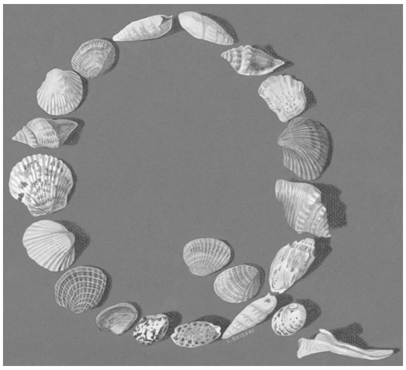


Chapter 2
Running Qshell

Use the "Start Qshell" CL command (STRQSH or QSH) to enter the Qshell environment. STRQSH has one parameter, CMD. The default value for CMD is *NONE, which means that no Qshell command is to be executed.

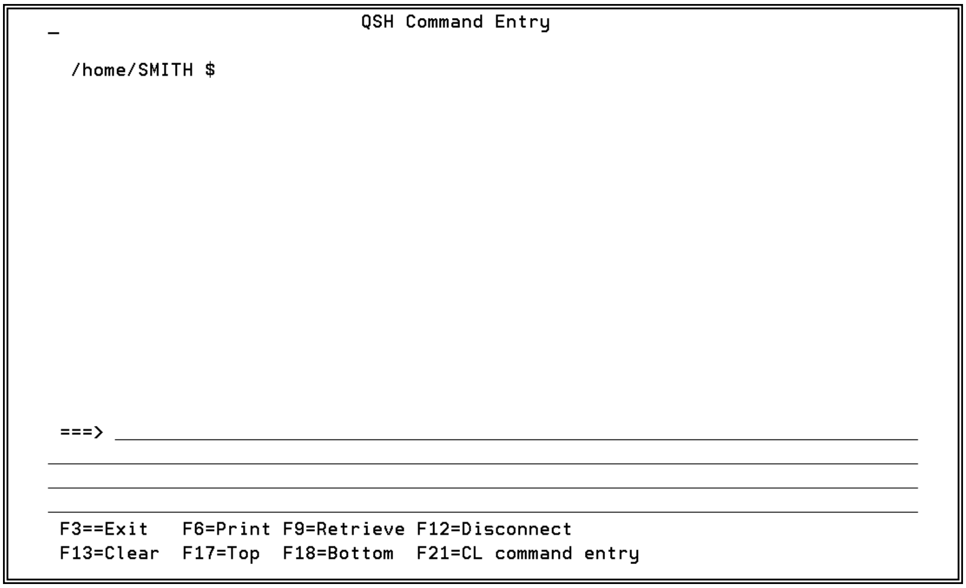
What happens when you run STRQSH depends on three things:

- Whether Qshell is running in a batch job or an interactive job
- Whether the CMD parameter specifies a Qshell command
- Whether Qshell is already active in an interactive job

The rest of this chapter examines these variations on STRQSH.

QSHELL IN AN INTERACTIVE JOB

If you start Qshell in an interactive job, and do not put a Qshell command in the CMD parameter, Qshell opens an interactive terminal session, as shown in Figure 2.1. If you use terminal services under Java, you will find this terminal session very familiar, since Qshell and Java use the same terminal support.



```
 _                             QSH Command Entry

    /home/SMITH $




















  ===> ____________________________________________________________________
 ___________________________________________________________________________
 ___________________________________________________________________________
 ___________________________________________________________________________
  F3==Exit   F6=Print F9=Retrieve F12=Disconnect
  F13=Clear  F17=Top  F18=Bottom  F21=CL command entry
```

Figure 2.1: Both Java and Qshell use the same interactive terminal support.

The Output Area

Below the panel title is the output area. When a Qshell session begins, the output area is blank except for the Qshell primary prompt string, which in Figure 2.1 is the value */home/SMITH $*.

The dollar sign is a prompt character. Qshell uses four prompt characters, shown in Table 2.1, to indicate that it needs input. In chapter 9, you will learn how to change the values of these prompt characters. For example, you will be able to use a percent sign (%) instead of a dollar sign for the PS1 prompt character.

