

Select.Seek Subroutine

Description

Select.Seek is used to change the current position in a select list.

Syntax

Select.Seek(*mode*, *cursor*, *position*, *flag*)

Using Select.Seek

Use Select.Seek to move either absolutely or relatively within a select list. You can go directly to the top, bottom, or specific place in a select list, or you can go to a position that is relative to one of those absolute locations. For example, you can go to the top of to 5 positions on either side of the top.

The select list may be obtained either through Basic+ (Select or Select...By) or by the RList subroutine. The select list may be either latent or resolved.

When using Select.Seek, make note of the following requirements:

- You may execute either a standard or an extended Select process, but you must use the Readnext By command, with a "non-terminating" option. The selections may be either ascending or descending.
- If you are using a standard Select process, ou must still use the extended Readnext By command to read record keys. In that case, use 0 as the cursor number.

You may specify relative positions by using *position*. For example, the code:

```
Select.Seek(current_seek$, 0, 5, flag)
```

will take you five positions past your current position. Similarly,

```
Select.Seek(current_seek$, 0, 5, flag)
```

will take you five positions back from the current position. Also,

```
Select.Seek(bottom_seek$, 0, -5, flag)
```

will take you to the fifth position before the end of the Select list.

You may also mark a particular position, do some processing that reposition the select list, then return to the marked position.

An error code is returned in *flag*.

Parameters

The Select.Seek subroutine has the following parameters:

Parameter	Description
-----------	-------------

<i>mode</i>	<p><i>Mode</i> determines either the position in the select list (top, bottom, current) from which to calculate the new position (top+0, top+5, current-1), or it sets a marker so that you can later restore the current position.</p> <p>There are five possible arguments for <i>mode</i>:</p> <table> <tr> <th>Argument</th><th>Meaning</th></tr> <tr> <td>0</td><td>Begin from the top</td></tr> <tr> <td>1</td><td>Begin from the bottom</td></tr> <tr> <td>2</td><td>Begin from the current position</td></tr> <tr> <td>3</td><td>Mark the current position</td></tr> <tr> <td>4</td><td>Return to the previously marked position</td></tr> </table> <p>This argument must be supplied, and it must be numeric.</p>	Argument	Meaning	0	Begin from the top	1	Begin from the bottom	2	Begin from the current position	3	Mark the current position	4	Return to the previously marked position
Argument	Meaning												
0	Begin from the top												
1	Begin from the bottom												
2	Begin from the current position												
3	Mark the current position												
4	Return to the previously marked position												
<i>cursor</i>	<p>An argument for the <i>cursor</i> parameter must be a value from 0 through 8. This argument must be supplied, and it must be numeric.</p> <ul style="list-style-type: none"> If you selected record keys using an Basic+ or RList Select statement, <i>cursor</i> must be 0. 												
<i>position</i>	<p><i>Position</i> has two functions, depending on <i>mode</i>:</p> <table> <tr> <th>When mode is</th><th>Meaning of <i>position</i></th></tr> <tr> <td>0</td><td>Offset value</td></tr> <tr> <td>1</td><td>Offset value</td></tr> <tr> <td>2</td><td>Offset value</td></tr> <tr> <td>3</td><td>Return variable</td></tr> <tr> <td>4</td><td>Position data</td></tr> </table> <p>When the value of <i>position</i> is an offset value, it specifies how far ahead (a positive value) or behind (a negative value) the default value a location is to be. For example, if the top_seek argument is specified (mode 0), an argument for <i>position</i> will position the select list either ahead or behind the top position an amount that is the value of <i>position</i>. To go to an absolute top, bottom, or current position, make <i>position</i> null or zero.</p> <p>When marking or restoring the select list position, <i>position</i> contains the positioning data needed by Select.Seek to restore the select position. Therefore, <i>position</i> returns a value when executing the mark_seek command (mode 3), then supplies that value for the restore_seek process (mode 4).</p>	When mode is	Meaning of <i>position</i>	0	Offset value	1	Offset value	2	Offset value	3	Return variable	4	Position data
When mode is	Meaning of <i>position</i>												
0	Offset value												
1	Offset value												
2	Offset value												
3	Return variable												
4	Position data												
<i>flag</i>	<p>Pass an argument for <i>flag</i> to return true or false, depending on the success of the operation.</p>												

Values Returned

Flag returns true if the process succeeded, false if it did not.

Position returns data specifying the current position after a **mark_seek** operation.

Example

```
/* The following code opens the SAMPLE_CUSTOMERS file, runs Select on it, then exercises various Select.Seek functions. */
```

```
Declare subroutine Select.Seek, FsMsg, Msg
```

```
Equ top_seek$      to 0
Equ bottom_seek$   to 1
Equ current_seek$  to 2
Equ mark_seek$     to 3
Equ restore_seek$  to 4
```

```
filename = "SAMPLE_CUSTOMERS"
```

```

fieldname = "COMPANY_NAME"
cursor    = 0
position  = ""

ClearSelect    ; // Just in case

Open filename to filevar else FsMsg(); Stop
Select filevar

// Move to the first record key in the select list
mode = top_seek$
place = "top"
GoSub go_seek    ; // set up arguments for a top_seek

// move to the last record key in the select list
mode = bottom_seek$
place = "bottom"
GoSub go_seek

// mark the current position in the select list
mode = mark_seek$
place = "bottom"
GoSub go_seek

// move to fifth record key from the bottom of the select list
mode = bottom_seek$
position = -5
place = position : " from bottom"
GoSub go_seek

// return to the position marked above
mode = restore_seek$
position = store_position ; // restore the marked position
Select.Seek(mode, cursor, position, flag)
If flag else Stop
Msg(@Window, "Here is restored ID: " : @ID)

Stop ; // end of program

*****
* Internal Subroutines
*****

go_seek:
    Select.Seek(mode, cursor, position, flag)
    If flag then
        Readnext @ID using cursor By AN else
            FsMsg()
            Stop
        end
        Msg(@Window, "Seek to the " : place : " ID: " : @ID)
    end else
        Msg(@Window, "Seek to " : place : " failed")
    end
return

```