   // Get the current raw game time
   impComputeDHTRSS(&temp);          // Fill in the Days,Hours,Turns
                                     // etc...
                                     // based upon the "raw" value
   printf("%d\n", temp.value[TS_DAYS]);
                                     // Print out the current # of days
                                     // passed

See also:

   ImpLib.h ,
                impGetGameTimeRaw()
```

1.7 Set current game time

```
int  impSetGameTimeRaw(int);

This function sets the current raw game time.  Only the clock module should
do this!
```

1.8 Get current game time

```
                int  impGetGameTimeRaw(void);

This function returns the current raw game time.  This number is
meaningless unless you run it through impComputeDHTRSS().

See also:


                impComputeDHTRSS()
```

1.9 Set running XP total

```
                int  impSetPartyXP(int);

This function sets the current party XP value.  The encounter module uses
this to communicate with the experience module.  ImpPro's shared semaphore
is locked and written to with this function.

See also:
```

impGetPartyXP()

1.10 Get running XP total

```
int  impGetPartyXP(void);
```

Get the current cumulative party XP. The experience module checks this value when attempting to import XP from the encounter module. The return value is the current party XP.

See also:

impSetPartyXP()

1.11 Log a game event via GameLog

```
BOOL impLogEvent(UBYTE *event, ULONG id)
```

This function logs an event in the Game Log module. This is attempted via messaging and will fail if Game Log's message port is not up and running. The event is passed in via a pointer to a string no longer than 80 characters. If all goes well, TRUE is returned. FALSE indicates failure.

As of version 5, an ID is now required that indicates which module has sent the logging event.

Example:

```
impLogEvent("Fizzbin has died", MOD_CHARACTER);
```

1.12 Read a snapshot from file

```
BOOL impWriteSnapshot(struct ImpSnapshot *)
```

This function saves snapshot information to the shared snapshot file. This information allows you to record the size and position of your module's window(s). Fill in the is_ID field before calling this function with your module unique window ID.

Example:

```
struct ImpSnapshot ssMain;

ssMain.is_ID = MAINWIN_ID;
GetAttr(WINDOW_Bounds, WO_Window, (ULONG *)&ssMain.is_Bounds);
impWriteSnapshot(&ssMain);
```

See also:

impReadSnapshot()
, struct~ImpSnapshot

1.13 Write a snapshot to file

BOOL impReadSnapshot(struct ImpSnapshot *)

This function searches the shared snapshot file for your module's snapshot information and copies it into the structure you supply. Fill in the is_ID field before calling this function.

Example:

```
struct ImpSnapshot ssMain;

ssMain.is_ID = MAINWIN_ID; // Our ID is ULONG, generated by MAKE_ID
if (impReadSnapshot(&ssMain))
   SetAttrs(WO_Window, WINDOW_Bounds, &ssMain.is_Bounds, TAG_END);
```

See also:

impWriteSnapshot()
, struct~ImpSnapshot

1.14 Bring up the standard ARexx selector

BOOL impARexxSelector(STRPTR extension)

This function will bring up the ImpPro standard ARexx selector containing a list of ARexx scripts with the specified extension in the "ImpPro:Scripts" directory. When the user selects a script, it will be launched and the function will return immediately. TRUE is returned if the user launches a script, FALSE if the user cancels

Example:

```
impARexxSelector("dungeon");  // Bring up all of the Dungeon scripts
```

See also:

impLaunchCommand()
, impLaunchCommandWait()

1.15 Launch an synchronous task

```
BOOL impLaunchCommandWait(STRPTR commandstring)
```

This function will spawn a task using the System routine in dos.library. Include the entire path and any arguments. TRUE indicates success, FALSE indicates failure. This function returns when the command has completed.

Example:

```
impLaunchCommandWait("SYS:Utilites/Multiview PUBSCREEN IMP.SCREEN");
```

See also:

impLaunchCommand()

1.16 Launch an asynchronous task

```
BOOL impLaunchCommand(STRPTR commandstring)
```

This function will spawn a task using the System routine in dos.library. Include the entire path and any arguments. TRUE indicates success, FALSE indicates failure. This function returns immediately.

Example:

```
impLaunchCommand("SYS:Utilites/Multiview PUBSCREEN IMP.SCREEN");
```

See also:

impLaunchCommandWait()

1.17 Return a series of variables to ARexx

```
BOOL impReturnStrSTEM(struct RexxMsg *rxm, STRPTR basename, ULONG ↩
    numvars,
       UBYTE **varnames, UBYTE **varvalues);
```

This function will return an array of variables to an ARexx program. Pass in the RexxMsg from your port, the desired base name of the STEM variable, the number of variables, an array of pointers to UBYTES containing the names of the variables, and finally the values of the variables. The return value indicates success.

Example:

```
struct RexxMsg *rxm;
static UBYTE retbuf[2][80];
UBYTE *vars[] = {"FX0", "FY0", "FX1", "FY1"};
UBYTE *vals[2];
```

```
   sprintf(retbuf[0], "%ld", fx0);
   sprintf(retbuf[1], "%ld", fy0);
   for (i = 0; i < 2; i++)
      vals[i] = retbuf[i];

   impReturnStrSTEM(rxm, "MySTEM", 2, vars, vals);
```

See also:

```
                impReturnNumSTEM()
```

1.18 Return a series of variables to ARexx

```
                BOOL impReturnNumSTEM(struct RexxMsg *rxm, STRPTR basename, ULONG ↩
                   numvars,
                      ULONG numvars, UBYTE **varvalues);
```

This function will return an array of variables to an ARexx program. Pass in the RexxMsg from your port, the desired base name of the STEM variable, the number of variables, and finally the values of the variables. The return value indicates success.

Unlike

```
                impReturnStrSTEM()
```

, this function does not allow you to name each STEM variable. Instead, the variable names will be the basename with an index value:

```
   <basename>.0
   <basename>.1
      ....
   <basename>.numvars - 1
```

Example:

```
   struct RexxMsg *rxm;
   static UBYTE retbuf[2][80];
   UBYTE *vals[2];

   sprintf(retbuf[0], "%ld", fx0);
   sprintf(retbuf[1], "%ld", fy0);
   for (i = 0; i < 2; i++)
      vals[i] = retbuf[i];

   impReturnStrSTEM(rxm, "MySTEM", 2, vals);
```

See also:

```
                impReturnStrSTEM()
```

1.19 Goto a file label

```
BOOL impGotoFileLabel(BPTR fp, STRPTR label);

This function rewinds file fp and searches for the specified file label.
File labels are simply placeholders in an ASCII data file in this form:

   ~ATTRIBUTES
   18
   13
   14
   16
   13
   ~END

If the search is successful, the function returns TRUE and the file
position is set to the line immediately following the label.  If the label
is not found, FALSE is returned and the file pointer is at the end of the
file.

Example:

   if (impGotoFileLabel("ATTRIBUTES"))
      FGets(fp, buffer, 80);
```

1.20 Example Imp Professional Modules

```
                The only difference between writing a normal C program and writing ↩
                    an
ImpPro module is that you need to check for ImpPro's public screen and open
your window there if it is available.  Otherwise, run the program on
workbench. If you wish to use the functions in "
                imppro.library
                " you must
open that as well.

The first module example uses BGUI, and can be found in the "BGUIModule.h"
and "BGUIModule.c" files in this archive.

The second module example uses GadTools, and can be found in the
"GTModule.h" and "GTModule.c" files in this archive.

I would prefer that all modules be written using BGUI by Jan.van.den.Baard.
The development archive for this excellent GUI engine can be found in the
"dev/gui" directory of the Aminet.  If you need any help getting started
using BGUI, just drop me a line and I'll be happy to help you.  Also be
sure to join the BGUI mailing list.
```

1.21 Other misc information

```
Of course I can always be contacted at fors0037@gold.tc.umn.edu if you have
any questions while writing a ImpPro module.
```

The only version of the final release that will contain copyrighted TSR material will be the version that is distributed via "ftp.mpgn.com"

When you have a working module, please send me a copy (with or without source) so I can check it out and release it to the other beta-testers.

Once again, thank you for your interest in writing an ImpPro module, and happy coding!!

1.22 Index

Index of database Developer.guide

Documents

```
                Set running XP total

                Write a snapshot to file
                Buttons

BGUIModule.c
BGUIModule.h
GTModule.c
GTModule.h

                impArexxSelector()

                impComputeDHTRSS()

                impGetGameTimeRaw()

                impGetPartyXP()

                impGotoFileLabel()

                impInterpretAndRoll()

                impLaunchCommand()

                impLaunchCommandWait()
                impLaunchCommandWait()
ImpLib.h

                impLogEvent()

                ImpPro~shared~library

                imppro.library

                impRand()

                impReadSnapshot()

                impReturnNumSTEM()

                impReturnStrSTEM()

                impSeedRand()

                impSetGameTimeRaw()

                impSetPartyXP()

                impWriteSnapshot()

                Other~misc~junk

                Sample~ImpPro~modules
                struct~ImpSnapshot
```