In the output area, Qshell lists the commands you give it and the response to those commands. Figure 2.2 shows part of the output area of an interactive Qshell session. First, the *ls* command tells Qshell to display a list of files whose names end with a period and the characters *csv*. Qshell responds with a list of three file names and the Qshell prompt, which in this session is set to the

Table 2.1: Qshell Prompt Characters

Prompt Character	Description	Default Value	New in Release V5R2?
PS1	Primary prompt string. Qshell is ready for a command.	$	No
PS2	Secondary prompt string. Part of a command has been entered. Qshell is waiting for the remainder of the command.	>	No
PS3	Select command prompt. Qshell is waiting for the user to choose an option presented with the select compound command.	#?	Yes
PS4	Debug prompt string. Qshell is displaying data in debug mode.	+	No

current directory followed by a dollar sign. The prompt informs the user that Qshell has completed the request and is ready for work.

```
> ls *.csv
  cust.csv        two.csv         uuu.csv
  /home/SMITH $
> rm -f *.csv
  /home/SMITH $
> ls *.txt
  cvthex.txt      filestar.txt    output.txt      readdata4.txt
  demo.txt        ftpin.txt       parm2.txt       serr.txt
  dirlist.txt     ftpmodel.txt    readdata.txt    sout.txt
  errorlog.txt    grepout.txt     readdata2.txt   tabdelimf.txt
  filedot.txt     mylsout.txt     readdata3.txt
  /home/SMITH $
```

Figure 2.2: The Qshell output area holds commands and responses.

The next command in Figure 2.2 is the *rm*, which removes directory entries. In this example, it deletes all CSV (comma-separated value) files.

The second *ls* command in Figure 2.2 lists files that end with *.txt*. Qshell responds with the appropriate list of file names and another Qshell prompt.

The Command Line

Below the output area is the command line, where you type the commands you want Qshell to carry out. Do not confuse this command line with a CL command line. You cannot directly execute CL commands from the Qshell command line. However, Qshell has a system utility you can use to execute CL commands.

The Function-Key Legend

Below the command line is the function-key legend. Table 2.2 describes the function keys that are active in the shell.

Table 2.2: Qshell Function-Key Assignments

Key	Name	Description
F3	Exit	Close the interactive session and end Qshell. Contrast with F12.
F5	Refresh	Refresh the output area.
F6	Print	Copy the contents of the output area to a spooled file.
F7	Page up (roll down)	Page backward through the output area. If a number is on the command line, page back that many lines; otherwise, page back one full screen.
F8	Page down (roll up)	Page forward through the output area. If a number is on the command line, page forward that many lines; otherwise, page forward one full screen.
F9	Retrieve a previous command.	If the cursor is on the command line, retrieve previously executed commands in reverse order. If the cursor is in the output area, retrieve the command on which the cursor rests.

