The System Monitor

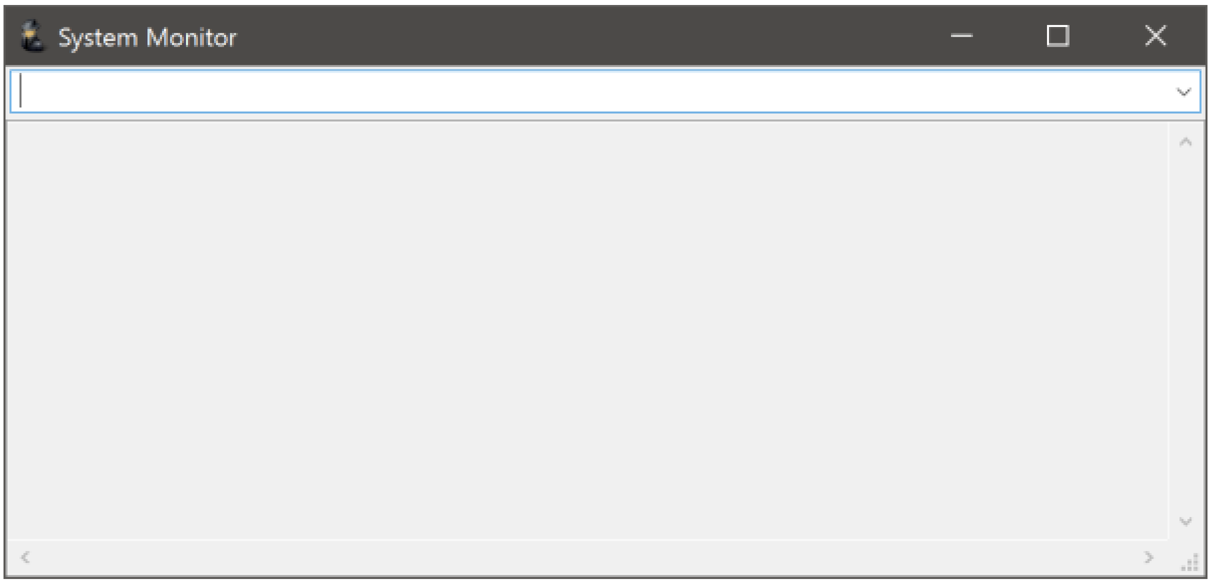
The System Monitor is a useful tool that allows a direct command-line interface into the Presentation Server. As well as allowing you to execute specific Basic+ stored procedures it also supports a set of intrinsic commands that expose common tasks too.

Opening the System Monitor

The System Monitor can be accessed in several ways:

1. By setting the "SM" command line switch to "1" when starting the Presentation Server.
2. By setting the "showMonitor" value to "1" in the RXI file when starting the Presentation Server.
3. By selecting the "System Monitor" item from the IDE "View" menu.
4. By using the Ctrl-F5 shortcut key combination in the IDE.
5. By using the SYSTEMMONITOR VISIBLE property with the Set_Property function in a Basic+ program at runtime.

When opened the System Monitor presents a simple interface consisting of a command line at the top and a read-only text box that displays the results of any executed commands:



The System monitor remembers its position between sessions by storing the window placement data in the registry when it is closed.

Executing System Monitor commands

Using the tool is simple: enter the statement in the command line and press the "Enter" key to execute it.

Some things to bear in mind when using the command line are:

- An auto-complete mechanism is attached to the command line that includes the last 100 commands executed. This list can be shown by clicking the down-arrow button next to the command line, or the system will show matching items as you type.
    - This list is saved to the SYSLISTS table between sessions using a key structure of:

      ```
      <appid> "*" <userid> "*PS_COMMANDSTACK"
      ```

      E.g.

      ```
      EXAMPLES*DAFFYDUCK*PS_COMMANDSTACK
      ```

- Parameters passed to the command may be separated by spaces or by commas. E.g. the following two commands are equivalent:

  ```
  my_test_proc patient_data 100 0
  my_test_proc patient_data, 100, 0
  ```

- If a parameter includes spaces then it may be enclosed by matching quotes (single or double) instead:

  ```
  my_test_proc "some parameter"
  my_test_proc "something else's"
  my_test_proc 'something "quoted" string'
  ```

- If you wish to pass a null value for a parameter simply pass an empty quoted string like so:

  ```
  my_test_proc "" param2 "" "Parameter 4"
  my_test_proc "", param2, "", "Parameter 4"
  ```

- All text in the command line is converted to upper-case before it is executed. This includes any quoted text *unless* the command is prefixed with a "_" character to preserve it.

  E.g. This command will set the control's text using an upper case string:

  ```
  sp mywindow.edl_surname text "Duck"
  ```

  While this command will preserve the mixed-case:

  ```
  _sp mywindow.edl_surname text "Duck"
  ```

- If you wish to pass system delimiters to the command you can do so using a comma to replace the delimiter and using one or more matching sets of nested enclosing brackets ("[" and "]") to indicate which delimiter the comma represents.

  - For an @fm delimiter use a single set of brackets like so:

    ```
    sp mywindow.lst_ids list ["Item One","Item 2","Item 3"]
    ```

  - For an @vm delimiter use a double set of nested brackets like so:

    ```
    sp mywindow.edt_ids array [["Item One","Item 2","Item 3"]]
    ```

  - For an @svm delimiter use a triple set of nested brackets like so:

    ```
    sp mywindow.edl_name font [[["Courier New", -13, 400]]]
    ```

  - … and so on down to @stm.

  This delimiter notation may be combined to pass parameters with more than one delimiter,

  e.g. for an EditTable ARRAY property:

  ```
  sp mywin.tbl_1 array [ [["Cell 1,1","Cell 2,1"]],[["Cell 2,1","Cell 2,2"]] ]
  ```

- You may specify a previously executed command by entering a "." followed immediately by the index of the command in the saved list. For example, entering ".1" retrieves the last command entered, ".2" retrieves the command before that and so on.

Command results

If the command returns any results, they appear in the text area at the bottom of the window. Programs executed from the System Monitor command line may also update the results area by using the Basic+ Send_Dyn function.

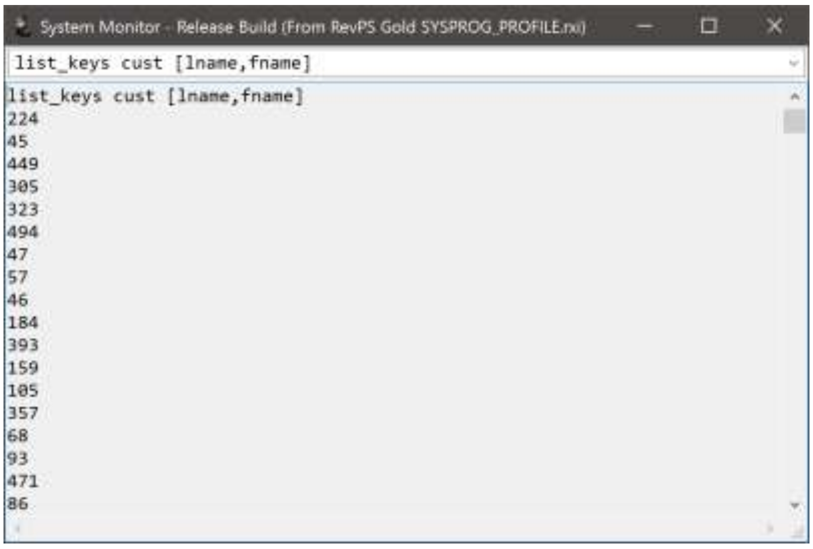
System Monitor commands

This section defines the intrinsic System Monitor commands available. These are:

Name	Description
ABOUT	Displays version and licensing information.
CFG	Displays information extracted from the RXI file.
CLL	Clears the list of previously executed commands.
CLS	Clears the results.
CMD	Opens a Windows Command Prompt.
DLG	Executes a form using the Dialog_Box function.
EM	Executes the Exec_Method function.
EXEC	Executes a form using the Start_Window function.
EVAL	Executes Basic+ statements.
GC	Issues a GarbageCollect operation and purges cached data.
GP	Executes the Get_Property function.
IDT	Displays a date in internal format.
LE	Excutes the Get_Repos_Entities function to return a list of matching repository entities.
LIST	Executes an RLIST "LIST" statement.
LO	Executes the SYSTEM OBJECTLIST method.
LOGTO	Closes the current application and opens another one.
OFF	Shuts down the Presentation Server.
QUIT	Synonym for the OFF command.
RUN	Executes the specified Basic+ stored procedure.
RXI	Synonym for the CFG command.
SHL	Execute the Windows ShellExecute function to open a document or run another Windows program.
SP	Executes the Set_Property function.
STOP	Closes a specified window, or all windows.

