

## PROPERTIES II

--> (For Global, Window, and Menu properties, see chapter "Properties I")

A property is a category of feature of an object; that is to say, the value of a property helps to describe the current state of the object, and each property of each object has a set of possible values, one of which it has at all times.

The syntax for referring to objects is "[the] propertyName of objectRef", unless it is a "global" property in which case you just say "[the] propertyName".

The syntax for changing the value of a property is "set the propertyName [of objectRef] to newValue". You cannot "put" into a property (and you cannot "set" a container). -- However, some properties cannot be "set" at all (they are "read-only"); a couple can be "set" but can not be read!

A few properties can be set by a shortcut command, eg "lock screen" instead of "set the lockScreen to true".

These symbols are used to describe properties in what follows:

- = cannot be set
- ¶ = can only be set (no get!)
- Δ = linked to an Info dialog
- § = shortcut command equivalent
- ∂ = default

## Stack

NB: a stack need not be open to have its properties examined or changed.

cantDelete <boolean> -- Δ If true, the "Delete Stack..." menuitem balks if chosen; but the stack is not finder-locked.

cantAbort <boolean> -- Δ If true, "Stop" (to stop a handler in progress) is disabled

cantPeek <boolean> -- Δ If true, "(SH)-OPT (to examine where buttons and fields are) is disabled; thus "(SH)-OPT-Click (to edit a button or field script) is also disabled.

cantModify <boolean> -- Δ If true, HC will balk if the user attempts to type in a field or script, or to move or delete a button or field, and menuitems involving changing the stack (such as Compact Stack, Delete Stack, Copy, New Card, etc.) are disabled; a script can still make changes, but they are "temporary" -- no such actions are written to disk, and are removed when leaving each card. The user can still be permitted to type in fields and use paint tools "temporarily" by setting the (global) UserModify to true. This can be a good way to speed up stack response time; if values have to be saved, they can be sent to a different stack.

name -- setting also changes the name in the Finder, and in the window titlebar if the stack is open; new name must be Finder-legal (eg no colons in it) or HC complains. You can ask for the long name, eg <stack "HD:Desktop Folder:myStack">, the name, eg <stack "myStack">, or the short name, eg <myStack>; the first two constitute a valid stack reference as is, while the third requires "stack" before it, eg "stack theShortName".

•size <#bytes>

•freesize <#bytes> -- this is the space that (unfortunately) accumulates as changes are made to a stack, and has to be eliminated from time to time by compacting.

•version <comma-list> -- returns a five-item list. The first four items are all versions of HC (for their format, consult the global Version property, in chapter "Properties I"): the stack-creator, the most recent stack-compactor, the oldest modifier since compaction, and the most recent modifier. (That's according to the Reference stack. The manual says something totally different, but we all know what it's worth.) Fifth item is the Secs at the time of most recent modification.

•reportTemplates <return-list> -- names of all report templates

script <str>

scriptingLanguage

## â€¢2001

## Background

•number <theNum> -- you can refer to the background as "bg theNum". Numbers reflect the order of addition to the stack, but are adjusted to remain sequential if a bg is deleted. Thus there is no guarantee that the first card will have "bg 1" as its background.

•id <theID> -- a unique unchanging numerical identifier; you can refer to the background as "bg id theID".

name <str> -- Δ You can ask for the long name, which returns a full bgRef including full stack pathname, eg <bkgnd "myBg" of stack "HD:Desktop Folder:myStack">; or, the name, which returns a valid bgRef, eg <bkgnd "myBg">; or, the short name, which returns just the string, eg <myBg>, so that you would have to say "bg theShortName" to refer to the bg. If a bg has no name, all three forms substitute a full ref by id, eg: "bkgnd id 4399".

cantDelete <boolean> -- Δ If true, HC balks if an attempt is made to delete the only cd of the bg.

dontSearch <boolean> -- Δ If true, find actions skip all fields (cd & bg) of all cds of the bg.

showPict <boolean> -- whether the background graphics are visible. §show/hide bg pict. A script can draw in a hidden bg picture but there is no time savings.

script <str>

scriptingLanguage

## â€¢2001

## Card

•number <theNum> -- you can refer to the card as "cd theNum". The numbers reflect the order within the stack. [NB: although you can ask for the (function) "the number of cds of this bg", and although you can identify a card by its sequence within its background, eg "cd 2 of bg 1", you cannot find out what number a card is within its background! Grrrrr!]

