

NAME

entry – Create and manipulate 'entry' one-line text entry widgets

SYNOPSIS

entry *pathName* ?*options*?

STANDARD OPTIONS

-background	-highlightthickness	-selectbackground
-borderwidth	-insertbackground	-selectborderwidth
-cursor	-insertborderwidth	-selectforeground
-exportselection	-insertofftime	-takefocus
-font	-insertontime	-textvariable
-foreground	-insertwidth	-xscrollcommand
-highlightbackground	-justify	
-highlightcolor	-relief	

See the **options** manual entry for details on the standard options.

WIDGET-SPECIFIC OPTIONS

Command-Line Name: **-disabledbackground**
 Database Name: **disabledBackground**
 Database Class: **DisabledBackgr ound**

Specifies the background color to use when the entry is disabled. If this option is the empty string, the normal background color is used.

Command-Line Name: **-disabledforeground**
 Database Name: **disabledForeground**
 Database Class: **DisabledF oreground**

Specifies the foreground color to use when the entry is disabled. If this option is the empty string, the normal foreground color is used.

Command-Line Name: **-invalidcommand** or **-invcmd**
 Database Name: **invalidCommand**
 Database Class: **In validCommand**

Specifies a script to eval when **-validatecommand** returns 0. Setting it to {} disables this feature (the default). The best use of this option is to set it *tobell*. See **V ALIDATION** below for more information.

Command-Line Name: **-readonlybackground**
 Database Name: **readonlyBackground**
 Database Class: **ReadonlyBackgr ound**

Specifies the background color to use when the entry is readonly. If this option is the empty string, the normal background color is used.

Command-Line Name: **-show**
 Database Name: **show**
 Database Class: **Sho w**

If this option is specified, then the true contents of the entry are not displayed in the window. Instead, each character in the entry's value will be displayed as the first character in the value of this option, such as **"*"**. This is useful, for example, if the entry is to be used to enter a password. If characters in the entry are selected and copied elsewhere, the information copied will be what is displayed, not the true contents of the entry.

Command-Line Name: **-state**
 Database Name: **state**
 Database Class: **State**

Specifies one of three states for the entry: **normal**, **disabled**, or **readonly**. If the entry is **readonly**, then the value may not be changed using widget commands and no insertion cursor will be displayed, even if the input focus is in the widget; the contents of the widget may still be selected. If the entry is **disabled**, the value may not be changed, no insertion cursor will be displayed, the contents will not be selectable, and the entry may be displayed in a different color, depending on the values of the **-disabledforeground** and **-disabledbackground** options.

Command-Line Name: **-validate**
 Database Name: **validate**
 Database Class: **Validate**

Specifies the mode in which validation should operate: **none**, **focus**, **focusin**, **focusout**, **key**, or **all**. It defaults to **none**. When you want validation, you must explicitly state which mode you wish to use. See **VALIDATION** below for more.

Command-Line Name: **-validatecommand** or **-vcmd**
 Database Name: **validateCommand**
 Database Class: **ValidateCommand**

Specifies a script to eval when you want to validate the input into the entry widget. Setting it to {} disables this feature (the default). This command must return a valid Tcl boolean value. If it returns 0 (or the valid Tcl boolean equivalent) then it means you reject the new edition and it will not occur and the **-invalidcommand** will be evaluated if it is set. If it returns 1, then the new edition occurs. See **VALIDATION** below for more information.

Command-Line Name: **-width**
 Database Name: **width**
 Database Class: **Width**

Specifies an integer value indicating the desired width of the entry window, in average-size characters of the widget's font. If the value is less than or equal to zero, the widget picks a size just large enough to hold its current text.

DESCRIPTION

The **entry** command creates a new window (given by the *pathName* argument) and makes it into an entry widget. Additional options, described above, may be specified on the command line or in the option database to configure aspects of the entry such as its colors, font, and relief. The **entry** command returns its *pathName* argument. At the time this command is invoked, there must not exist a window named *pathName*, but *pathName*'s parent must exist.

An entry is a widget that displays a one-line text string and allows that string to be edited using widget commands described below, which are typically bound to keystrokes and mouse actions. When first created, an entry's string is empty. A portion of the entry may be selected as described below. If an entry is exporting its selection (see the **-exportselection** option), then it will observe the standard X11 protocols for handling the selection; entry selections are available as type **STRING**. Entries also observe the standard Tk rules for dealing with the input focus. When an entry has the input focus it displays an *insertion cursor* to indicate where new characters will be inserted.

