NAME

pack — Geometry manager that packs around edges of cavity

SYNOPSIS

pack *option arg ?arg ...?*

DESCRIPTION

The **pack** command is used to communicate with the packer, a geometry manager that arranges the children of a parent by packing them in order around the edges of the parent. The **pack** command can have any of several forms, depending on the *option* argument:

pack *window ?window ...? ?options?*
If the first argument to **pack** is a window name (any value starting with "."), then the command is processed in the same way as **pack configure**.

pack configure *window ?window ...? ?options?*

The arguments consist of the names of one or more content windows followed by pairs of arguments that specify how to manage the content. See **THE PACKER ALGORITHM** below for details on how the options are used by the packer. The following options are supported:

-after *other*
Other must the name of another window. Use its container as the container for the content, and insert the content just after *other* in the packing order.

-anchor *anchor*
Anchor must be a valid anchor position such as **n** or **sw**; it specifies where to position each content in its parcel. Defaults to **center**.

-before *other*
Other must the name of another window. Use its container as the container for the content, and insert the content just before *other* in the packing order.

-expand *boolean*
Specifies whether the content should be expanded to consume extra space in their container. *Boolean* may have any proper boolean value, such as **1** or **no**. Defaults to 0.

-fill *style*
If a content's parcel is larger than its requested dimensions, this option may be used to stretch the content. *Style* must have one of the following values:

none
Give the content its requested dimensions plus any internal padding requested with **-ipadx** or **-ipady**. This is the default.

x
Stretch the content horizontally to fill the entire width of its parcel (except leave external padding as specified by **-padx**).

y
Stretch the content vertically to fill the entire height of its parcel (except leave external padding as specified by **-pady**).

both
Stretch the content both horizontally and vertically.

-in *container*
Insert the window at the end of the packing order for the container window given by *container*.

-ipadx *amount*
Amount specifies how much horizontal internal padding to leave on each side of the content. *Amount* must be a valid screen distance, such as **2** or **.5c**. It defaults to 0.

-ipady *amount*
Amount specifies how much vertical internal padding to leave on each side of the content. *Amount* defaults to 0.

-padx *amount*
Amount specifies how much horizontal external padding to leave on each side of the content. *Amount* may be a list of two values to specify padding for left and right separately. *Amount* defaults to 0.

-pady *amount*
Amount specifies how much vertical external padding to leave on each side of the content. *Amount* may be a list of two values to specify padding for top and bottom separately. *Amount* defaults to 0.

-side *side*
Specifies which side of the container the content will be packed against. Must be **left**, **right**, **top**, or **bottom**. Defaults to **top**.

If no **-in**, **-after** or **-before** option is specified then each of the content will be inserted at the end of the packing list for its parent unless it is already managed by the packer (in which case it will be left where it is). If one of these options is specified then all the content will be inserted at the specified point. If any of the content are already managed by the geometry manager then any unspecified options for them retain their previous values rather than receiving default values.

pack forget *window* ?*window ...*?
Removes each of the *window*s from the packing order for its container and unmaps their windows. The content will no longer be managed by the packer.

pack info *window*
Returns a list whose elements are the current configuration state of the window given by *window* in the same option-value form that might be specified to **pack configure**. The first two elements of the list are "**-in** *container*" where *container* is the window's container.

pack propagate *container* ?*boolean*?
If *boolean* has a true boolean value such as **1** or **on** then propagation is enabled for *container*, which must be a window name (see **GEOMETRY PROPAGATION** below). If *boolean* has a false boolean value then propagation is disabled for *container*. In either of these cases an empty string is returned. If *boolean* is omitted then the command returns **0** or **1** to indicate whether propagation is currently enabled for *container*. Propagation is enabled by default.

pack slaves *window*
Returns a list of all of the content windows in the packing order for *window*. The order of the content windows in the list is the same as their order in the packing order. If *window* has no content then an empty string is returned.

pack content *window*
Synonym for **pack slaves** *window*.