•id -- a unique unchanging numerical identifier. You can ask for the long id, which returns a full cdRef by id, including full stack pathname, eg <card id 4625 of stack "HD:Desktop Folder:myStack">; or, the id, which returns a valid cdRef, eg <card id 4625>; or the short id, which returns just a number <theShortID>, so that you would have to say "cd id theShortID" to refer to the cd. [NB that only cards have a "long id" and "short id" different from the ID; all other objects simply return a number for all three. Grrr.]

name <str> -- Δ You can ask for the long name, which returns a full cardRef including full stack pathname, eg <card "myCd" of stack "HD:Desktop Folder:myStack">; or, the name, which returns a valid cdRef, eg <card "myCd">; or, the short name, which returns just the string, eg <myCd>, so that you would have to say "cd theShortName" to refer to the card. If a cd has no name, all three forms substitute a full ref by id, eg: "card id 4399".

•owner -- equivalent to the Name of the background of the card (and callable in the same three formats, long owner, owner, and short owner). This is how to find out what background a card belongs to without going there.

cantDelete <boolean> -- Δ If true, HC balks if an attempt is made to delete the card.

dontSearch <boolean> -- Δ If true, find actions skip all fields (cd & bg) of the card (but if false, find actions will still skip the fields of the card if the DontSearch of the bg is true).

marked <boolean> -- Δ Handy for isolating particular cards; certain commands can operate on marked cards specifically. § [un]mark cardRef.

showPict <boolean> -- whether the card graphics are visible. §show/hide cd pict. A script can draw in a hidden cd picture but there is no time savings.

script <str>

scriptingLanguage

rect -- this is really a stack feature, in that changing the Rect of any card changes the rect of all cards of the stack (the stack's "size"), and in that the user changes it through the Resize dialog of the Stack Info dialog; but it is rightly classed as a card property in that it can only be referred to for the current stack. Aspects of the Rect are also accessible via •left, •top, •right, •bottom, •topLeft, •botRight, height, width. Note that this is not a "proper" Rect, in that the "right" and "top" are in fact always 0; thus all you can really change is the Height and Width, and setting a card's Rect to, say, "10,10,100,100" actually sets it to "0,0,90,90". The card Rect represents the maximum size of the card window; a card whose Rect is larger than the card window can be examined piecemeal via the Scroll palette. Reducing the Rect of a card squeezes it towards its topLeft; fields, buttons and graphics off to the right or bottom are not lost, but they cannot be seen.