Entries are capable of displaying strings that are too long to fit entirely within the widget's window. In this case, only a portion of the string will be displayed; commands described below may be used to change the view in the window. Entries use the standard-**xscrollcommand** mechanism for interacting with scrollbars (see the description of the **-xscrollcommand** option for details). They also support scanning, as described below.

VALIDATION

Validation works by setting the **-validatecommand** option to a script (*validateCommand*) which will be evaluated according to the **-validate** option as follows:

none	Default. This means no validation will occur.
focus	<i>validateCommand</i> will be called when the entry receives or loses focus.
focusin	<i>validateCommand</i> will be called when the entry receives focus.
focusout	<i>validateCommand</i> will be called when the entry loses focus.
key	<i>validateCommand</i> will be called when the entry is edited.
all	<i>validateCommand</i> will be called for all above conditions.

It is possible to perform percent substitutions on the value of the **-validatecommand** and **-invalidcommand** options, just as you would in a **bind** script. The following substitutions are recognized:

%d	Type of action: 1 for insert , 0 for delete , or -1 for focus, forced or textvariable validation.
%i	Index of char string to be inserted/deleted, if any, otherwise -1.
%P	The value of the entry if the edit is allowed. If you are configuring the entry widget to have a new textvariable, this will be the value of that textvariable.
%s	The current value of entry prior to editing.
%S	The text string being inserted/deleted, if any, {} otherwise.
%v	The type of validation currently set.
%V	The type of validation that triggered the callback (key, focusin, focusout, forced).
%W	The name of the entry widget.

In general, the **-textvariable** and **-validatecommand** options can be dangerous to mix. Any problems have been overcome so that using the **-validatecommand** will not interfere with the traditional behavior of the entry widget. Using the **-textvariable** for read-only purposes will never cause problems. The danger comes when you try set the **-textvariable** to something that the **-validatecommand** would not accept, which causes **-validate** to become *none* (the **-invalidcommand** will not be triggered). The same happens when an error occurs evaluating the **-validatecommand**.

Primarily, an error will occur when the **-validatecommand** or **-invalidcommand** encounters an error in its script while evaluating or **-validatecommand** does not return a valid Tcl boolean value. The **-v alidate** option will also set itself to **none** when you edit the entry widget from within either the **-validatecommand** or the **-invalidcommand**. Such editions will override the one that was being validated. If you wish to edit the entry widget (for example set it to {}) during validation and still have the **-validate** option set, you should include the command

```
after idle {%W config -validate %v}
```

in the **-validatecommand** or **-invalidcommand** (whichever one you were editing the entry widget from). It is also recommended to not set an associated **-textvariable** during validation, as that can cause the entry widget to become out of sync with the **-textvariable**.

WIDGET COMMAND

The **entry** command creates a new Tcl command whose name is *pathName*. This command may be used to invoke various operations on the widget. It has the following general form:

```
pathName subcommand ?arg arg ...?
```

Subcommand and the *args* determine the exact behavior of the command.

INDICES

Many of the widget commands for entries take one or more indices as arguments. An index specifies a particular character in the entry's string, in any of the following ways:

<i>number</i>	Specifies the character as a numerical index, where 0 corresponds to the first character in the string.
---------------	---

anchor	Indicates the anchor point for the selection, which is set with the select from and select adjust widget commands.
end	Indicates the character just after the last one in the entry's string. This is equivalent to specifying a numerical index equal to the length of the entry's string.
insert	Indicates the character adjacent to and immediately following the insertion cursor.
sel.first	Indicates the first character in the selection. It is an error to use this form if the selection is not in the entry window.
sel.last	Indicates the character just after the last one in the selection. It is an error to use this form if the selection is not in the entry window.
@number	In this form, <i>number</i> is treated as an x-coordinate in the entry's window; the character spanning that x-coordinate is used. For example, "@0" indicates the left-most character in the window.

Abbreviations may be used for any of the forms above, e.g. "e" or "sel.f". In general, out-of-range indices are automatically rounded to the nearest legal value.