Any other commands entered are treated as the name of a Basic+ stored procedure and the System Monitor will try to execute them as such.

E.g. Executing the LIST_KEYS stored procedure for the CUST table, sorting by the LNAME and FNAME columns:


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
list_keys cust [lname,fname]
```

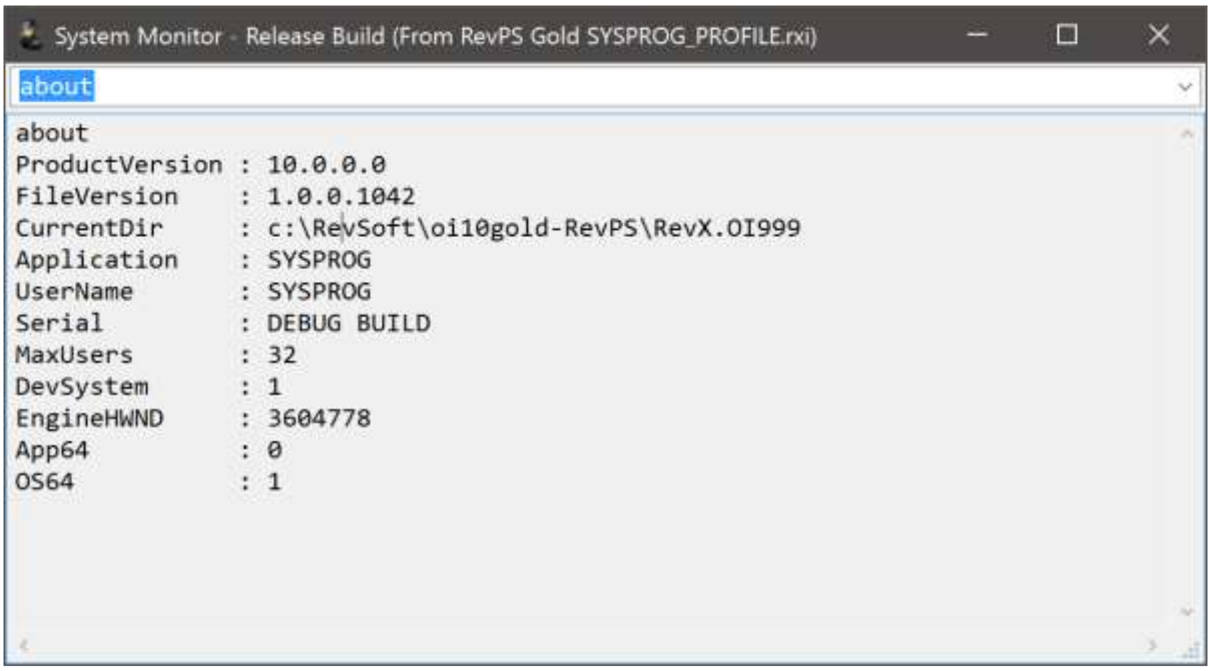
```
list_keys cust [lname,fname]
224
45
449
305
323
494
47
57
46
184
393
159
105
357
68
93
471
86
```

ABOUT

This command displays the following information in the results area:

- Product version
- File version
- Current directory
- Application ID
- User name
- Serial number
- Maximum licensed users
- Development System Flag
- Engine Window Handle
- A flag denoting if the Presentation Server is a 64-bit version
- A flag denoting if the operating system is 64-bit.

E.g.


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

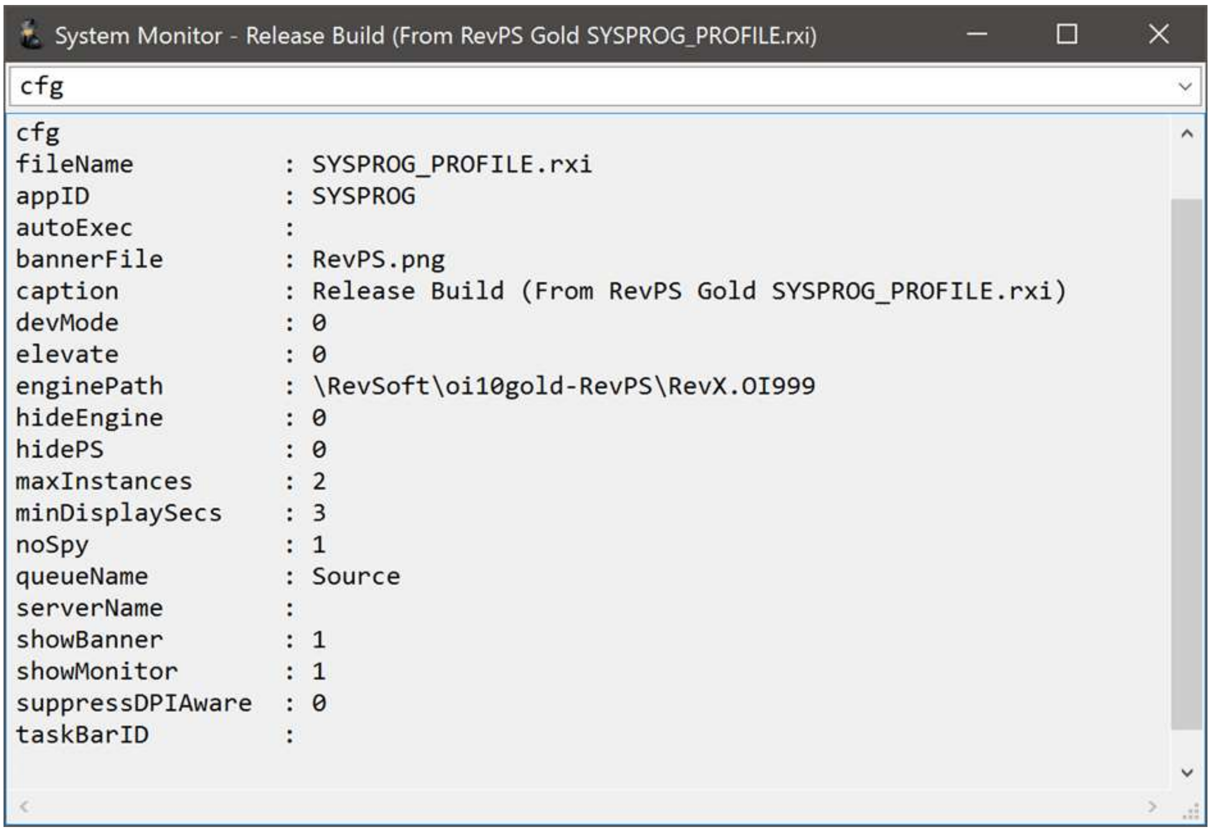
```
about
```

