CHAPTER 12

Additional Programmer Utilities

This chapter describes utilities the programmer can use for a variety of purposes. All of these utilities are character-based.

Displaying the Current Namespace with %DIR

%DIR displays the name of the current namespace. If you invoke %DIR using the entry point INT^%DIR, the utility returns two variables:

- the current namespace name in the variable %DIR
- the default direct associated with the current namespace in %SYSDIR

This provides an easy way for the programmer to obtain data about the user's current namespace, as in this example:

```
%SYS>DO ^%DIR

You're in namespace %SYS
Default directory is c:\cachesys\mgr\
%SYS>DO INT^%DIR

%SYS>WRITE %DIR
%SYS
%SYS>WRITE %SYSDIR
c:\cachesys\mgr\
```

Adjusting the Priority of Jobs with %PRIO

The %PRIO utility lets you adjust the priority of a process and the way the process uses memory buffer space. Adjusting a process priority allows you to fine-tune your system to finish important jobs faster while keeping less important jobs from using all of your system resources.

There are three process priority modes:

Normal mode

Uses load balancing to divide the resources among the running processes. This is the normal operating mode.

High mode

Places a higher importance on the resources the high processes use, allocating more of the available memory and buffer space to the high processes. An example of a process you might want to run in high priority is a job scheduler so it can start jobs exactly at the scheduled time.

Low mode

Places a lower importance on the job allowing the system to allocate more resources to more important jobs. An example of a process you might want to run in low priority mode is a regularly scheduled monthly accounting of system resources.

Interactive jobs always begin at normal process priority without batch mode on. On Windows systems, jobs started with the JOB command also begin at normal process priority. On UNIX and OpenVMS systems, jobs started with the job command run in the background (in batch mode) and with a lower process priority.

%PRIO adjusts the job's priority at an operating system level. On UNIX platforms, a job must be run as supervisor to raise its priority.

Setting a background process to high priority causes it to compete directly with interactive jobs for the available CPU time. Since background jobs have no pauses for terminal I/O, a background job at high priority competes with great persistence. Give a background job high priority only when you want them to complete quickly at the expense of good response time for your interactive users.

When you set a background job to low priority, it starts running at a priority lower than interactive users, rather than running for awhile before the system determines it is a background job running in batch mode.

Calling %PRIO

The %PRIO utility has three entry points for priority mode.

Table 12-1: Internal Entry Points for %PRIO

Entry Point	**Definition**
LOW^%PRIO	Low priority with batch mode on
NORMAL^%PRIO	Normal priority with batch mode off
HIGH^%PRIO	High priority with batch mode off

Adjusting Buffer Allocation

The Configuration Manager allows you to specify what percentage of the total number of global buffers are available for background processes. This ensures that interactive processes will always have buffers to run in. The option to set how many global buffers are available for background jobs is "% Global Buffers for Batch" under Memory on the Advanced tab of the Configuration Manager.

Changing or Listing Namespaces with %CD

Use the %CD utility to change to a different namespace or to list all namespaces in your current configuration.

Note: If a specified new namespace has a remote default directory, the default directory of the current process is not changed while namespace is changed.

To change to a different namespace:

1. Issue the following command:
   USER> ***DO %CD***
2. Enter the name of the desired namespace at the following prompt:
   Namespace:

The following example illustrates the use of %CD to change from namespace ACCOUNT to %SYS.

```
ACCOUNT>DO ^%CD

Namespace: USER
You're in namespace USER
Default directory is c:\cachesys\mgr\
USER>
```

Alternatively, you can do either of the following to change to a different namespace:

- Issue the ZNSPACE command.
- Call the $ZU(5) function.

For example, to change to the namespace %USER, issue the following command:

```
%SYS> ZNSPACE "USER"
```

To display a list of all defined namespaces in your active configuration:

1. Issue the following command:
   %SYS> ***DO ^%CD***
2. Enter ? (a question mark) at the Namespace prompt:
   Namespace: ?