SUBCOMMANDS

The following commands are possible for entry widgets:

pathName **bbox** *index*

Returns a list of four numbers describing the bounding box of the character given by *index*. The first two elements of the list give the x and y coordinates of the upper-left corner of the screen area covered by the character (in pixels relative to the widget) and the last two elements give the width and height of the character, in pixels. The bounding box may refer to a region outside the visible area of the window.

pathName **cget** *option*

Returns the current value of the configuration option given by *option*. *Option* may have any of the values accepted by the **entry** command.

pathName **configure** *?option? ?value option value ...?*

Query or modify the configuration options of the widget. If no *option* is specified, returns a list describing all of the available options for *pathName* (see **Tk_ConfigureInfo** for information on the format of this list). If *option* is specified with no *value*, then the command returns a list describing the one named option (this list will be identical to the corresponding sublist of the value returned if no *option* is specified). If one or more *option-value* pairs are specified, then the command modifies the given widget option(s) to have the given value(s); in this case the command returns an empty string. *Option* may have any of the values accepted by the **entry** command.

pathName **delete** *first ?last?*

Delete one or more elements of the entry. *First* is the index of the first character to delete, and *last* is the index of the character just after the last one to delete. If *last* is not specified it defaults to *first*+1, i.e. a single character is deleted. This command returns an empty string.

pathName **get**

Returns the entry's string.

pathName **icursor** *index*

Arrange for the insertion cursor to be displayed just before the character given by *index*. Returns an empty string.

pathName **index** *index*

Returns the numerical index corresponding to *index*.

pathName **insert** *index string*

Insert the characters of *string* just before the character indicated by *index*. Returns an empty string.

pathName **scan** *option args*

This command is used to implement scanning on entries. It has two forms, depending on *option*:

pathName **scan mark** *x*

Records *x* and the current view in the entry window; used in conjunction with later **scan dragto** commands. Typically this command is associated with a mouse button press in the widget. It returns an empty string.

pathName **scan dragto** *x*

This command computes the difference between its *x* argument and the *x* argument to the last **scan mark** command for the widget. It then adjusts the view left or right by 10 times the difference in x-coordinates. This command is typically associated with mouse motion events in the widget, to produce the effect of dragging the entry at high speed through the window. The return value is an empty string.

pathName **selection** *option arg*

This command is used to adjust the selection within an entry. It has several forms, depending on *option*:

pathName **selection adjust** *index*

Locate the end of the selection nearest to the character given by *index*, and adjust that end of the selection to be at *index* (i.e. including but not going beyond *index*). The other end of the selection is made the anchor point for future **select to** commands. If the selection is not currently in the entry, then a new selection is created to include the characters between *index* and the most recent selection anchor point, inclusive. Returns an empty string.

pathName **selection clear**

Clear the selection if it is currently in this widget. If the selection is not in this widget then the command has no effect. Returns an empty string.

pathName **selection from** *index*

Set the selection anchor point to just before the character given by *index*. Does not change the selection. Returns an empty string.

pathName **selection present**

Returns 1 if there are characters selected in the entry, 0 if nothing is selected.

pathName **selection range** *start end*

Sets the selection to include the characters starting with the one indexed by *start* and ending with the one just before *end*. If *end* refers to the same character as *start* or an earlier one, then the entry's selection is cleared.

pathName **selection to** *index*

If *index* is before the anchor point, set the selection to the characters from *index* up to but not including the anchor point. If *index* is the same as the anchor point, do nothing. If *index* is after the anchor point, set the selection to the characters from the anchor point up to but not including *index*. The anchor point is determined by the most recent **select from** or **select adjust** command in this widget. If the selection is not in this widget then a new selection is created using the most recent anchor point specified for the widget. Returns an empty string.

pathName **validate**

This command is used to force an evaluation of the **-validatecommand** independent of the conditions specified by the **-validate** option. This is done by temporarily setting the **-validate** option to **all**. It returns 0 or 1.

pathName xview args

This command is used to query and change the horizontal position of the text in the widget's window. It can take any of the following forms:

pathName xview

Returns a list containing two elements. Each element is a real fraction between 0 and 1; together they describe the horizontal span that is visible in the window. For example, if the first element is .2 and the second element is .6, 20% of the entry's text is off-screen to the left, the middle 40% is visible in the window, and 40% of the text is off-screen to the right. These are the same values passed to scrollbars via the `-xscrollcommand` option.

pathName xview index

Adjusts the view in the window so that the character given by *index* is displayed at the left edge of the window.

pathName xview moveto fraction

Adjusts the view in the window so that the character *fraction* of the way through the text appears at the left edge of the window. *Fraction* must be a fraction between 0 and 1.

pathName xview scroll number what

This command shifts the view in the window left or right according to *number* and *what*. *Number* must be an integer. *What* must be either **units** or **pages** or an abbreviation of one of these. If *what* is **units**, the view adjusts left or right by *number* average-width characters on the display; if it is **pages** then the view adjusts by *number* screenfuls. If *number* is negative then characters farther to the left become visible; if it is positive then characters farther to the right become visible.