```
about
ProductVersion : 10.0.0.0
FileVersion    : 1.0.0.1042
CurrentDir     : c:\RevSoft\oi10gold-RevPS\RevX.OI999
Application    : SYSPROG
UserName       : SYSPROG
Serial         : DEBUG BUILD
MaxUsers       : 32
DevSystem      : 1
EngineHWND     : 3604778
App64          : 0
OS64           : 1
```

CFG

This command displays the *raw* configuration data extracted from the RXI file.

E.g.


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

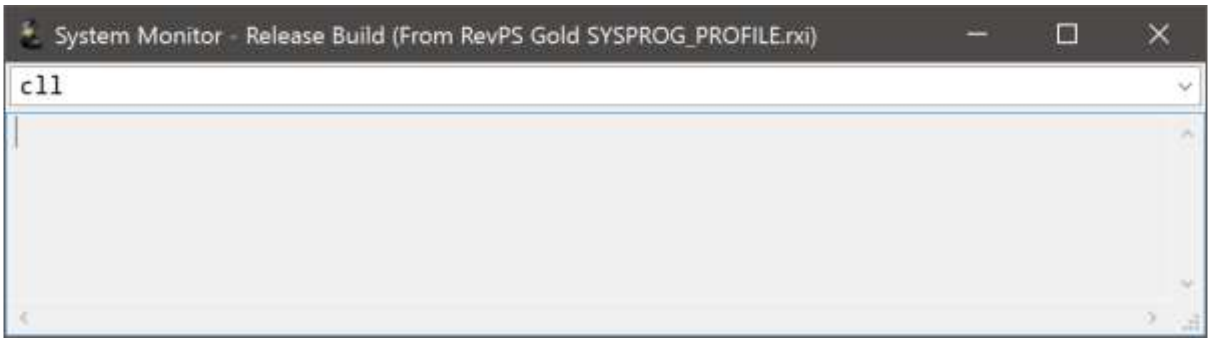
```
cfg
```

```
cfg
fileName           : SYSPROG_PROFILE.rxi
appID              : SYSPROG
autoExec           :
bannerFile         : RevPS.png
caption            : Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)
devMode            : 0
elevate            : 0
enginePath         : \RevSoft\oi10gold-RevPS\RevX.OI999
hideEngine         : 0
hidePS             : 0
maxInstances       : 2
minDisplaySecs     : 3
noSpy              : 1
queueName          : Source
serverName         :
showBanner         : 1
showMonitor        : 1
suppressDPIAware   : 0
taskBarID          :
```

CLL

This command clears the saved list of previously executed commands.

E.g.

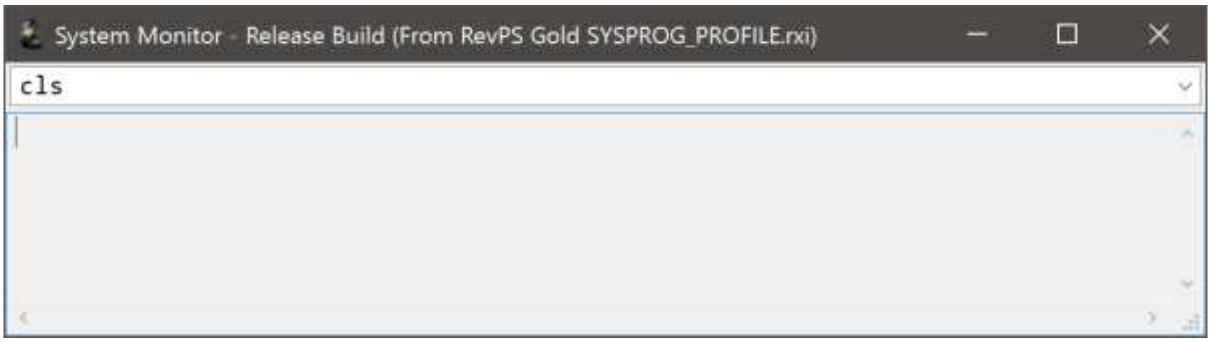

System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
cll
```

CLS

This command clears the results area.

E.g.

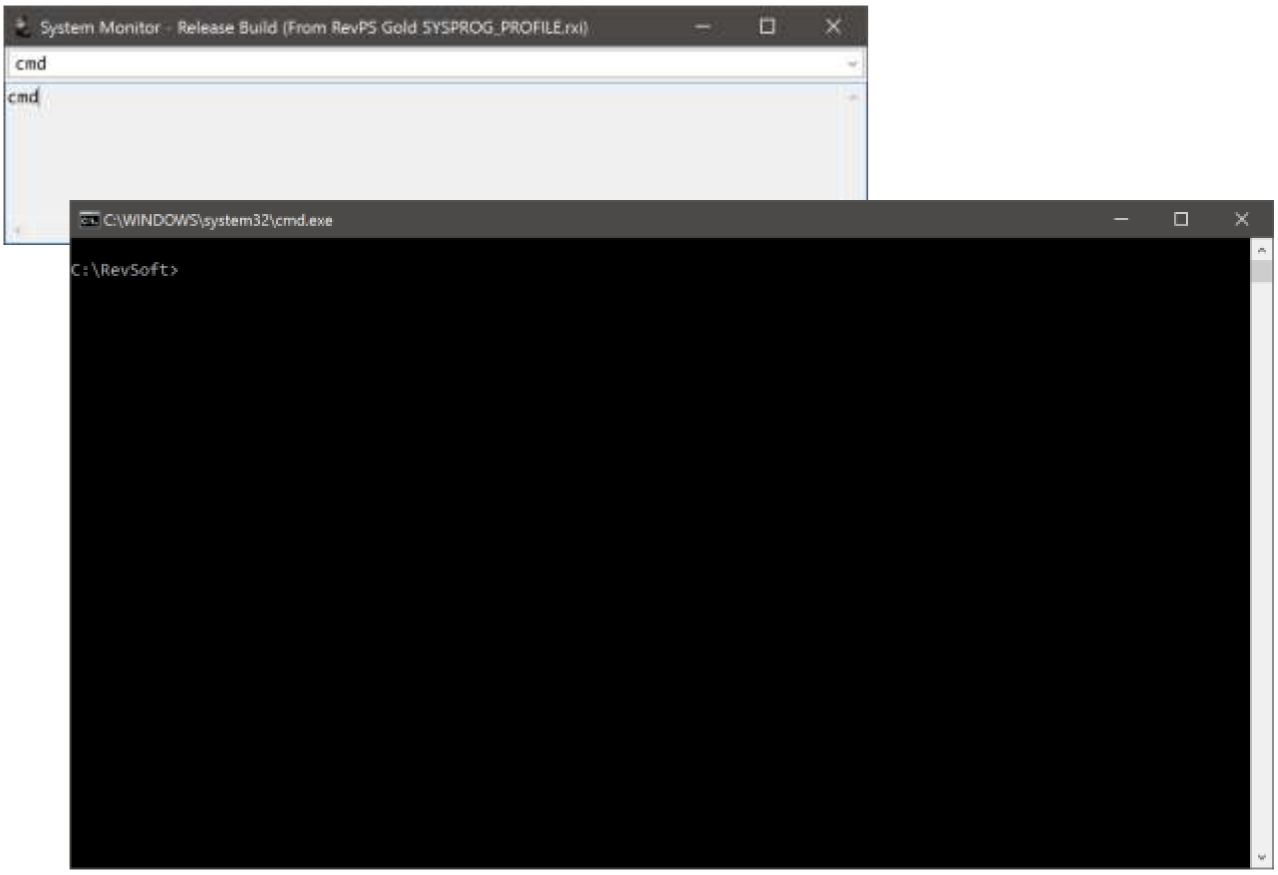

System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
cls
```

CMD

This command opens a Windows Command Prompt.

E.g.


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)
C:\WINDOWS\system32\cmd.exe