Table 2.2: Qshell Function-Key Assignments, *continued*

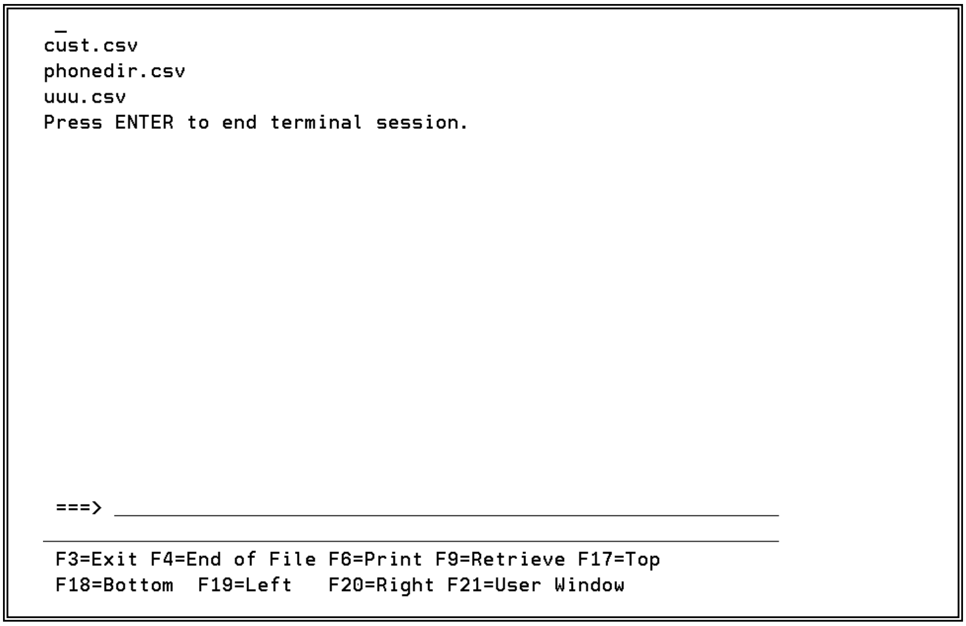
Key	Name	Description
F11	Toggle line wrap	Determine whether long lines in the output area are wrapped or truncated.
F12	Disconnect	Close the interactive session, but do not end Qshell. You may resume the Qshell session with the STRQSH command. Contrast with F3.
F13	Clear	Clear the output area. This is more than a clear-screen function. It also clears the session history, so you will not be able to page back through the output area to previously executed commands.
F14	Adjust command line length	The command line is normally four lines long. Pressing F14 with a number adjusts the command area to that number of lines. If no number is on the command line, pressing F14 adjusts the command line to four lines.
F17	Top	Position to beginning of output area.
F18	Bottom	Position to end of output area.
F19	Left	Move the output area window to the left. This is used when the output area is truncated because line wrap is off. (See the description of the F11 key.)
F20	Right	Move the output area window to the right, also used when the output area is truncated.
F21	Command entry	Open a window with an OS/400 command line.

The CMD Parameter

You may specify a Qshell command up to 5,000 characters long in the CMD parameter of the STRQSH command. Here is an example:

```
STRQSH CMD('ls *.csv')
```

In this case, Qshell does not open a terminal session. Instead, it executes the *ls* command and ends.



```
 _
cust.csv
phonedir.csv
uuu.csv
Press ENTER to end terminal session.




















 ===> ________________________________________________________
______________________________________________________________
 F3=Exit F4=End of File F6=Print F9=Retrieve F17=Top
 F18=Bottom  F19=Left   F20=Right F21=User Window
```

Figure 2.3: Qshell opens a temporary terminal session, if necessary.

If the command specified in the CMD parameter produces output, Qshell opens a temporary C runtime terminal session, as shown in Figure 2.3. You can also direct the output elsewhere, as discussed later in this chapter.

Input, however, is a different story. If you are running a script in this manner and Qshell encounters a request for terminal input, it ignores the request.

QSHELL IN BATCH JOBS

You may also run Qshell in a batch job, within certain limits:

- Input requests are ignored.
- If the command produces output, Qshell spools the output.

The following example shows how to submit a Qshell command to run in batch mode:

```
SBMJOB CMD(QSH CMD('rm -f *.csv')) JOB(REMOVEFILE)
```

If the CMD parameter contains the value *NONE, Qshell does nothing and the job ends normally.

REDIRECTING QSHELL OUTPUT

As of V5R2, you may use the QIBM_QSH_CMD_OUTPUT environment variable to control the destination of Qshell output. The allowable values are listed in Table 2.3.

Table 2.3: Allowable Values of Environment Variable QIBM_QSH_CMD_OUTPUT

Value	Description
NONE	The output is discarded.
STDOUT (default)	The output is directed to a C runtime terminal session.
FILE=*name*	The output is directed to the file in *name*. If the file already exists, the output overwrites the previous contents.
FILEAPPEND=*name*	The output is appended to the file in *name*.

