RESERVED SIGNS

--> invariable, reserved terms and signs...

âŒ¥2001 Keywords

Keywords are reserved words (they cannot be used as variables, and they are not trappable messages or commands) defining the structure and flow of handlers. They are:

```
  on messageName [param[,param2...]]
  function messageName [param[,param2...]]
  end messageName
```

pass messageName -- must be the same as the on/function messageName; all incoming parameters will automatically be passed on; execution of the handler will cease here, but the structure must still be completed with an end

return value -- if a function, the value will be substituted in the calling line; if a handler, the value will go into the Result; execution of the handler will cease here, but the structure must still be completed with an end

exit messageName | `to HyperCard` -- execution of the handler (or, if "to HyperCard", all handlers) will cease here, but the structure must still be completed with an end

```
  global var[,var2...]

  repeat forever
              numberOfTimes
              until condition
              while condition
              with var = startNum [down] to endNum
  exit repeat
  next repeat
  end repeat

  if ... then ... [else ...]

  if ...
  then ...
  [else ...]

  if ... then
    ...
  [else
    ...]
  end if
```

send messageString `to HyperCard` | objectInThisStack | `stack` otherStackName -- see Messages for more info. send also has two other uses: (1) for sending a 'dosc' AppleEvent to another program and (2) for running an object script written in another scripting dialect; see Communication for more info.

do expression -- expression can be a string construction, to force evaluation of the whole string before its execution; this is a way to get multiple lines, or special punctuation, into a single line. Or it can be a container, in which case the whole container is executed, line by line. [A bug in earlier versions prevented this second use (you could only "do" a line at a time), but this is now fixed.] do also has another use, for running an expression or contatiner contents written in another scripting dialect; see Communication for more info.

âŒ¥2001 Operators

Operators are reserved symbols and words used for combining "factors" into "expressions". They are given here in order of precedence; there are ten levels, and those on the same level are evaluated right to left. Parenthesised expressions are evaluated from inmost outwards.

()
- [negative number]   not   there is [not] a[n] | no
^
*   /   div   mod
+   -
&   &&
<   >   <=   ≤   >=   ≥   contains   is [not] a[n] | in | within
=   is   is not   <>   ≠
and
or

Notes: "within" is for rects, "in" is for strings. "There is" can be followed by `cd pict, bg pict,` or any of the following (plus a further specifier): `scriptingLanguage, program [ID], disk, folder, application, document, file, stack, menu, menuItem, window, cd, bg, fld, btn, part.` ("Program" means a currently running program.) "Is a" can be followed by `number, integer, point, rect, date, logical.`

âŒ¥2001 Constants

Constants are reserved words standing in the same syntax as variables, except that they cannot have anything put into them; they have pre-defined literal values.

colon comma space tab return quote empty
true false up down
eof formfeed linefeed
pi zero..ten

âŒ¥2001 Objects and Chunks

These are terms for referring to objects and parts of objects and containers.

stack bg cd part fld btn
me -- reference to the object containing the currently executing handler; with select and a field you may need to use text of me to distinguish the contents from the field itself
the target -- also, the long target, the short target. Reference to the object to which the current message was originally sent; for the format, see chapter "Properties II" under "Name" of object-type. Under some circumstances with a button or field you may need to use target (without "the") to mean the contents.
the result -- returned by "return" in a message handler, or with an error-message if certain operations fail (so that checking "the result is empty" can indicate success), or can be set by an XCMD; reset to "empty" after the next command executes, so check it first thing
it
char word item line
any first last middle second third fourth fifth sixth seventh eighth ninth tenth
next prev this
id
msg