## â€¢2001

## Field

- `number <theNum>` -- you can refer to the field as "`cd|bg fld theNum`". The numbers reflect the layering order of fields within the bg or cd, with 1 being furthest away. A new field is created nearest.
- `partNumber <theNum>` -- you can refer to the field as "`cd|bg part theNum`". The numbers reflect the layering order of fields and buttons within the bg or cd, with 1 being furthest away. Changing the PartNumber changes the layering order (with automatic changing of the PartNumbers of other fields and buttons, as needed).
- `id <theID>` -- a unique unchanging numerical identifier. You can refer to the field as "`cd|bg fld id theID`".
- `name <str>` -- Δ You can ask for the long name, which returns a full fieldRef including card name and stack pathname, eg `<card|bkgn field "myField" of card "myCard" of stack "HD:Desktop Folder:myStack">`; or, the name, which returns a valid fieldRef, eg `<card|bkgn field "myField">`; or, the short name, which returns just the string, eg `<myField>`, so that you would have to say "`cd|bg fld theShortName`" to refer to the field. If a field has no name, all three forms substitute a full ref by id, eg: "`card|bkgn field id 2`" ( and if the card has no name, then in the Long Name a ref by id appears, eg: "`card id 4399`").
- `rect <theRect>` -- aspects of the Rect are also accessible via left, top, right, bottom, topLeft, botRight, height, width: changing any of the first six moves the whole field without resizing; changing the Height or Width grows or shrinks the field round its center point. The field's scrollbars, if any, are included. Measurement is from the topLeft of the card. The Right can be made less than the Left, and the Bottom can be made less than the Top; in either case, the field becomes impossible to see (though its Visible can still be true) and the Height/Width is reported as "0". The Rect can be partly or completely off the card rect, in which case as much of the field as is off the card becomes impossible to see (though its Visible can still be true).
- `loc <point>` -- the center point of the Rect.
- `visible <boolean>` -- § show/hide fieldRef
- `scroll <numPixels>` -- generates an error if not a scrolling field. If a scrolling field, this is the number of pixels of the field "hidden" above the top boundary. It can be virtually impossible for a script to scroll a field to a certain text unless you already know the scroll value for that text, or unless DontWrap and FixedLineHeight are both true, because the calculation is in pixels and because a line is not the same as a word-wrap "line".
- `script <str>`
- `scriptingLanguage`
- `style <transparent | opaque | rectangle | shadow | scrolling>` -- Δ
- `dontSearch <boolean>` -- Δ If true, find actions skip the field (but if false, find actions will still skip the field if the DontSearch of the cd or bg is true, or if the sharedText of the bg fld is true).
- `lockText <boolean>` -- Δ If true, user cannot edit text by clicking in field and typing, and field receives mouse-click messages. If false, mouse-clicking in field sends no mouse-click messages, but editing messages (EditField, ExitField, etc.) come into play. No § (grrrr).
- `autoTab <boolean>` -- Δ In a non-scrolling field, if true, a ReturnInField message reaching HC will cause a TabKey message to be sent if the insertion point is in the last fully visible line of the field. (Setting on a scrolling field does nothing.)
- `dontWrap <boolean>` -- Δ If true, text does not word-wrap (ie, only return-characters cause wrap). Always true when AutoSelect is true; setting to false sets AutoSelect to false.
- `fixedLineHeight <boolean>` -- Δ If true, field's TextHeight is used for all text regardless of size of text; if false, spacing between "lines" automatically adjusts to make room for largest text in each "line". Always true when ShowLines is true; setting to false sets ShowLines to false.
- `showLines <boolean>` -- Δ If true, dotted lines show the base of each "line", and FixedLineHeight becomes true.
- `wideMargins <boolean>` -- Δ If true, extra left- and right-margin space appears between text and boundaries.
- `sharedText <boolean>` -- Δ Useful only in background fields; for a cd field, setting has no effect, and getting returns Empty. If true, same text appears in field as instanced for all cds of same background. A bg fld shows / allows editing of text entered from background layer if SharedText is true, and shows / allows editing of text entered from card layer if SharedText is false; when value of SharedText changes, the text entered in the other layer is not lost, so in effect all bg flds possess two texts, the unshared (card) and the shared (background). If true, find actions skip the field even if DontSearch is false (but if false, find actions will still skip the field if the DontSearch of the field or cd or bg is true).
- `autoSelect <boolean>` -- Δ If true but LockText is false, DontWrap is made true, but otherwise has no effect. If true and LockText is true, DontWrap is made true, and field acts as a "list field": clicking a line with text in it (or SH-clicking or SH-dragging multiple contiguous lines, if MultipleLines is true) selects entire line(s), and deselects all other lines (as well as sending the various mouse messages). The selection is not deselected when user works or clicks elsewhere. A line consisting of just a Return has "text" in it, so be sure to eliminate such lines at the end if you don't want the user selecting a blank line.
- `multipleLines <boolean>` -- Δ If true but AutoSelect is false, or AutoSelect is true but LockText is false, has no effect. For effect if true when both AutoSelect and LockText are true, see on AutoSelect, above.
- `textAlign <right|left|center>` -- Δ, Δ = left
- `textFont <str>` -- Δ, Δ = geneva
- `textSize <integer>` -- Δ, Δ = 12
- `textHeight <numPixels>` -- Δ, Δ = 16
- `textStyle <commaList>` -- Δ, Δ = plain

NB: the Text... properties applied to a field are default values only; particular text chunks may be given other values for textFont, textSize, and textStyle, overriding the field's defaults, via the Font and Style menus, and a script can get or set the values of these three properties for a text chunk. If you get one of these for a chunk of text where the value is not the same for every char, "mixed" is returned.

## â€¢2001

## Button

- number <theNum> -- you can refer to the button as "cd|bg btn theNum". The numbers reflect the layering order of buttons within the bg or cd, with 1 being furthest away. A new button is created nearest.

- partNumber <theNum> -- you can refer to the button as "cd|bg part theNum". The numbers reflect the layering order of fields and buttons within the bg or cd, with 1 being furthest away. Changing the PartNumber changes the layering order (with automatic changing of the PartNumbers of other fields and buttons, as needed).

- id <theID> -- a unique unchanging numerical identifier. You can refer to the button as "cd|bg btn id theID".

- name <str> -- Δ You can ask for the long name, which returns a full buttonRef including card name and stack pathname, eg <card|bkgn button "myButton" of card "myCard" of stack "HD:Desktop Folder:myStack">; or, the name, which returns a valid buttonRef, eg <card|bkgn button "myButton">; or, the short name, which returns just the string, eg <myButton>, so that you would have to say "cd|bg btn theShortName" to refer to the button. If a button has no name, all three forms substitute a full ref by id, eg: "card|bkgn button id 2" (and if the card has no name, then in the Long Name a ref by id appears, eg: "card id 4399").