Viewing or Listing Namespace Translations with %NSP

Use the %NSP utility to get translations for namespaces.

Use %NSP with no entry point to get a detailed listing of all namespaces.

%NSP has two entry points,

- SHOW — Provides detailed namespace mapping information for the specified namespaces. If you don't specify a namespace, it will show mapping information about your current namespace.
- LIST — Displays a list of defined namespaces in your active configuration.

The following example shows the use of the LIST^%NSP entry point:

```
USER> DO LIST^%NSP

Here are the defined namespaces:
     %CACHELIB
     %SYS
     ACCOUNT
     SAMPLES
     USER

USER>
```

Examining the Global Directory with %GLO

The %GLO utility uses device 63 and the View Buffer to examine the global directory and to store a list of all the globals it finds there in the ^UTILITY global. It uses the following three formats:

```
^UTILITY("GLO",
    = journaling flag
        ^ protection mask
        ^ data growth area
        ^ first pointer block
        ^ collation type (numeric or string)
```

for all globals

```
^UTILITY("GLO",globalname)
    = journaling flag
        ^ protection mask
        ^ data growth area
        ^ first pointer block
        ^ collation type (numeric or string)
```

for the specified global

```
^UTILITY("GLO",0)
    = number of globals in this namespace
        ^ default directory
```

You will probably never directly call the %GLO utility from Programmer Mode. For the most part, it is a utility used by other utilities.

^%GLO has two additional entry points that are useful for building an extended global reference:

- **defdir**^%GLO returns the default directory of the current namespace.
- **defsys**^%GLO returns the host name of the system holding the default directory of the current namespace.

You can use them together to create an extended global reference:

```
^(^defdir^%GLO,^defsys^%GLO)^globalname(
```

The following is an example of the three formats:

```
%SYS>DO ^%GLO

%SYS>DO ^%G

Device: <RETURN>   Right margin: 80= <RETURN>

Global ^UTILITY("GLO",
^UTILITY("GLO",0)=14^g:\openm\mgr\
       "ACCOUNT")=0^195^7600^7211^1
         "ALPHA")=0^195^11600^11250^1
          "BETA")=0^195^0^16776961^1
        "ERTRAP")=0^195^7600^7207^1
         "GAMMA")=0^195^0^16776962^1
        "LAMBDA")=0^195^7600^7207^1
          "NAME")=0^195^7600^7209^1
       "NEWDATA")=0^195^7600^7218^1
        "REPORT")=0^195^7600^7214^1
       "ROUTINE")=0^195^7600^7204^1
         "SPOOL")=0^195^7600^7208^1
           "SYS")=0^215^267^3999^1
           "UPS")=0^195^7600^7206^1

Global ^UTILITY("GLO","SYS")
^UTILITY("GLO","SYS")=0^215^267^3999^1

Global ^UTILITY("GLO",0)
^UTILITY("GLO",0)=14^g:\openm\mgr\

Global ^

%SYS>
```

Selecting a Set of Globals for an Operation with %GSET

The %GSET utility is used by other utilities to select a set of globals for some operation. You will probably never call this utility from Programmer Mode.

The %GSET utility lets you specify a set of globals. The selected globals are stored in the ^UTILITY global in the following format ($J is the job number):

^UTILITY($J#256,*globalname*)=""

For information on methods you can use to specify globals for %GSET, see “Selecting Globals and Global Nodes” on page 10-2.

```
%SYS> DO ^%GSET

All Globals? No => No
Global ^SYS
1 item selected from
33 available globals
Global ^  <RETURN>
%SYS>d ^%G

Device: <RETURN>     Right margin: 80=> <RETURN>

Global ^UTILITY($J#256,"SYS")
^UTILITY(139,"SYS")=

Global ^
```

Select Routines for an Operation with %RSET

The %RSET utility selects a set of routines for some operation. %RSET does not differentiate between .MAC, .INT, .INC, and .OBJ routines.

