

Terminal Paging

This chapter covers the following topics:

- Overview
 - POPEN Function
 - PWRT Function
 - PREAD Function
 - PLIMIT Function
-

Overview

The Com-plete terminal paging functions enable an application program to create a temporary disk file data set (referred to as a page file) and read or write data to the page file in blocks of data (referred to as pages). The pages of data in the page file can be dynamically displayed at the terminal from which the application program is executing without terminating the application program. The dynamic display of pages is accomplished by use of the Com-plete online utility program UP.

The page file is allocated as a temporary SD file data set, and is available for use only until another program is initiated from the same terminal. Each terminal can have only one page file per COM-PASS stack level assigned to it at any time, and the page file assigned to one terminal cannot be accessed by an application program executing from another terminal. Each page file can contain a maximum of 255 pages.

A terminal user must use the Com-plete paging utility UP to display the contents of a page file. The use of the UP utility program is described in the *IBM User's Guide for the Graphical Data Display Manager*. The UP functions used to display pages written to a page file are independent of the application program. The application program is not active while you use the UP paging utility to display data because the application program and the UP utility execute as separate subtasks of the operating system.

After using the UP utility functions to display the contents of a page file at the terminal, you can choose to change the contents of the page being displayed. Since the paging functions do not perform terminal I/O, the application program has to issue a terminal I/O READ function to obtain the information being displayed. The data is then written to the page file with a terminal paging WRITE function in order to force the change to the paging file.

The available paging functions are summarized in the following table. Each of these functions is described separately in this chapter.

| Function | Description |
|----------|--|
| POPEN | Allocates the SD file space required for the page file in use. |
| PWRT | Writes data to the page file. |
| PREAD | Reads data from the page file and places it in a working storage area in the application program. |
| PLIMIT | Restricts, or limits, the pages that can be displayed at a given terminal, based upon application-selected criteria. |

POPEN Function

The POPEN function is used to allocate, or create, a page file. The page file created is assigned to the terminal from which the application program issuing the POPEN function is called. Since the POPEN function reserves an area for the page file, it must be the first paging function issued in the application program.

The POPEN function arguments enable the application program to determine the size of the page file. Page file size is determined by two items:

- The number of pages to be reserved for the page file
- The size of each page in the page file

Each item must be specified at the time the POPEN function is issued.

Format

The format for using the POPEN function is:

POPEN (retcode,length,numrec)

| | |
|---------|--|
| retcode | Required. A fullword where Complete places the return code upon completion of the operation. |
| length | Required. A binary halfword containing the length of each page expressed as a number of characters. |
| numrec | Required. A binary halfword containing the total number of pages in the file. numrec can have a maximum value of 255. |

Return Codes

A return code of 0 is issued upon normal completion of the POPEN function.

Abends

An abnormal termination may occur during execution of the POPEN function. Possible causes include:

- An invalid *length* argument was specified;
- An invalid *numrec* argument was specified;
- Too large a page file was requested;
- A disk error occurred during open processing;
- Not enough space is available in the SD allocation pool to create the requested page file;
- An incorrect number of arguments was specified.

PWRT Function

The PWRT function is used to write data to a page file that has been created with the POPEN function. The data to be written to the page file resides in a working storage area of the application program and is written on a page basis. The amount of data to be written cannot exceed the size of one page, but can be smaller than one page. Pages can be written either sequentially or directly.

Several PWRT options are available to assist in the formatting of pages. These format options include:

- Erase the screen for a display terminal prior to displaying the page contents.
- Insert a new-line symbol (X'15') into the page data to be displayed in order to format individual lines of output.
- Treat the data in the page to be displayed as text, displaying the data in such a manner that words will not be split between lines.
- Insert terminal-dependent control characters into the page data in order to format the page when displayed. In this situation, the application program should be executed from only one device type, since the control characters for one device type terminal may produce unpredictable results when used with another kind of device type terminal.
- Do not automatically erase a display terminal screen when the page is displayed. This option might be used when a display type terminal is to be used and the application program is to create a page that, when displayed, will contain an explanation of a series of variables. The first page might be a list of all variables, and successive pages might then be used to explain each variable. Since the screen is not erased, the successive pages can literally add information to the screen.