THE PACKER ALGORITHM

For each container the packer maintains an ordered list of content windows called the *packing list*. The **-in**, **-after**, and **-before** configuration options are used to specify the container for each content and the content's position in the packing list. If none of these options is given for a content then the content is added to the end of the packing list for its parent.

The packer arranges the content windows for a container by scanning the packing list in order. At the time it processes each content, a rectangular area within the container is still unallocated. This area is called the *cavity*; for the first content it is the entire area of the container.

For each content the packer carries out the following steps:

1. The packer allocates a rectangular *parcel* for the content along the side of the cavity given by the content's **-side** option. If the side is top or bottom then the width of the parcel is the width of the cavity and its height is the requested height of the content plus the **-ipady** and **-pady** options. For the left or right side the height of the parcel is the height of the cavity and the width is the requested width of the content plus the **-ipadx** and **-padx** options. The parcel may be enlarged further because of the **-expand** option (see **EXPANSION** below)

2. The packer chooses the dimensions of the content. The width will normally be the content's requested width plus twice its **-ipadx** option and the height will normally be the content's requested height plus twice its **-ipady** option. However, if the **-fill** option is **x** or **both** then the width of the content is expanded to fill the width of the parcel, minus twice the **-padx** option. If the **-fill** option is **y** or **both** then the height of the content is expanded to fill the width of the parcel, minus twice the **-pady** option.

3. The packer positions the content over its parcel. If the content is smaller than the parcel then the **-anchor** option determines where in the parcel the content will be placed. If **-padx** or **-pady** is non-zero, then the given amount of external padding will always be left between the content and the edges of the parcel.

Once a given content has been packed, the area of its parcel is subtracted from the cavity, leaving a smaller rectangular cavity for the next content. If a content does not use all of its parcel, the unused space in the parcel will not be used by subsequent content. If the cavity should become too small to meet the needs of a content then the content will be given whatever space is left in the cavity. If the cavity shrinks to zero size, then all remaining content on the packing list will be unmapped from the screen until the container window becomes large enough to hold them again.

EXPANSION

If a container window is so large that there will be extra space left over after all of its content have been packed, then the extra space is distributed uniformly among all of the content for which the **-expand** option is set. Extra horizontal space is distributed among the expandable content whose **-side** is **left** or **right**, and extra vertical space is distributed among the expandable content whose **-side** is **top** or **bottom**.

GEOMETRY PROPAGATION

The packer normally computes how large a container must be to just exactly meet the needs of its content, and it sets the requested width and height of the container to these dimensions. This causes geometry information to propagate up through a window hierarchy to a top-level window so that the entire sub-tree sizes itself to fit the needs of the leaf windows. However, the **pack propagate** command may be used to turn off propagation for one or more containers. If propagation is disabled then the packer will not set the requested

width and height of the packer. This may be useful if, for example, you wish for a container window to have a fixed size that you specify.

RESTRICTIONS ON CONTAINER WINDOWS

The container for each content must either be the content's parent (the default) or a descendant of the content's parent. This restriction is necessary to guarantee that the content can be placed over any part of its container that is visible without danger of the content being clipped by its parent.

PACKING ORDER

If the container for a content is not its parent then you must make sure that the content is higher in the stacking order than the container. Otherwise the container will obscure the content and it will appear as if the content has not been packed correctly. The easiest way to make sure the content is higher than the container is to create the container window first: the most recently created window will be highest in the stacking order. Or, you can use the **raise** and **lower** commands to change the stacking order of either the container or the content.

EXAMPLE

```
# Make the widgets
label .t -text "This widget is at the top"    -bg red
label .b -text "This widget is at the bottom" -bg green
label .l -text "Left\nHand\nSide"
label .r -text "Right\nHand\nSide"
text .mid
        .mid insert end "This layout is like Java's BorderLayout"
# Lay them out
pack .t   -side top    -fill x
pack .b   -side bottom -fill x
pack .l   -side left   -fill y
pack .r   -side right  -fill y
pack .mid -expand 1    -fill both
```

SEE ALSO

grid, **place**

KEYWORDS

geometry manager, location, packer, parcel, propagation, size