DEFAULT BINDINGS

Tk automatically creates class bindings for entries that give them the following default behavior. In the descriptions below, "word" refers to a contiguous group of letters, digits, or "_" characters, or any single character other than these.

- [1] Clicking mouse button 1 positions the insertion cursor just before the character underneath the mouse cursor, sets the input focus to this widget, and clears any selection in the widget. Dragging with mouse button 1 strokes out a selection between the insertion cursor and the character under the mouse.
- [2] Double-clicking with mouse button 1 selects the word under the mouse and positions the insertion cursor at the end of the word. Dragging after a double click will stroke out a selection consisting of whole words.
- [3] Triple-clicking with mouse button 1 selects all of the text in the entry and positions the insertion cursor at the end of the line.
- [4] The ends of the selection can be adjusted by dragging with mouse button 1 while the Shift key is down; this will adjust the end of the selection that was nearest to the mouse cursor when button 1 was pressed. If the button is double-clicked before dragging then the selection will be adjusted in units of whole words.
- [5] Clicking mouse button 1 with the Control key down will position the insertion cursor in the entry without affecting the selection.
- [6] If any normal printing characters are typed in an entry, they are inserted at the point of the insertion cursor.
- [7] The view in the entry can be adjusted by dragging with the middle mouse button (button 2, or button 3 in TkAqua). If the middle mouse button is clicked without moving the mouse, the selection is copied into the entry at the position of the mouse cursor.

- [8] If the mouse is dragged out of the entry on the left or right sides while button 1 is pressed, the entry will automatically scroll to make more text visible (if there is more text off-screen on the side where the mouse left the window).
- [9] The Left and Right keys move the insertion cursor one character to the left or right; they also clear any selection in the entry and set the selection anchor. If Left or Right is typed with the Shift key down, then the insertion cursor moves and the selection is extended to include the new character. Control-Left and Control-Right move the insertion cursor by words, and Control-Shift-Left and Control-Shift-Right move the insertion cursor by words and also extend the selection. Control-b and Control-f behave the same as Left and Right, respectively. Meta-b and Meta-f behave the same as Control-Left and Control-Right, respectively.
- [10] The Home key, or Control-a, will move the insertion cursor to the beginning of the entry and clear any selection in the entry. Shift-Home moves the insertion cursor to the beginning of the entry and also extends the selection to that point.
- [11] The End key, or Control-e, will move the insertion cursor to the end of the entry and clear any selection in the entry. Shift-End moves the cursor to the end and extends the selection to that point.
- [12] The Select key and Control-Space set the selection anchor to the position of the insertion cursor. They do not affect the current selection. Shift-Select and Control-Shift-Space adjust the selection to the current position of the insertion cursor, selecting from the anchor to the insertion cursor if there was not any selection previously.
- [13] Control-/ selects all the text in the entry.
- [14] Control-\ clears any selection in the entry.
- [15] The F16 key (labelled Copy on many Sun workstations) or Meta-w copies the selection in the widget to the clipboard, if there is a selection.
- [16] The F20 key (labelled Cut on many Sun workstations) or Control-w copies the selection in the widget to the clipboard and deletes the selection. If there is no selection in the widget then these keys have no effect.
- [17] The F18 key (labelled Paste on many Sun workstations) or Control-y inserts the contents of the clipboard at the position of the insertion cursor.
- [18] The Delete key deletes the selection, if there is one in the entry. If there is no selection, it deletes the character to the right of the insertion cursor.
- [19] The BackSpace key and Control-h delete the selection, if there is one in the entry. If there is no selection, it deletes the character to the left of the insertion cursor.
- [20] Control-d deletes the character to the right of the insertion cursor.
- [21] Meta-d deletes the word to the right of the insertion cursor.
- [22] Control-k deletes all the characters to the right of the insertion cursor.
- [23] Control-t reverses the order of the two characters to the right of the insertion cursor.

If the entry is disabled using the **-state** option, then the entry's view can still be adjusted and text in the entry can still be selected, but no insertion cursor will be displayed and no text modifications will take place except if the entry is linked to a variable using the **-textvariable** option, in which case any changes to the variable are reflected by the entry whatever the value of its **-state** option.

The behavior of entries can be changed by defining new bindings for individual widgets or by redefining the class bindings.

SEE ALSO

ttk::entry(n)

entry(n)

Tk Built-In Commands

entry(n)

KEYWORDS

entry, widget