Note that the size of a page in a page file can be less than or larger than the size of an output display for a specific terminal device type. If the terminal to be used is a display type terminal, then multiple pages can be displayed at one time.

PWRT causes a page to be written to the page file. If the terminal in use is a display type terminal, the screen is erased prior to a display operation when the page is requested for display.

PWRTT causes a page to be written to the page file. The data is treated as text and, when displayed at the terminal, words are not split between two lines. If the terminal in use is a display type terminal, the screen is erased prior to a display operation when the page is requested for display.

PWRTS causes a page to be written to the page file. The data to be written to the page file must contain all terminal device-dependent control characters. These characters are used to format the terminal display when the terminal operator requests a display of the page. If the terminal in use is a display type terminal, the screen is not erased prior to a display operation when the page is requested for display.

PWRTSE causes a page to be written to the page file. The data to be written to the page file must contain all terminal device-dependent control characters. These characters are used to format the terminal display when the terminal operator requests a display of the page. If the terminal in use is a display type terminal, the screen will be erased prior to a display operation when the page is requested for display.

Format

The format for using the PWRT function is:

```
PWRT[T|S|SE] (retcode,area[,numrec][,length])
```

| | |
|---------|--|
| retcode | Required. A fullword where Complete places the return code upon completion of the operation. |
| area | Required. The buffer in the application program's working storage area from which the data is to be written. |
| numrec | Optional. A binary halfword that contains the number of the page to be written. Default: If not specified or zero, the page number is assumed to be one higher than the previously-written page. If a previous page has not been written, the page number is assumed to be first (page number 1). |
| length | Optional. A binary halfword that contains the length in bytes of the page to be written. The length must not be larger than the length of the page specified in the POPEN statement for the page file. Default: The page length specified in the POPEN statement for the page file. |

Return Codes

A return code of 0 is issued upon normal completion of the PWRT function.

Abends

An abnormal termination may occur during execution of the PWRT function. Possible causes include:

- An invalid argument was specified;
- A function other than POPEN was specified, and POPEN was never specified;

- An incorrect number of arguments was specified;
- A disk error occurred while processing the page file.

PREAD Function

The PREAD function is used to read data from a page file. Data being read is placed in a working storage buffer area within the application program and is read on a page basis. The amount of data to be read cannot exceed a page in size, but can be less than a page. In addition, pages can be read either sequentially or at random.

After a page of data is read from the page file by the PREAD function, the application program can display the data to the terminal by use of the Com-plete terminal I/O functions. If you want to modify the contents of a displayed page, the application program must read the modified page from the terminal with a terminal I/O READ function and then write the page to the page file using the PWRT function.

Format

The format for using the PREAD function is:

`PREAD (retcode,area [,numrec][,length])`

| | |
|---------|---|
| retcode | Required. A fullword where Com-plete places the return code upon completion of the operation. |
| area | Required. The buffer area in the working storage area of the application program where the page data is to be placed. |
| numrec | Optional. A binary halfword containing the number of the page to be read.Default: Pages are read sequentially. If numrec is zero, the page to be read is assumed to be one higher than the previous page read. If a previous page has not been read, the first page is read. |
| length | Optional. A binary halfword containing the amount of data to be read from the page indicated. length must not be larger than the length specified in the POPEN function for the page file.Default: The page length specified with the POPEN function for the page file. |

Return Codes

The following return codes are issued by the PREAD function.

| | |
|---|--|
| 0 | Normal completion. |
| 4 | The page requested has not been written. |

Abends

An abnormal termination may occur during execution of the PREAD function. Possible causes include:

- An invalid argument was specified;
- The page file was not opened prior to issuing the PREAD function;
- An incorrect number of arguments was specified;
- A disk I/O error occurred while reading the page file.

