

Re: Another @ Get command reference

- >
- > If you use the Screen Builder to create your data entry screens, you
- > may not have to use this command at all. The Screen Builder
- > automatically generates the commands that create lists.
- >
- > The list is displayed inside a box, often with a scroll bar to the
- > right. The scroll bar lets you move quickly through the items with
- > the mouse and provides a visual indication of your position in the
- > list. Another quick way to move around in the list is to press the
- > Home key to go to the first item, or the End key to go to the last
- > item. This method works even if the list does not have a scroll bar.
- >
- > The items in the list are obtained from an array or a popup. Include
- > FROM <array> to build the list from an array. Include POPUP <popup
- > name> to build the list from a popup created with DEFINE POPUP.
- >
- > Issue READ or READ CYCLE to activate the list.
- >
- > <row, column>
- > The upper-left corner of the list is placed at the location
- > designated by <row, column>. The row and column coordinates are
- > numeric expressions that can range from 0 through the maximum number
- > of rows and columns on the desktop (FoxPro for MS-DOS), the main
- > FoxPro window (FoxPro for Windows), or a user-defined window. Rows
- > are numbered from top to bottom, and columns are numbered from left
- > to right.
- >
- > In FoxPro for Windows the position and size of the list in the main
- > FoxPro window or in a user-defined window are determined by the font
- > of the main FoxPro window or the user-defined window. Most fonts
- > Microsoft Windows uses can be displayed in a wide variety of sizes,
- > and some are proportionally spaced; thus, positions and sizes depend
- > on the current font. A row corresponds to the height of the current
- > font. A column corresponds to the average width of a letter in the
- > current font. See the Fonts topic in this online help facility for
- > additional information about fonts.
- >
- > To facilitate precise positioning of the list in the main FoxPro
- > window or in a user-defined window, you can use decimal fractions for
- > row and column coordinates in FoxPro for Windows. In FoxPro for
- > MS-DOS, decimal fractions used for row and column coordinates are
- > rounded to the nearest integer value.
- >
- > <memvar> | <field>
- > When you choose an item from a list, a value corresponding to your
- > choice is stored to the memory variable or array element <memvar>, or
- > the field <field>. <memvar> or <field> must be of numeric or
- > character type. If <memvar> or <field> is numeric, the chosen item's
- > position in the list is stored. If <memvar> or <field> is character,
- > the chosen item's prompt is stored.
- >

Re: Another @ Get command reference

- > Initial Option Selection
- > When a list is displayed, the value of <memvar> or <field>
- > determines which list option (if any) is initially selected. For
- > example, if the value of <memvar> or <field> is 4, the fourth option
- > in the list is selected when the list is activated by READ. If
- > <memvar> or <field> doesn't correspond to any of the options in the
- > list (the value is less than 1 or greater than the number of
- > options), no option is initially selected.
- >
- > FROM <array>
- > The FROM <array> clause creates a list from an array. If the array
- > is one-dimensional, the contents of the first array element is the
- > first item in the list, the contents of the second array element is
- > the second item and so on.
- >
- > If the array is two-dimensional, the elements in the first column of
- > the array are used to create the list items. The first element in
- > the first column is the first item in the list, the second element in
- > the first column is the second item and so on.
- >
- > RANGE <expN1> [, <expN2>]
- > List items by default start with the contents of the first array
- > element. You can designate a different starting element in the array
- > by including RANGE <expN1>. For example, if the array is
- > one-dimensional and <expN1> is 3, the third element in the array is
- > the first item in the list, the fourth element is the second item,
- > and so on.
- >
- > An element's position number in a two-dimensional array is
- > determined by counting along rows. For example, suppose you create
- > the following 3-by-3 array:
- >
- > a b c
- > d e f
- > g h i
- >
- > Elements a, b, and c are in position numbers 1, 2, and 3. Elements
- > d, e and f are in position numbers 4, 5, and 6, and so on. If a
- > two-dimensional array is used, only elements in the same column as
- > array element <expN1> become items in the list. For example, if
- > <expN1> is 2, the contents of elements b, e, and h are the list
- > items. If <expN1> is 5, only the contents of elements e and h are
- > included.
- >
- > If you include a starting element <expN1>, you can also specify the
- > number of elements in the list by including <expN2>. If <expN2>
- > isn't included, the contents of all array elements from the starting
- > element <expN1> through the last element in the column are items in
- > the list.
- >
- > If SHOW GETS is issued, the RANGE clause is re-evaluated. If the