```
cmd
```

```
cmd
```

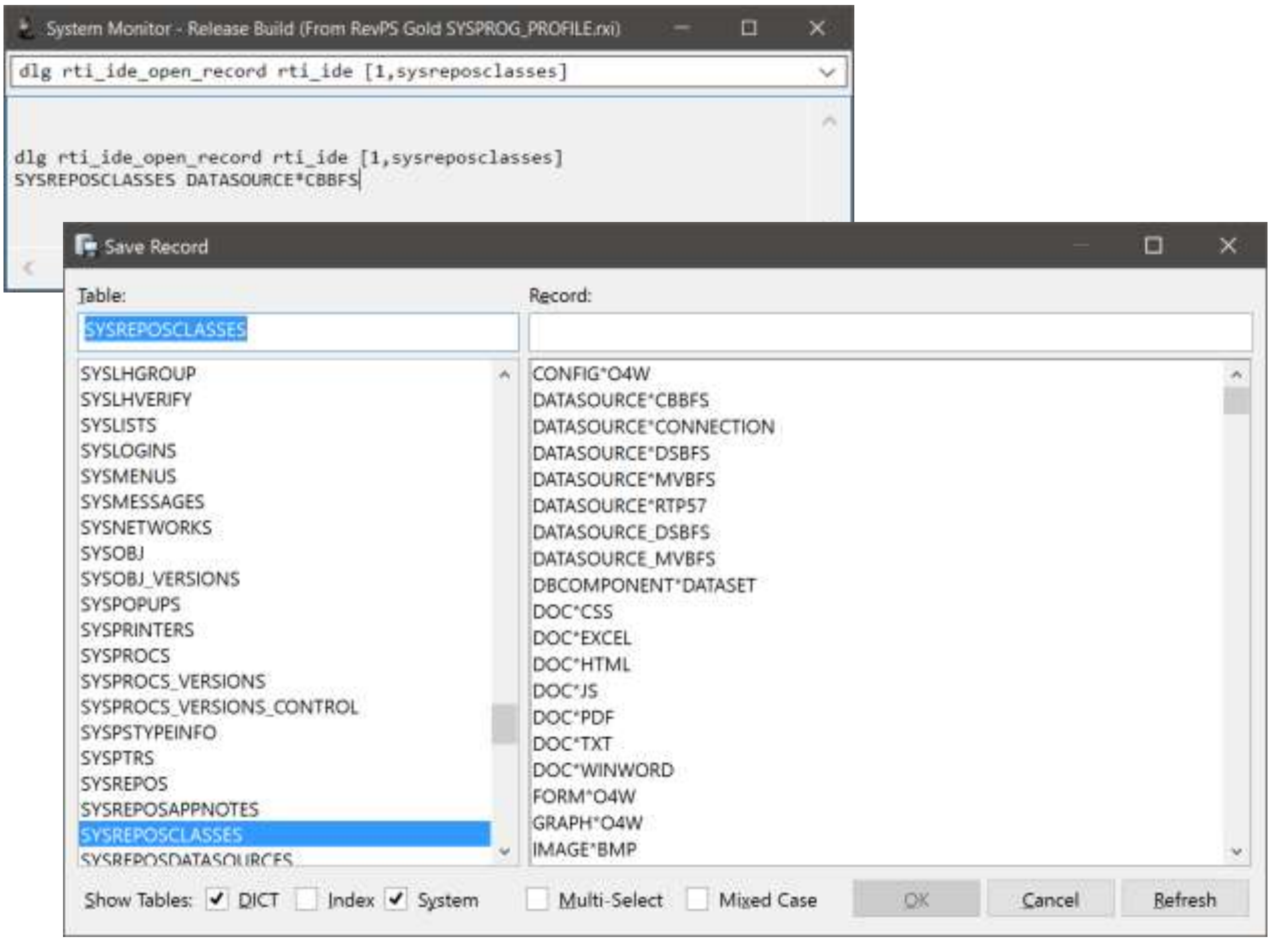
```
C:\RevSoft>
```

DLG

This command executes a form using the Dialog_Box stored procedure. The parameters passed are the same as for calling Dialog_Box in Basic+.

E.g.

Launch the RTI_IDE_OPEN_RECORD dialog box using RTI_IDE as the parent window and setting a mode of 1 and a table name of SYSREPOSCLASSES in the "CreateParam" parameter:



System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
dlg rti_ide_open_record rti_ide [1,sysreposclasses]
```

```
dlg rti_ide_open_record rti_ide [1,sysreposclasses]
SYSREPOSCLASSES DATASOURCE*CBBFS
```

Save Record

Table: SYSREPOSCLASSES

Record:

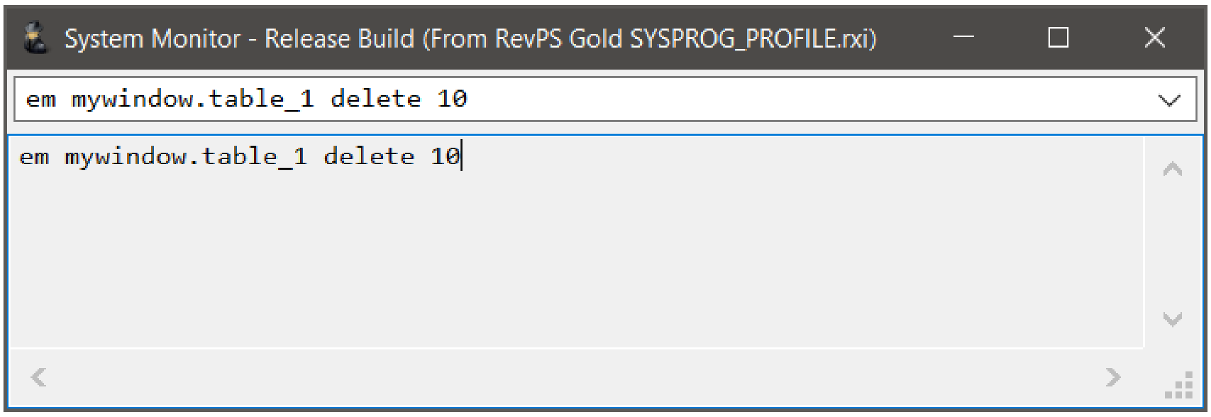
SYSLHGROUP	CONFIG*O4W
SYSLHVERIFY	DATASOURCE*CBBFS
SYSLISTS	DATASOURCE*CONNECTION
SYSLOGINS	DATASOURCE*DSBFS
SYSMENUS	DATASOURCE*MVBFS
SYSMESSAGES	DATASOURCE*RTP57
SYSNETWORKS	DATASOURCE_DSBFS
SYSOBJ	DATASOURCE_MVBFS
SYSOBJ_VERSIONS	DBCOMPONENT*DATASET
SYSPOPUPS	DOC*CSS
SYSPRINTERS	DOC*EXCEL
SYSPROCS	DOC*HTML
SYSPROCS_VERSIONS	DOC*JS
SYSPROCS_VERSIONS_CONTROL	DOC*PDF
SYSPSTYPEINFO	DOC*TXT
SYSPTRS	DOC*WINWORD
SYSREPOS	FORM*O4W
SYSREPOSAPPNOTES	GRAPH*O4W
SYSREPOSCLASSES	IMAGE*BMP
SYSREPOSDATASOURCES	

Show Tables: [✓ DICT] Index [✓ System] Multi-Select Mixed Case

EM

This command executes the Exec_Method procedure to invoke an object's method. The parameters passed are the same as for calling Exec_Method in Basic+.

E.g. Deleting the 10th row in an edit table:


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
em mywindow.table_1 delete 10
```

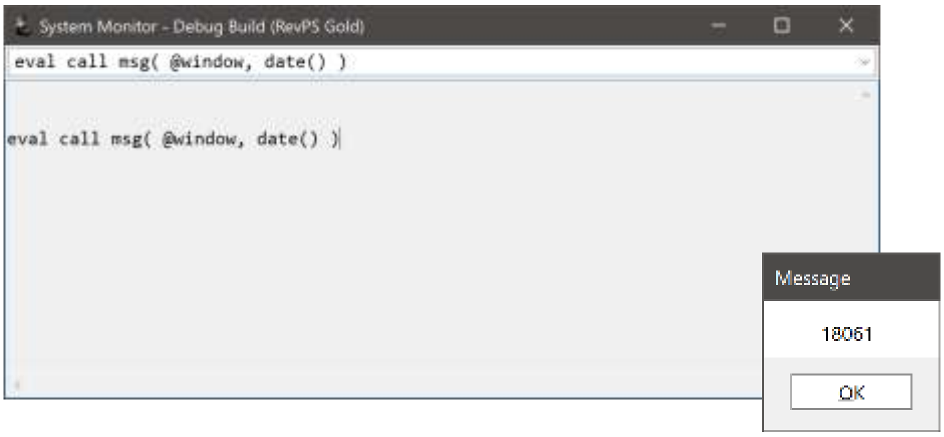
```
em mywindow.table_1 delete 10
```

EVAL

This command compiles and executes a Basic+ statement. Multiple statements should be delimited by a ";" character.

E.g.

Display a message box displaying the current date in internal format:



```
eval call msg( @window, date() )
```

```
eval call msg( @window, date() )
```

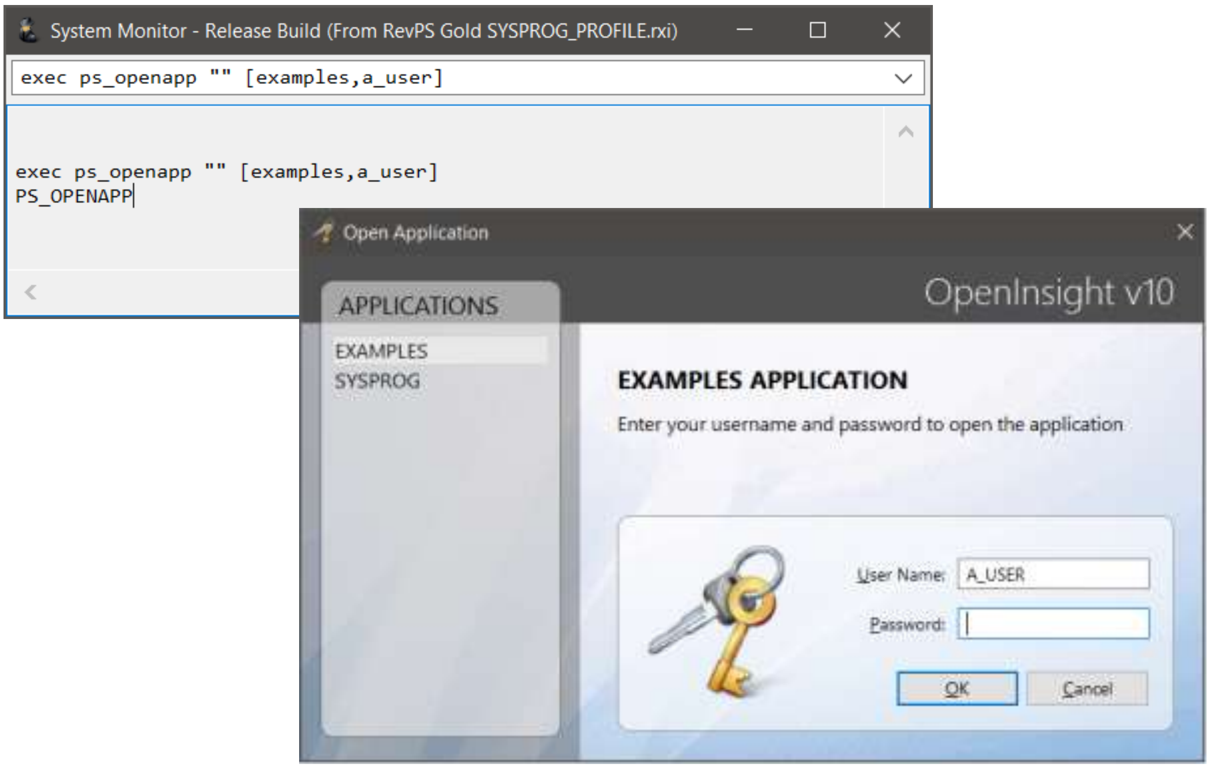
```
18061
```

EXEC

This command executes a form using the Start_Window stored procedure. The parameters passed are the same as for calling Start_Window in Basic+.

E.g.

Launch the PS_OPENAPP window with a null parent window and setting an Application ID of "EXAMPLES" and a Username of "A_USER" in the "CreateParam" parameter:



```
exec ps_openapp "" [examples,a_user]
```

```
exec ps_openapp "" [examples,a_user]
PS_OPENAPP
```

System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)
Open Application
OpenInsight v10
APPLICATIONS
EXAMPLES
SYSPROG
EXAMPLES APPLICATION
Enter your username and password to open the application
User Name: A_USER
Password:
OK
Cancel

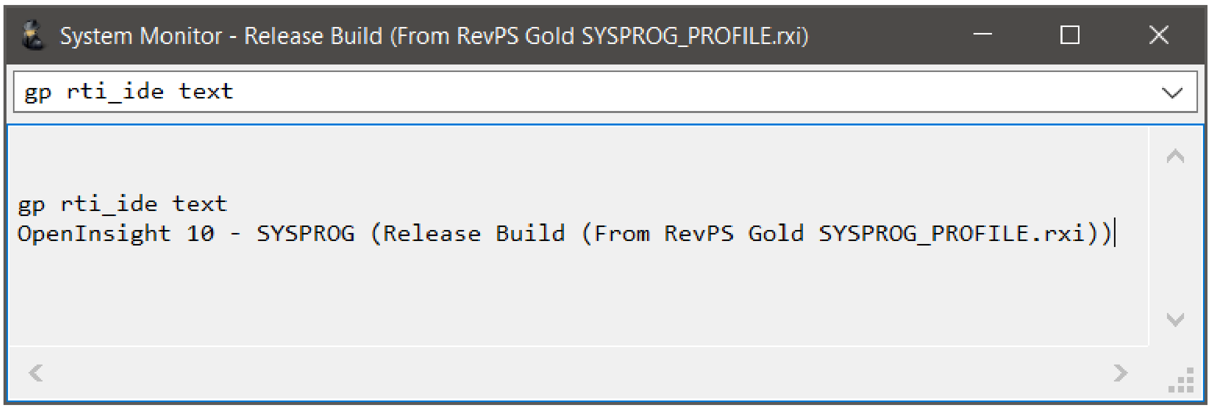
GC

This command executes a garbage collect instruction and executes the RTI_Purge_All stored procedure to clear out a range of cached system and IDE data.

GP

This command executes the Get_Property procedure to return the value of an object property. The parameters passed are the same as for calling Get_Property in Basic+.

E.g. Get the TEXT property of the RTI_IDE window:



```
gp rti_ide text
```

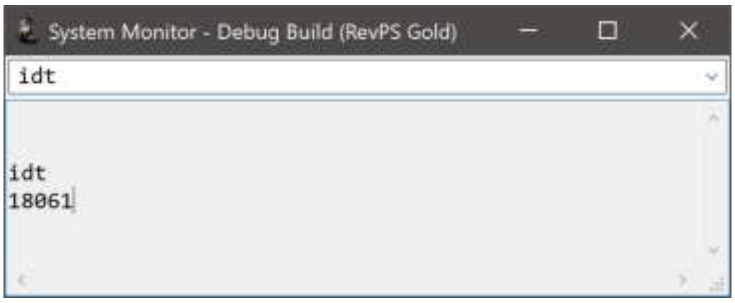
```
gp rti_ide text
OpenInsight 10 - SYSPROG (Release Build (From RevPS Gold SYSPROG_PROFILE.rxi))
```

System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

IDT

This command returns the internal format for a specified date, or if no date is specified the current date is used instead. The conversion is performed using the "D" format specifier for the Iconv operation, unless an "E" parameter is passed, in which case the "DE" format specifier is used instead.

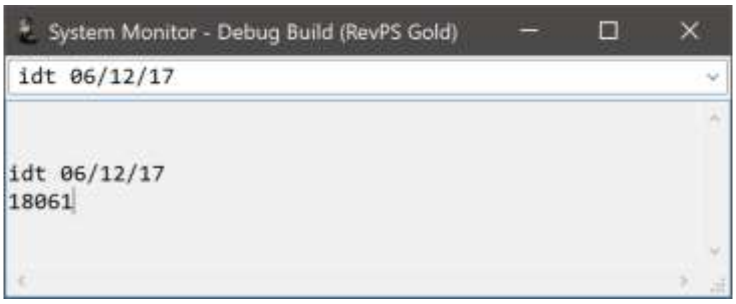
E.g. Return the current date in internal format:



```
idt
```

```
idt
18061
```

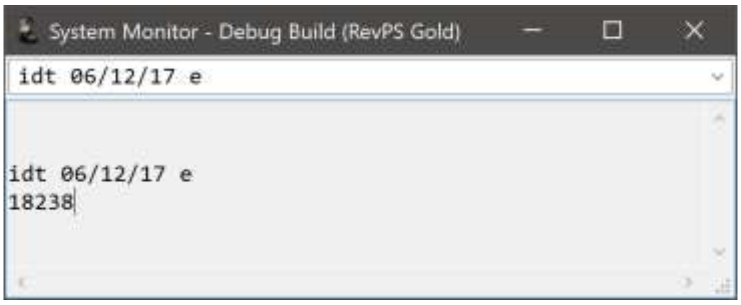
E.g. Return a specific date in internal format using the "D" format specifier