In the following example of the QIBM_QSH_CMD_OUTPUT environment variable, the names of comma-delimited files are written to the file lsout.txt in the current directory, replacing any previous contents:

```
ADDENVVAR ENVVAR(QIBM_QSH_CMD_OUTPUT) VALUE('FILE=lsout.txt')
STRQSH CMD('ls *.csv')
```

As another example, the names of files that begin with a lowercase *b* are appended to the end of stream file lsout.txt, which is in the current directory:

```
ADDENVVAR ENVVAR(QIBM_QSH_CMD_OUTPUT)
VALUE('FILEAPPEND=lsout.txt') STRQSH CMD('ls b*')
```

In both of these examples, output is sent to IFS files in the current directory. Writing to a stream file is generally the sensible approach. However, that is not to say that Qshell won't write to a database file. Here, the names of comma-delimited files are written to program-described database file, lsout, in the FILEINFO library:

```
ADDENVVAR ENVVAR(QIBM_QSH_CMD_OUTPUT) +
   VALUE('FILE=/qsys.lib/fileinfo.lib/lsout.file/lsout.mbr')
CRTPF FILE(FILEINFO/LSOUT) RCDLEN(24)
STRQSH CMD('ls *.csv')
```

The output data is written to FILEINFO/lsout, ignoring record length. Each line of output is terminated by a hexadecimal-25 character.

To get a better idea of how output is written to database files, see Figure 2.4, which displays the Qshell output in character format, and Figure 2.5, which displays the output in hexadecimal format.

Processes

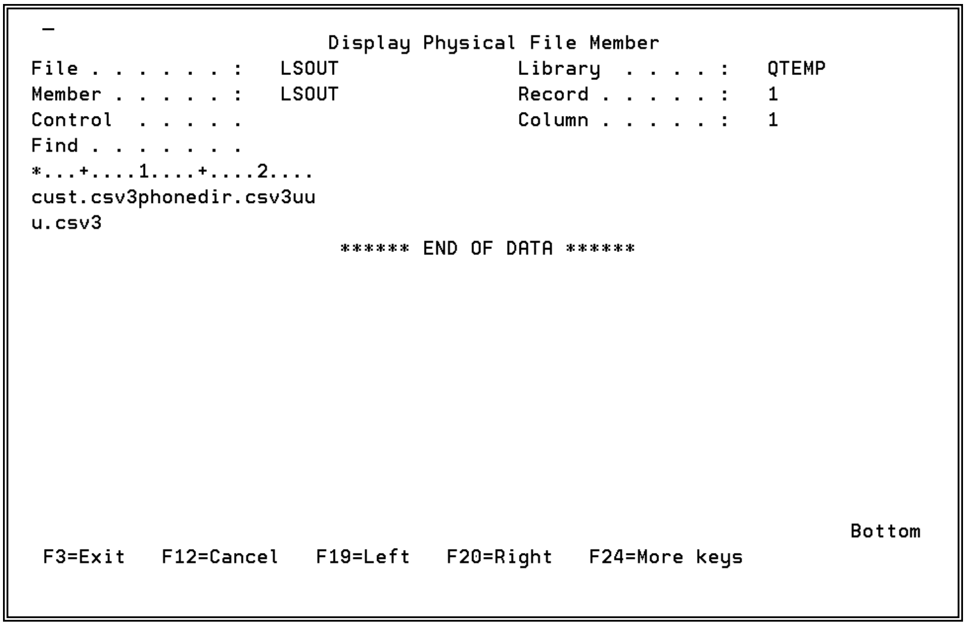
On Unix systems, a process is a running program. The Qshell interpreter is a process. Each utility that is not a built-in command runs in its own process. Scripts run in their own processes, unless they are invoked with the source (dot) utility.

Under Qshell, every process runs in its own job, so the terms *process* and *job* are interchangeable.