The selected routine names are stored in the ^UTILITY global in the following format, where $J is the job number:

```
^UTILITY($J#256,0)=[number of routines selected];ROU
^UTILITY($J#256,[routine name])= ""
```

You can also use the tag KERNEL to store the routine names in $J:

```
KERNEL^%RSET
```

The KERNEL option prevents conflicts that may otherwise occur on systems with large job numbers.

If %RSET is called with the variable %JO defined, the value of %JO will be used for the first level subscript instead of the value $J#256. Otherwise, %JO will be set equal to $J and used as the first level subscript. %JO is also returned by this utility to the calling routine.

If %RSET is called with the variable %JO defined and ^UTILITY(%JO) is also defined, the %RSET utility will ask the following question:

```
Use old list of routines?
```

If you answer “Yes”, the utility will take no further action and will assume that the list under ^UTILITY(%JO) is the desired one. If you answer “No”, it will then ask the standard questions.

The %RSET utility looks in the ^ROUTINE global for existing routines. If you have routines without source code that you want %RSET to consider, you should call OBJECT^%RSET. The subsequent interaction is identical to that for the main entry point.

This example shows how to define a set routines:

```
USER> DO ^%RSET

All Routines? No => No

Routine: M-Q
Routine: <RETURN>
1 routine

USER>
```

Selecting Routines from within a Program with %RSETN

The %RSETN utility is used by the other Caché routine utilities to select routines. You will probably never call this utility directly from the Caché Programmer Mode prompt. You may, however, want to call %RSETN to select routines from within a program.

The ^mtemp Global

The selected routine names are stored in the **^mtemp** global in one of two formats. In both formats the first subscript, %msub, is the variable returned by %RSETN.

```
^mtemp(%msub,system@namespace,"extension",version,"name")=""
```

Besides %msub, ^mtemp accepts the following subscripts:

Table 12-2: Other Subscripts Accepted by ^mtemp

Subscript	Meaning
system@namespace	System and namespace where the routine resides.
extension	Extension of the routine (MAC, INC, INT, or OBJ).
version	Version number of the routine.
name	Name of the routine.

Parameters of %RSETN

%RSETN has the following format:

^%RSETN(*prompt*,*access*,*extensions*,*sort*)

The four parameters are described in the following table.

Table 12-3: %RSETN Parameters

Parameter	Meaning
prompt	Specifies the prompt that should be used to ask for routines. If this value is null, the default is the Routine(s): prompt.
access	Use “D” if selecting routines across namespaces is allowed; “S” if selecting routines across systems is allowed. If this value is null, access across namespaces and systems is not allowed.
extensions	A list of the routine extensions that are valid, separated by a comma. If this value is null, only routines with MAC and INT extensions are valid.
sort	Specifies the order in which routines should be sorted; use “DEVN” to specify that routines should be sorted according to Namespace/System, Extension, Version, and Name; use “DNEV” to specify that routines should be sorted according to Namespace/System, Name, Extension, and Version. If this value is null, routines are sorted according to the DEVN order.

Using Two-column Formats

If you are using a two-column format, (like the format used by the %RCOPY utility, which asks “Copy From” in one column and “Copy To” in another), the prompt and access parameters should have the delimiter $C(1) with the second piece.

This example calls %RSETN and specifies: "Routine Names:" for the prompt parameter; a null value for the access parameter so that access across namespaces and systems is prohibited; INT as the only acceptable extension; and the DEVN sort order.

```
%SYS>D ^%RSETN("Routine Names: ","INT","DEVN")

Routine Names: AAA.INT
Routine Names: BBB.INT
Routine Names: CCC.INT

%SYS>D ^%G

Global ^mtemp(%msub

^mtemp(163,"@","INT",1,"AAA")=
        "BBB")=
        "CCC")=
```