Re: Another @ Get command reference

- > value of <expN1> or <expN2> has changed, the list is updated to
- > reflect the changes.
- >
- > The contents of a list can be dynamically changed. You can insert
- > and remove items by modifying the array. The ACOPY(), ADEL(), ADIR(),
- > AELEMENT(), AFIELDS(), AINS(), ALLEN(), ASCAN(), ASORT() and
- > ASUBSCRIPT() functions facilitate the manipulation of arrays.
- >
- > POPUP <popup name>
- > The list can also be built from a popup created with DEFINE POPUP.
- > Each popup item is used to create an item in the list.
- >
- > To create a list from a popup, first create the popup with DEFINE
- > POPUP. Include in the POPUP <popup name> clause the name of the popup
- > created with DEFINE POPUP.
- >
- > You can create a popup (and thus, a list) containing records from a
- > field in a table/.DBF (PROMPT FIELDS), files available on disk
- > (PROMPT FILES) or the names of the fields in a .DBF (PROMPT
- > STRUCTURE).
- >
- > The following example demonstrates how to create a list from a popup.
- > DEFINE POPUP is used to create a popup containing the names of .DBF
- > files available on disk. The .DBF names appear as options in the
- > list. MARGIN is included to provide an additional space for the mark
- > character. The SCROLL option places a scroll bar to the right of the
- > list.
- >
- > CLEAR
- > SET TALK OFF
- > STORE 1 TO mchoice
- >
- > DEFINE POPUP scrollopts FROM 0, 0 PROMPT FILES LIKE *.DBF ;
- > MARGIN SCROLL
- >
- > @ 2,2 GET mchoice POPUP scrollopts SIZE 8, 20
- >
- > READ && Activate the list.
- >
- > FUNCTION <expC1> | PICTURE <expC2>
- > To terminate the READ when an item is chosen from the list, include
- > FUNCTION '&T' or PICTURE '@&T'. Include FUNCTION '&N' or PICTURE
- > '@&N' to prevent the READ from terminating when an item is chosen
- > from the list. For example:
- >
- > ... FUNCTION '&T' ...
- > ... PICTURE '@&T' ...
- >
- > If a FUNCTION or PICTURE clause isn't included, the READ isn't
- > terminated when an item is chosen.
- >

Re: Another @ Get command reference

- > FONT <expC3> [, <expN3>]
- > The character expression <expC3> is the name of the font, and the
- > numeric expression <expN3> is the font size. For example, the
- > following clause can be used to display the items in the list in 16
- > point Roman font:
- >
- > FONT 'ROMAN', 16
- >
- > If you include the FONT clause but omit the font size <expN3>, a 10
- > point font is used.
- >
- > If the FONT clause is omitted and the list is placed in the main
- > FoxPro window, the main FoxPro window font is used. If the FONT
- > clause is omitted and the list is placed in a user-defined window,
- > the user-defined window font is used.
- >
- > If the font you specify is not available, Windows substitutes a font
- > with similar font characteristics.
- >
- > The FONT clause is ignored in FoxPro for MS-DOS.
- >
- > STYLE <expC4>
- > In FoxPro for Windows, include the STYLE clause to specify a font
- > style for the items in the list. The styles that are available for a
- > font are determined by Windows. If the font style you specify is not
- > available, Windows substitutes a font style with similar
- > characteristics.
- >
- > The font style is specified with <expC4>. If the STYLE clause is
- > omitted, the standard font style is used.
- >
- > Character Font Style
- > ÄÄÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄ
- > B Bold
- > I Italic
- > N Normal
- > O Outline
- > S Shadow
- > - Strikeout
- > U Underline
- >
- > You can include more than one character to specify a combination of
- > font styles. For example, the following clause specifies Bold Italic:
- >
- > STYLE 'BI'
- >
- > The STYLE clause is ignored in FoxPro for MS-DOS.
- >
- > DEFAULT <expr>
- > When you choose an item from the list, your choice is saved in a
- > memory variable, array element or field you specified. If you