The Terminal Session

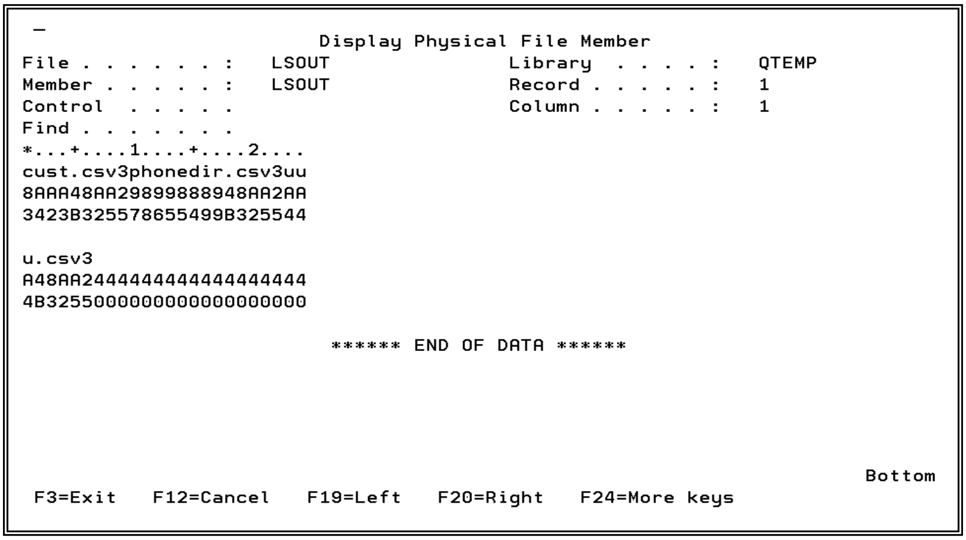
The terminal session appears to be an interactive session, but that is not entirely true. A terminal session is actually a combination of jobs.

When a user starts an interactive Qshell session, OS/400 starts a batch-immediate job running the Qshell interpreter, a program named QZSHSH. At this point, the user is running two jobs: the interactive job and the batch-immediate job.



```
 _
                         Display Physical File Member
File . . . . . . :   LSOUT               Library  . . . . :   QTEMP
Member . . . . . :   LSOUT               Record . . . . . :   1
Control  . . . . .                       Column . . . . . :   1
Find . . . . . . .
*...+....1....+....2....
cust.csv3phonedir.csv3uu
u.csv3
                          ****** END OF DATA ******

                                                                       Bottom
 F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

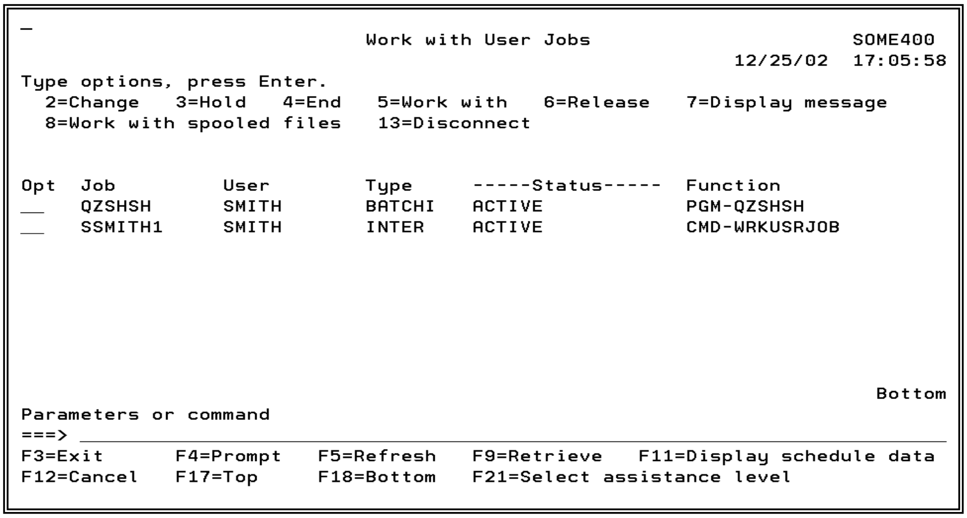
Figure 2.4: Qshell ignores the record length of a database file.



```
 _
                         Display Physical File Member
File . . . . . . :   LSOUT               Library  . . . . :   QTEMP
Member . . . . . :   LSOUT               Record . . . . . :   1
Control  . . . . .                       Column . . . . . :   1
Find . . . . . . .
*...+....1....+....2....
cust.csv3phonedir.csv3uu
8AAA48AA29899888948AA2AA
3423B325578655499B325544

u.csv3
A48AA2444444444444444444
4B3255000000000000000000

                          ****** END OF DATA ******

                                                                       Bottom
 F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

Figure 2.5: Qshell terminates each line of output with a hexadecimal-25 character.