- rect <theRect> -- aspects of the Rect are also accessible via left, top, right, bottom, topLeft, botRight, height, width: changing any of the first six moves the whole button without resizing; changing the Height or Width grows or shrinks the button round its center point. Measurement is from the topLeft of the card. What is measured is the whole button, including the circle of a radio button, the square of a checkBox button, and the title plus item of a popup button (see on TitleWidth, below, for more about popup behaviour). The Right can be made less than the Left, and the Bottom can be made less than the Top; in either case, the button becomes impossible to see (though its Visible can still be true) and the Height/Width is reported as "0". The Rect can be partly or completely off the card rect, in which case as much of the button as is off the card becomes impossible to see (though its Visible can still be true).

- loc <point> -- the center point of the Rect.

- visible <boolean> -- § show/hide buttonRef

- enabled <boolean> -- Δ § enable/disable buttonRef; a disabled button is greyed and mouse clicks within it generate no messages, though it does still receive messages about the mouse's location

- hilite <boolean> -- if true, all pixels within the intersection of the card window and the "shape" of the button are inverted (a transparent button's "shape" is its rect; an oval button's "shape" is the oval), except that for a radio button, the black dot appears in the circle, and for a checkBox, the X appears in the box. For a popup button, the value of the Hilite makes no difference and has no effect. Can be altered via mouse if AutoHilite is true or Family is non-zero.

- script <str>

- scriptingLanguage

- style <transparent | opaque | rectangle | roundRect | shadow | checkBox | radioButton | standard | default | oval> --

Δ

- autoHilite <boolean> -- Δ If true, the Hilite becomes true while the mouse is pressed within the button, and becomes false when the mouse is released or leaves; except that for a radio button or checkBox the circle/box thickens while the mouse is pressed within the button, and the filledness of the circle/box toggles when the mouse is released within the button. "Within the button" means within its Rect, except that for oval buttons it means within the oval. The AutoHilite makes no difference to a popup button; all popup buttons display auto-hilite behaviour. NB: AutoHilite behaviour is overridden if the button is in a Family; see below.

- sharedHilite <boolean> -- Applies only to background buttons, though setting on a cd btn causes no error, and getting on a cd btn returns Empty. If true, the Hilite for the btn as instanced on each cd of the bg is always the same; if false, the Hilite for each instance of the btn is independent. Changing the value from false to true sets the Hilite to false, and references to the Hilite from then on have to do with "shared" hilite value; meanwhile, the "unshared" Hilite values are retained, and are restored if the value is changed from true to false.

- showName <boolean> -- Δ If true, and if the Name has not been set to Empty, the Short Name appears within the button's shape.

- icon <[name]num> -- Δ If not 0 (meaning it has no icon), the designated icon (if available in the resource chain) appears within the button's rect, except for radio buttons and checkBoxes, which do not display their icon. Attempting to set the icon of a popup button gives an error.

- family [0-15] -- Δ If not 0 (meaning it is not part of a family), pressing the button hilites it and leaves it hilited, and unhilites any buttons in the same layer (cd or bg) whose Family is the same. Assigning a popup button a Family has no effect.

- titleWidth -- Δ Applies only to popup buttons (can get and set for all button styles, though). Denotes how many

pixels of the button's width, starting with its Left and heading rightwards, are to be reserved for its title when it is "at rest". When the mouse is held down within the button's Rect, the Rect is hilited and the contents of the button appear as a menu whose left edge is at the button's Left+TitleWidth (thus if the TitleWidth is larger than the width of the button, the menu will appear surprisingly off to the right of the button). When the button is at rest, the region of its Rect between the button's Left+TitleWidth and its Right is covered by a shadow box, assuming the Right is larger (if not, no shadow box appears). If this shadow box is wider than about 18 pixels, a down-triangle is fully visible within it. As the shadow box is made wider still, parts of the currently selected line of the button's contents appear: eg, "...", then "F...", then "Fi...", then "First...", then "First Line".

textAlign <right|left|center> --  $\Delta$ ,  $\partial$  = center. Has no effect on checkBoxes, radio buttons, and popups.

textFont <str> --  $\Delta$ ,  $\partial$  = chicago. Has no effect if button is displaying an icon, in which case the title, if displayed, will be in Geneva.

textSize <integer> --  $\Delta$ ,  $\partial$  = 12. Has no effect if button is displaying an icon, in which case the title, if displayed, will be sized 9.

textHeight <numPixels> --  $\Delta$ ,  $\partial$  = 16, but meaningless, since a button can have only one line of text.

textStyle <commaList> --  $\Delta$ ,  $\partial$  = plain. Has no effect if button is displaying an icon, in which case the title, if displayed, will be styled plain.

NB: the Text... properties applied to a button refer to the display of its name when ShowName is true. For popup buttons, both the title and the "menu" are affected.