Re: Another @ Get command reference

- > specify a memory variable that doesn't exist, it is automatically
- > created and initialized if you include the DEFAULT clause. However,
- > an array element isn't created if you specify an array element in a
- > DEFAULT clause. The DEFAULT clause is ignored if the memory variable
- > already exists or you specify a field.
- >
- > **Caution**
- > If the DEFAULT clause is omitted and the memory variable <memvar>
- > doesn't exist, the error message "Variable not found" is displayed.
- >
- > The DEFAULT expression <expr> determines the type of memory variable
- > created and its initial value. <expr> must be of numeric or
- > character type.
- >
- > **SIZE <expN4>, <expN5>**
- > The width of the list is, by default, determined by the width of the
- > longest item text in the list. The number of items in the popup or
- > array by default determines the number of items displayed in the
- > list. You can optionally specify the length and width of the list by
- > including SIZE. The length of the list in rows is specified by
- > <expN4>, and the width of the list in columns is specified by <expN5>.
- >
- > If there are more items than can be displayed in the list at one
- > time, a scroll bar is automatically placed to the right of the list
- > items.
- >
- > In FoxPro for Windows, the list's font determines the size of the
- > editing region. The list's font is specified with the FONT clause.
- > If the FONT clause is omitted, the list uses the font of its parent
- > window (the main FoxPro window or a user-defined window).
- >
- > **ENABLE | DISABLE**
- > Lists are by default enabled when READ is issued. You can prevent a
- > list from being activated when READ is issued by including DISABLE.
- > A disabled list cannot be selected and is displayed in disabled
- > colors. Use SHOW GET ENABLE to enable a disabled list.
- >
- > **MESSAGE <expC5>**
- > The MESSAGE clause character expression <expC5> is displayed when a
- > list is selected. In FoxPro for MS-DOS the message is centered on
- > the last line of the desktop and the message location can be changed
- > with SET MESSAGE.
- >
- > In FoxPro for Windows, the message is placed in the Windows-style
- > status bar. If the Windows-style status bar has been turned off with
- > SET STATUS BAR OFF, the message is placed on the last line of the
- > main FoxPro window.
- >
- > **VALID <expL1> | <expN6>**
- > You can include an optional VALID expression <expL1> or <expN6> that
- > is evaluated when an option is chosen from the list. That is, VALID

Re: Another @ Get command reference

- > isn't evaluated when you select an item, but when you actually choose
- > an item by selecting it and pressing Enter or double-clicking on the
- > item.
- >
- > Typically, <expL1> or <expN6> is a user-defined function. With a
- > user-defined function you can select, enable or disable other @ ...
- > GET input fields or objects, open a Browse window, open another data
- > entry screen or move to a new record. CLEAR READ can be included in
- > the user-defined function to terminate the READ.
- >
- > <expL1>
- > When a logical value is returned to the VALID clause, the logical
- > value is ignored and the list remains the current control. However,
- > you can specify a UDF that returns a logical value to the VALID
- > clause and then activates another object.
- >
- > <expN6>
- > A VALID clause that includes a numeric expression is used to specify
- > which object is activated after an item in the list is chosen.
- > Objects are @ ... GET input fields, check boxes, lists, popups,
- > spinners, text editing regions and each individual button in a set of
- > push, radio and invisible buttons.
- >
- > The numeric expression <expN6> has one of these effects:
- >
- > When <expN6> = 0, the list remains the active control.
- >
- > When <expN6> is positive, <expN6> indicates the number of objects
- > to advance. For example, when the list is selected and VALID returns
- > 1, the next object is activated. If <expN6> is greater than the
- > number of objects remaining, the READ is terminated (unless READ
- > CYCLE is issued to activate the objects).
- >
- > When <expN6> is negative, <expN6> specifies the number of objects
- > to move back. For example, when you're positioned on a list and
- > VALID returns -1, the previous object is activated. If <expN6> moves
- > back past the first object, the READ is terminated (unless READ CYCLE
- > is issued to activate the objects).
- >
- > WHEN <expL2>
- > The WHEN clause allows or prohibits selection of a list based on the
- > logical value of <expL2> which must evaluate to a logical true (.T.)
- > before any of the list can be selected. If <expL2> evaluates to a
- > logical false (.F.), the list cannot be selected and is skipped over
- > if placed between other objects.
- >
- > COLOR SCHEME <expN7>
- >> COLOR <color pair list>
- > If you do not include a COLOR clause, the list's colors are
- > determined by the color scheme for the desktop or the main FoxPro
- > window; if a list is placed in a user-defined window, the window's