You can use the Work with Active Jobs (WRKACTJOB) and Work with User Jobs CL command (WRKUSRJOBS) to see which jobs are running. The information returned by the WRKUSRJOB command is shown in Figure 2.6.



```
_
                             Work with User Jobs                      SOME400
                                                            12/25/02  17:05:58
Type options, press Enter.
  2=Change   3=Hold   4=End   5=Work with   6=Release   7=Display message
  8=Work with spooled files   13=Disconnect


Opt  Job         User        Type     -----Status-----  Function
__   QZSHSH      SMITH       BATCHI   ACTIVE            PGM-QZSHSH
__   SSMITH1     SMITH       INTER    ACTIVE            CMD-WRKUSRJOB








                                                                        Bottom
Parameters or command
===> ______________________________________________________________________
F3=Exit      F4=Prompt   F5=Refresh   F9=Retrieve   F11=Display schedule data
F12=Cancel   F17=Top     F18=Bottom   F21=Select assistance level
```

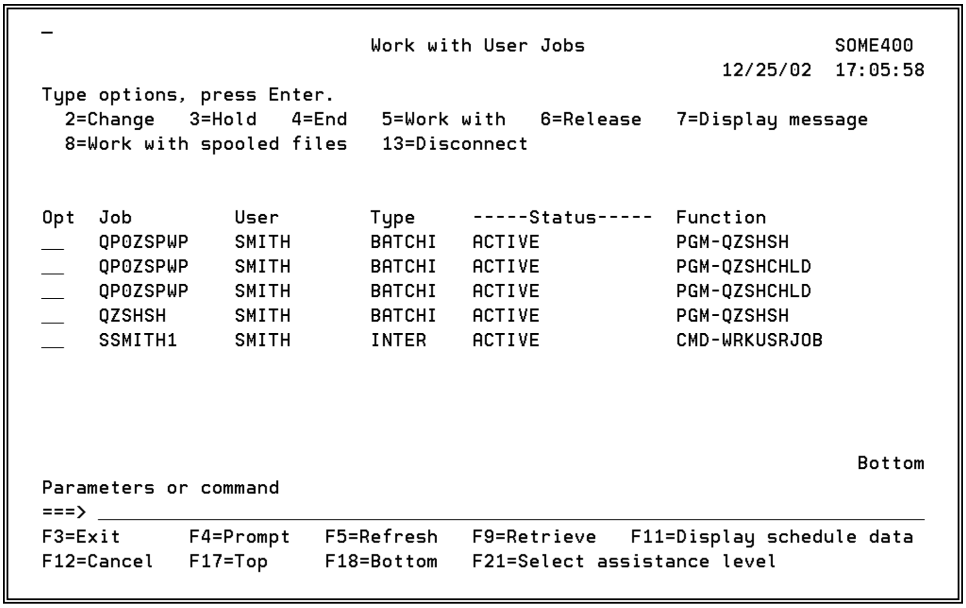
Figure 2.6: The Qshell interpreter runs as a batch immediate job.

If you start a subshell or run a utility, Qshell will start additional jobs as needed. These have the name QP0ZSPWP, as shown in Figure 2.7.

Under V5R2, you can use Qshell's *ps* (Display Process Status) command to display information about processes. This command is illustrated in Figure 2.8.