```
idt 06/12/17
```

```
idt 06/12/17
18061
```

E.g Return a specific date in internal format using the "DE" format specifier



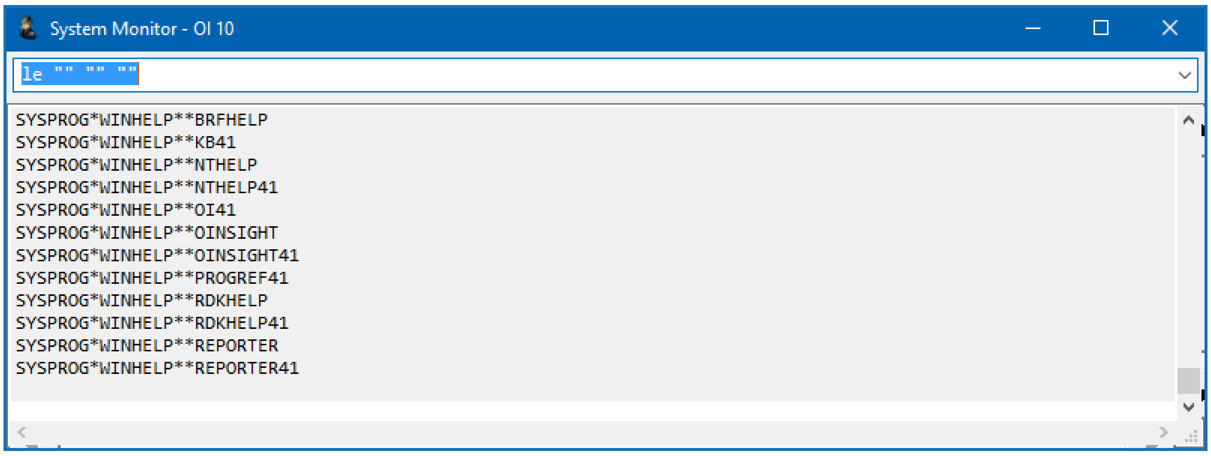
```
idt 06/12/17 e
```

```
idt 06/12/17 e
18238
```

LE

This command returns a list of Repository entities matching the passed criteria by executing the GET_REPOS_ENTITIES stored procedure. The parameters passed are the same as for the stored procedure.

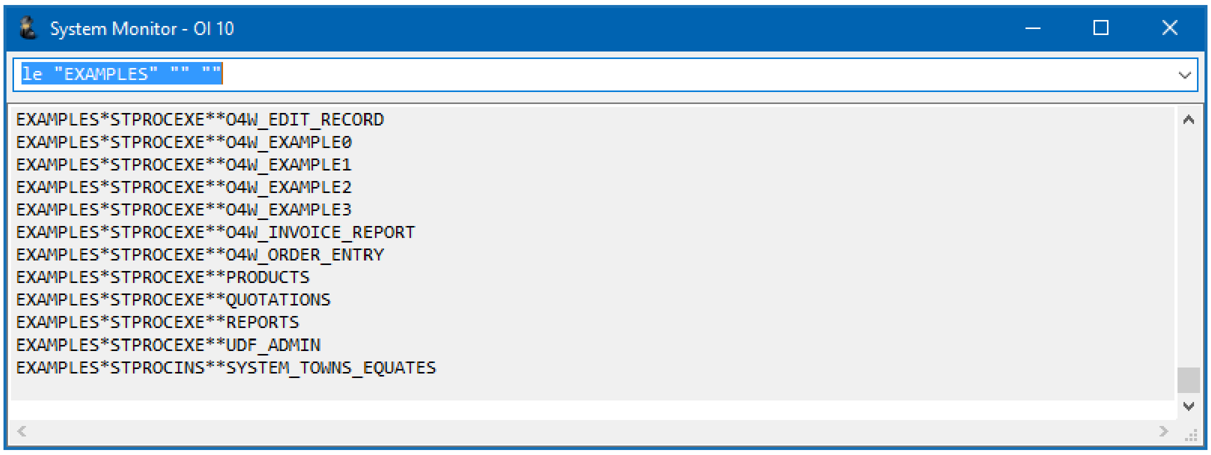
E.g. List all entities in the system:


System Monitor - OI 10

```
le "" "" ""
```

```
SYSPROG*WINHELP**BRFHELP
SYSPROG*WINHELP**KB41
SYSPROG*WINHELP**NTHELP
SYSPROG*WINHELP**NTHELP41
SYSPROG*WINHELP**OI41
SYSPROG*WINHELP**OINSIGHT
SYSPROG*WINHELP**OINSIGHT41
SYSPROG*WINHELP**PROGREF41
SYSPROG*WINHELP**RDKHELP
SYSPROG*WINHELP**RDKHELP41
SYSPROG*WINHELP**REPORTER
SYSPROG*WINHELP**REPORTER41
```

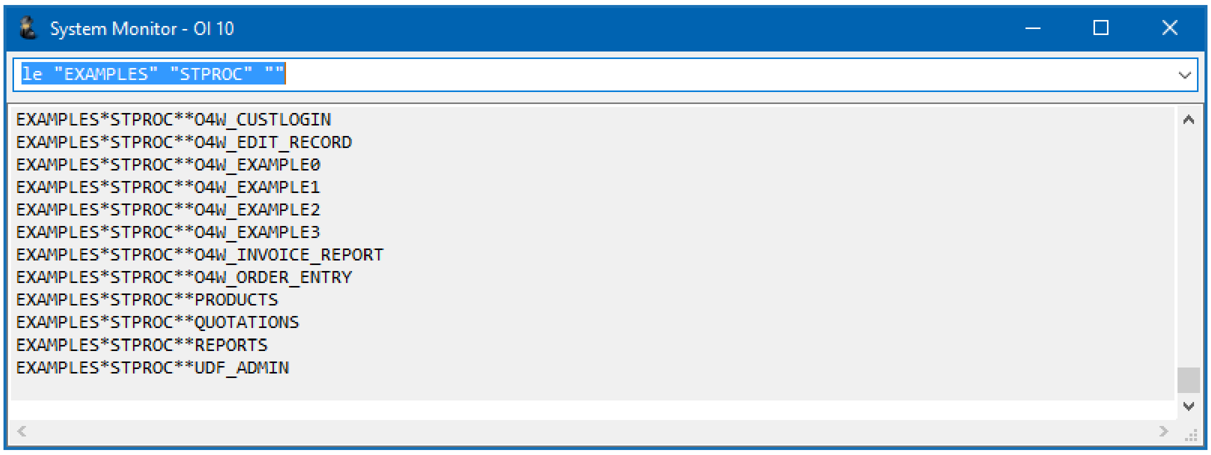
E.g. List all entites in the EXAMPLES application:


System Monitor - OI 10