PLIMIT Function

You normally display the pages of a page file using the Com-plete online utility program UP. This utility program is fully described in the Com-plete Utilities documentation. If you display pages from the page file, Com-plete keeps track of the following items:

- The number of the current page being displayed;
- The number of the highest page written to the page file by the application program.

For example, an application program has opened a page file with twenty pages using the POPEN function, but has only written ten pages to the page file using the PWRT function. Before any pages are displayed at the terminal, the current page and the highest page values within Com-plete will contain 1 and 10, respectively.

If you enter the command:

to display page 3, the value for the current page is changed to 3, and the value for the highest page remains at 10.

If you enter the command:

to display the next page, Com-plete determines the last page number displayed (in this case, page 3), increments the page count by one, and displays the next page (in this case, page 4). In this example, the current page value then becomes 4, and the last page value remains at 10.

If you enter the command:

to display the previous page, Com-plete determines the last page displayed from the current page value (in this case, page 4), decrements the page count by one, and displays the page (in this case, page 3). In this example, the current page value then becomes 3, and the last page value remains at 10.

If you enter the command:

to display the highest page written by the application program, Com-plete determines the value of the highest page (in this case, page 10), increments the page count accordingly, and displays the highest page written. In this example, the current page value then becomes 10, and the last page value remains at 10.

If you enter the command:

after having displayed the highest page written by the application program, Com-plete displays an error message at the terminal, indicating that the application program did not create any more pages.

The PLIMIT function enables the application program to dynamically interrogate the page number of the current page and to change the value for both the current page and the highest page. This feature is accomplished by using the PLIMIT function to pass data back and forth to Com-plete using a working storage area in the application program.

The page number of the last page actually displayed on the terminal screen is optionally placed into the working storage area of the application program by using the PLIMIT function. In addition, the current page and the highest the page available for display is optionally set by the application program by use of the PLIMIT function. If the PLIMIT function is not used, Com-plete sets the value of the current page to 1 and the value of the highest page to the number of the highest page written to the page file by the program.

The PLIMIT function is useful in a variety of circumstances. For example, if you want to change the information contained in a particular page, the application program must use a Com-plete terminal I/O function to read your entry. The PLIMIT function can then be used to determine the appropriate page number of the page being modified before the PWRT function is used to actually write the change to the page file.

The PLIMIT function is also useful when there are several pages in a page file, not all of which are meaningful to the terminal operator. For example, if a paging file has 20 pages arranged as follows:

Pages 1 through 4 - instructions
 Pages 5 through 13 - data
 Pages 14 through 20 - programming notes

the PLIMIT function can be used to specify:

Current page = 5
 Highest page = 13

If you enter the command:

pages after page 13 are not displayed, allowing you to avoid viewing the information on pages 14 through 20. Since the initialized current page value is 5, a note could be written on the first logical page of data (page 5) indicating that instructions are contained on pages 1 through 4. You can then view pages 1 through 4 if desired (to read the instructions) by using a command such as:

Format

The format for using the PLIMIT function is:

```
PLIMIT (retcode,pagehi[,pagecur][,getpcur])
```

| | |
|---------|---|
| retcode | Required. A fullword where Com-plete places the return code upon completion of the operation. |
| pagehi | Required. A binary halfword containing the number of the highest page to be available for display. if pagehi is zero, the highest page actually written to the page file will be used. |
| pagecur | Optional. A binary halfword containing the number of the page to be displayed in response to the command *UP/C.Default: 1 |
| getpcur | Optional. A binary halfword into which Com-plete places the number of the last page displayed by the terminal operator.Default: None. |

Return Codes

A return code of 0 is issued upon normal completion of the PLIMIT function.

Abends

An abnormal termination may occur during execution of the PLIMIT function. Possible causes include:

- The page number requested is invalid;
- The page file was not open at the time the PLIMIT function was issued.