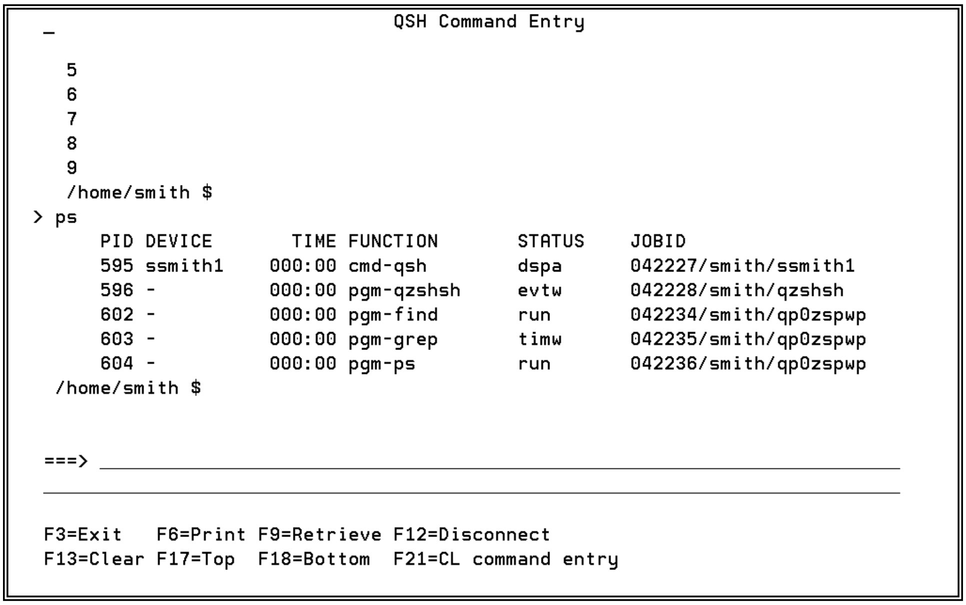
Prestart Jobs

You can speed up processing by using a prestart job, which is a job that begins running when a subsystem is started. When Qshell starts a new process, it will use a prestart job if one is available. This improves performance, because the system does not have to start a new job.



```
_
                           Work with User Jobs                      SOME400
                                                          12/25/02  17:05:58
Type options, press Enter.
  2=Change   3=Hold   4=End   5=Work with   6=Release   7=Display message
  8=Work with spooled files   13=Disconnect


Opt  Job         User        Type     -----Status-----  Function
__   QP0ZSPWP    SMITH       BATCHI   ACTIVE            PGM-QZSHSH
__   QP0ZSPWP    SMITH       BATCHI   ACTIVE            PGM-QZSHCHLD
__   QP0ZSPWP    SMITH       BATCHI   ACTIVE            PGM-QZSHCHLD
__   QZSHSH      SMITH       BATCHI   ACTIVE            PGM-QZSHSH
__   SSMITH1     SMITH       INTER    ACTIVE            CMD-WRKUSRJOB





                                                                       Bottom
Parameters or command
===> _________________________________________________________________________
F3=Exit      F4=Prompt   F5=Refresh   F9=Retrieve   F11=Display schedule data
F12=Cancel   F17=Top     F18=Bottom   F21=Select assistance level
```

Figure 2.7: The Qshell interpreter runs as a batch-immediate job.



```
                            QSH Command Entry
_
  5
  6
  7
  8
  9
  /home/smith $
> ps
     PID DEVICE       TIME FUNCTION       STATUS    JOBID
     595 ssmith1    000:00 cmd-qsh        dspa      042227/smith/ssmith1
     596 -          000:00 pgm-qzshsh     evtw      042228/smith/qzshsh
     602 -          000:00 pgm-find       run       042234/smith/qp0zspwp
     603 -          000:00 pgm-grep       timw      042235/smith/qp0zspwp
     604 -          000:00 pgm-ps         run       042236/smith/qp0zspwp
 /home/smith $


===> ______________________________________________________________________
___________________________________________________________________________

F3=Exit   F6=Print F9=Retrieve F12=Disconnect
F13=Clear F17=Top  F18=Bottom  F21=CL command entry
```

Figure 2.8: The ps *command displays information about Qshell processes.*

Use the Add Prestart Job Entry (ADDPJE) command to create a prestart job. For example, the following command adds a prestart job to the QINTER subsystem description:

```
ADDPJE SBSD(QSYS/QINTER) PGM(QSYS/QP0ZSPWP)
   INLJOBS(10) THRESHOLD(5) ADLJOBS(10)
   JOBD(QGPL/QDFTJOBD) MAXUSE(1) CLS(QGPL/QINTER)
```

To make Qshell use a prestart job, place a value of *Y* in the environment variable QSH_USE_PRESTART_JOBS. Use the *export* command so that child processes will also use prestart jobs:

```
export -s QSH_USE_PRESTART_JOBS=Y
```

A good place to put this command is in the /etc/profile special script file, which is discussed in chapter 4.

SUMMARY

The Qshell interpreter can run in both interactive and batch environments. In an interactive job, the user can enter commands for immediate execution.

Qshell executes commands in processes, which are implemented as jobs. You can improve the performance of Qshell commands by providing prestart jobs.