```
le "EXAMPLES" "" ""
```

```
EXAMPLES*STPROCEXE**O4W_EDIT_RECORD
EXAMPLES*STPROCEXE**O4W_EXAMPLE0
EXAMPLES*STPROCEXE**O4W_EXAMPLE1
EXAMPLES*STPROCEXE**O4W_EXAMPLE2
EXAMPLES*STPROCEXE**O4W_EXAMPLE3
EXAMPLES*STPROCEXE**O4W_INVOICE_REPORT
EXAMPLES*STPROCEXE**O4W_ORDER_ENTRY
EXAMPLES*STPROCEXE**PRODUCTS
EXAMPLES*STPROCEXE**QUOTATIONS
EXAMPLES*STPROCEXE**REPORTS
EXAMPLES*STPROCEXE**UDF_ADMIN
EXAMPLES*STPROCINS**SYSTEM_TOWNS_EQUATES
```

E.g. List all STPROC entities in the EXAMPLES application:

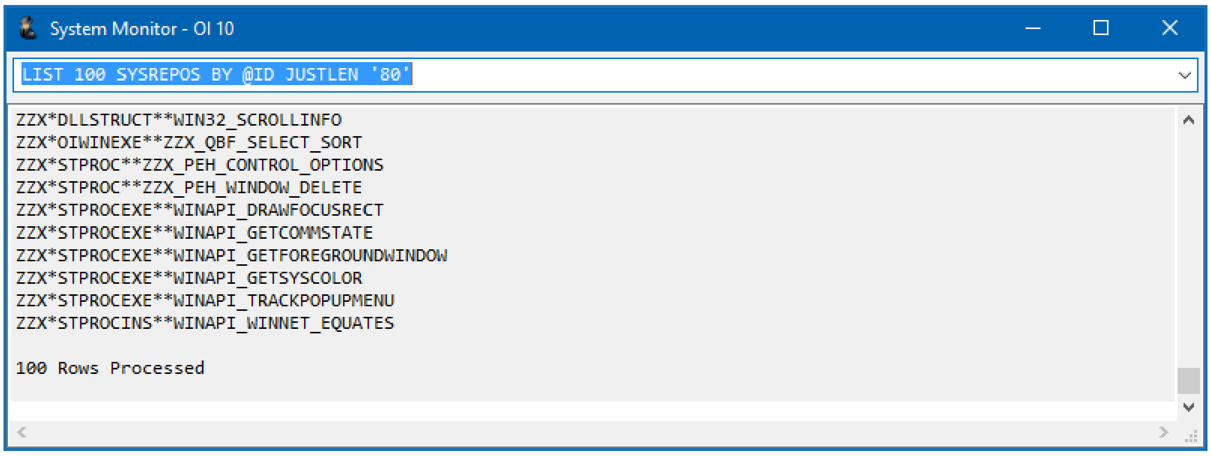

System Monitor - OI 10

```
le "EXAMPLES" "STPROC" ""
```

```
EXAMPLES*STPROC**O4W_CUSTLOGIN
EXAMPLES*STPROC**O4W_EDIT_RECORD
EXAMPLES*STPROC**O4W_EXAMPLE0
EXAMPLES*STPROC**O4W_EXAMPLE1
EXAMPLES*STPROC**O4W_EXAMPLE2
EXAMPLES*STPROC**O4W_EXAMPLE3
EXAMPLES*STPROC**O4W_INVOICE_REPORT
EXAMPLES*STPROC**O4W_ORDER_ENTRY
EXAMPLES*STPROC**PRODUCTS
EXAMPLES*STPROC**QUOTATIONS
EXAMPLES*STPROC**REPORTS
EXAMPLES*STPROC**UDF_ADMIN
```

LIST

This command executes an Rlist "LIST" statement and displays the results in the System Monitor:



```
LIST 100 SYSREPOS BY @ID JUSTLEN '80'
```

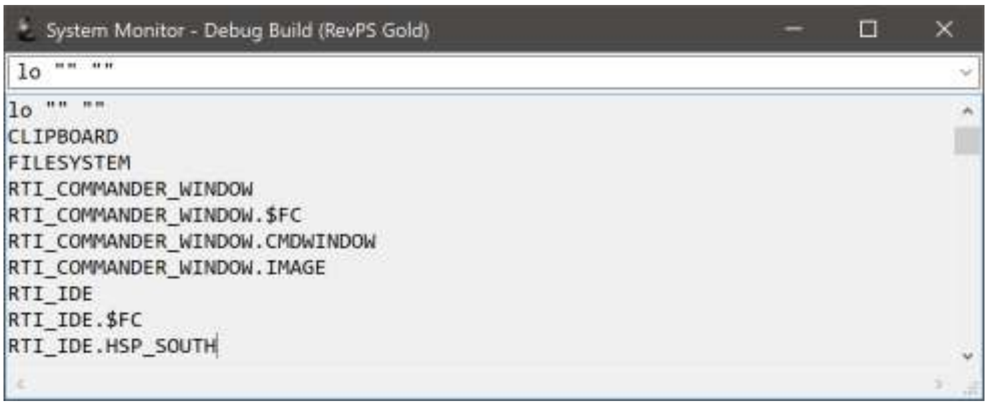
```
ZZX*DLLSTRUCT**WIN32_SCROLLINFO
ZZX*OIWINEXE**ZZX_QBF_SELECT_SORT
ZZX*STPROC**ZZX_PEH_CONTROL_OPTIONS
ZZX*STPROC**ZZX_PEH_WINDOW_DELETE
ZZX*STPROCEXE**WINAPI_DRAWFOCUSRECT
ZZX*STPROCEXE**WINAPI_GETCOMMSTATE
ZZX*STPROCEXE**WINAPI_GETFOREGROUNDWINDOW
ZZX*STPROCEXE**WINAPI_GETSYSCOLOR
ZZX*STPROCEXE**WINAPI_TRACKPOPUPMENU
ZZX*STPROCINS**WINAPI_WINNET_EQUATES

100 Rows Processed
```

LO

This command returns a list of objects matching the passed criteria by executing the SYSTEM OBJECTLIST method. The parameters passed are the same as for the OBJECTLIST method.

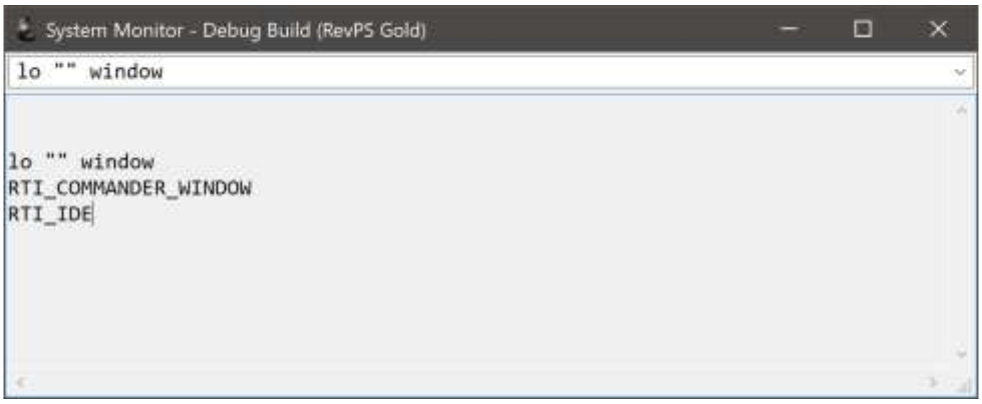
E.g. List all objects in the system:



```
lo "" ""
```

```
lo "" ""
CLIPBOARD
FILESYSTEM
RTI_COMMANDER_WINDOW
RTI_COMMANDER_WINDOW.$FC
RTI_COMMANDER_WINDOW.CMDWINDOW
RTI_COMMANDER_WINDOW.IMAGE
RTI_IDE
RTI_IDE.$FC
RTI_IDE.HSP_SOUTH
```

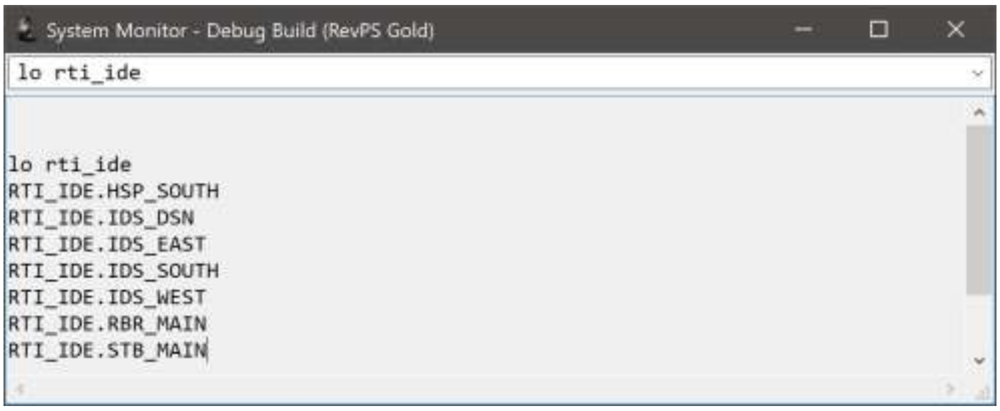
E.g. List all WINDOW types in the system:



```
lo "" window
```

```
lo "" window
RTI_COMMANDER_WINDOW
RTI_IDE
```

E.g list all objects that are children of a specified window:



```
lo rti_ide
```

```
lo rti_ide
RTI_IDE.HSP_SOUTH
RTI_IDE.IDS_DSN
RTI_IDE.IDS_EAST
RTI_IDE.IDS_SOUTH
RTI_IDE.IDS_WEST
RTI_IDE.RBR_MAIN
RTI_IDE.STB_MAIN
```

LOGTO

This command allows you to close the current application and open another one. The parameters that can be passed are the application ID, username, and password. If any of these arguments are incorrect or missing the system login dialog will be presented to allow for any corrections.

OFF

This command shuts down the Presentation Server, closing the application.

OL

Same as the LO command.

QUIT

Same as the OFF command.

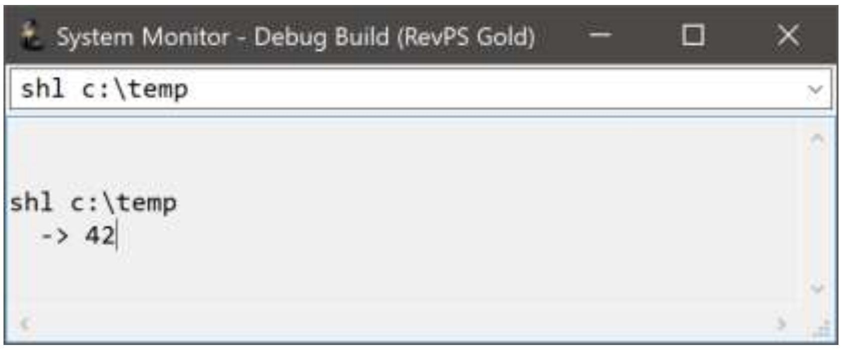
RXI

Same as the CFG command.

SHL

This command calls the Windows ShellExecute method to execute an external program or load a document. It can take two parameters: the name of program/document to load (required), and optionally any command line arguments that need to be passed.

E.g. Open the Windows Explorer in the c:\temp folder:



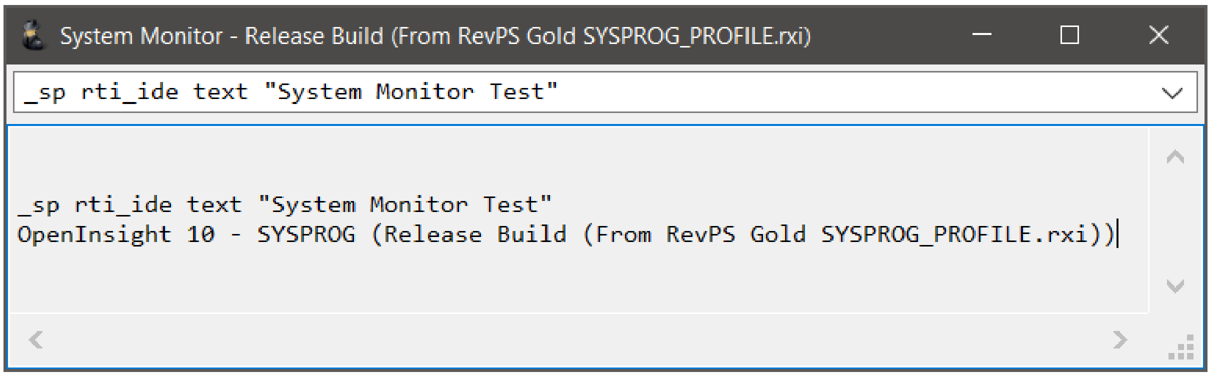
```
shl c:\temp
```

```
shl c:\temp
  -> 42
```

SP

This command executes the Set_Property procedure to set the value of an object property. The parameters passed are the same as for calling Set_Property in Basic+.

E.g. Set the TEXT property of the RTI_IDE window ensuring case is respected:


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
_sp rti_ide text "System Monitor Test"
```

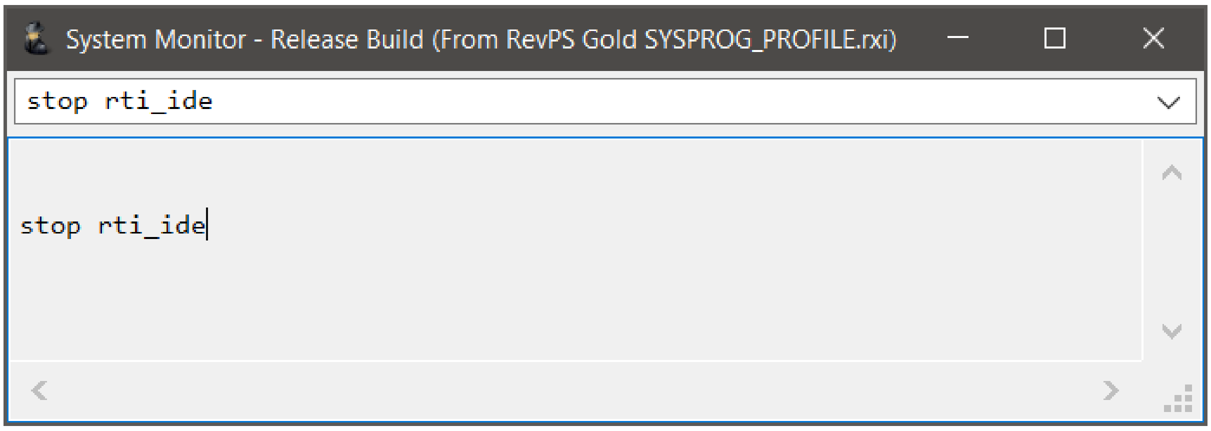
```
_sp rti_ide text "System Monitor Test"
OpenInsight 10 - SYSPROG (Release Build (From RevPS Gold SYSPROG_PROFILE.rxi))
```

STOP

This command closes a specified PS window, or all PS windows. Any window closed by the STOP command will *not* fire a CLOSE event, therefore doing this may leave "dangling" record locks in the case of data entry windows and so on, as no system cleanup code is executed.

To close a specific window simply pass the ID of the window as the first parameter to the STOP command.

E.g Terminate the RTI_IDE window:

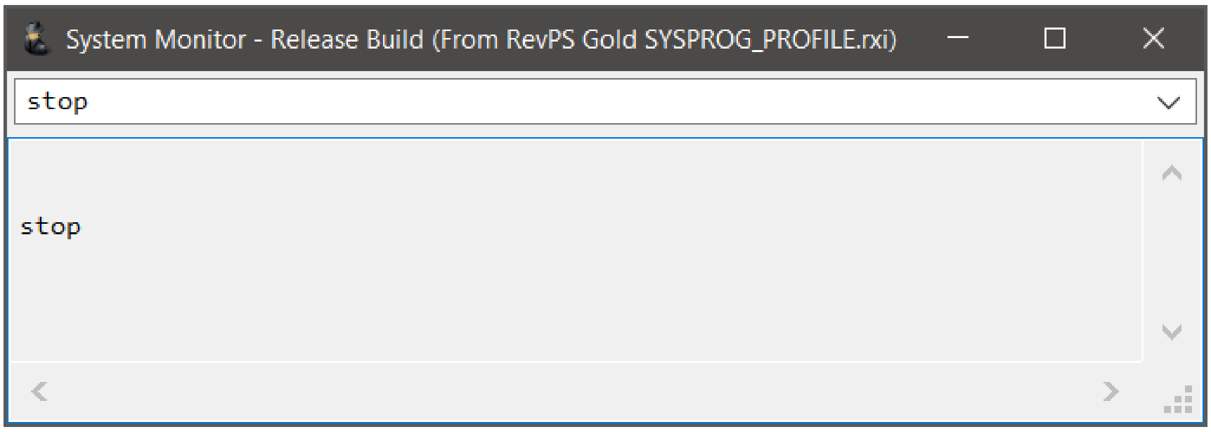

System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
stop rti_ide
```

```
stop rti_ide
```

To close all windows do not pass any parameters to STOP.

E.g. Terminate all PS windows:


System Monitor - Release Build (From RevPS Gold SYSPROG_PROFILE.rxi)

```
stop
```

```
stop
```