Re: Another @ Get command reference

- > color scheme determines the list's colors.
- >
- > The colors of a list can be specified by including the number of an
- > existing color scheme in the COLOR SCHEME clause or a set of color
- > pairs in the COLOR clause.
- >
- > A color scheme is a set of 10 predefined color pairs. The color
- > pairs in a color scheme can be changed with SET COLOR OF SCHEME. In
- > FoxPro for MS-DOS the color pairs in a color scheme can also be
- > changed in the Color Picker.
- >
- > A color pair is a set of two letters separated by a forward slash.
- > The first color letter specifies the foreground color and the second
- > letter specifies the background color.
- >
- > For example, this color pair specifies a red foreground on a white
- > background:
- >
- > R/W
- >
- > For a list of colors and their corresponding color letters, see the
- > SET COLOR commands or the Color Table by Color Pair topic in this
- > online help facility.
- >
- > A color pair can also be specified with a set of 6 RGB (Red Green
- > Blue) color values separated by commas. The first 3 color values
- > specify the foreground color and the second 3 color values specify
- > the background color. The color values can range from 0 through 255.
- >
- > The R/W color pair in the example above can also be specified with
- > this RGB color pair:
- >
- > RGB(255,0,0,255,255,255)
- >
- > Color Pair List
- > Number Attribute
- > ÄÄÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄ
- >
- > 1 Disabled option
- >
- > 2 Enabled option
- >
- > 3 Border and scroll bar*
- >
- > 5 Message
- >
- > 6 Selected list item
- >
- > 9 Enabled list
- >
- > 10 Disabled list

Re: Another @ Get command reference

```
> FLOAT SHADOW SYSTEM COLOR SCHEME 8
>
> *** Fill the array cityarray with city data ***
>
> SELECT DISTINCT city FROM customer INTO ARRAY cityarray
> SELECT customer
>
> *** Define a popup that contains the structure of customer ***
>
> DEFINE POPUP popstru PROMPT STRUCTURE SCROLL MARGIN MARK CHR(16)
>
> *** Define a popup containing data from the company field ***
>
> DEFINE POPUP popfield PROMPT FIELD company SCROLL MARGIN MARK CHR(16)
> ACTIVATE WINDOW example
> @ 1,3 SAY 'Structure Popup:'
>
> *** Get information using predefined popstru popup ***
>
> @ 2,2 GET liststructure POPUP popstru SIZE 11, 20;
> DEFAULT FIELD(1,'customer') WHEN refresh();
> VALID dispitem(liststructure) COLOR SCHEME 9
> @ 13,3 SAY liststructure SIZE 1, 18
> @ 1,26 SAY 'Field Popup:'
>
> *** Get information using predefined popfield popup ***
>
> @ 2,25 GET listfield POPUP popfield SIZE 11, 20 ;
> DEFAULT company WHEN refresh() VALID dispitem(listfield);
> COLOR SCHEME 9
> @ 13,26 SAY listfield SIZE 1, 18
> @ 1,50 SAY 'Array of City Names:'
>
> *** Get information using predefined array ***
>
> @ 2,49 GET arrayitem FROM cityarray SIZE 11, 20 ;
> DEFAULT cityarray(1) WHEN refresh() VALID dispitem(arrayitem);
> COLOR SCHEME 9
> @ 13,50 SAY arrayitem SIZE 1,18
> @ 14,63 GET ok FUNCTION '*t \!OK' DEFAULT 1 SIZE 1, 6
> READ CYCLE SHOW popshow() && Activate gets
> RELEASE WINDOW example
> RELEASE POPUPS popstru, popfield
>
> FUNCTION refresh
>
> *** Refresh window information without calling subroutine ***
>
> SHOW GETS OFF
> FUNCTION dispitem
>
```

