Strings

Strings contain sequences of characters. The character set is ASCII. A string can have zero, one, or more characters, up to several thousand.

»
Unlike other systems, there is no special syntax or object type for characters. A character in Glyphic Script is just a string with one element. When you get a character out of a string, you get a string with just that one character in it.

»
The grand-parent of string is group, so strings can understand many of the messages that groups do. See the chapter on groups later in this section.

String Operations

s & t
concatenate s and t
s && t
concatenates s and t with a space between

These operators produce a new, third string that has all the characters of the first, followed by all the characters of the second. The && operator does the same thing, but places a space character between the two strings in the final result. For example:

"cat" & "apult"
fi "catapult"

"cat" && "bird"
fi "cat bird"

s @ n
returns the n-th character of s
s @ n := c
sets the n-th character of s to be c

In the both cases, the index n must be between 1 and the size of the string s. The first character of s is numbered 1. In the second example, the character c is any one character string. Generally you don't set characters in a string, you just build new strings.

"abc" @ 2.

fi "b"

```
a := "def".

a @ 3 := "g".

a.
```

fi "deg"

```
s from n
the characters from position n to the end of the s
s from n to m
the characters from position n to position m of s
s from n for m
the m characters of s starting at position n
```

These return a new string which is a copy of part of the string s. The copy always starts at the n-th position. If the to argument is given, then the copy stops at the m-th position. If the for argument is given, then it continues for m characters. If no other arguments, then the copy continues to the end of the string.

```
"the dog is blue" from 12
```

fi "blue"

```
"the dog is blue" from 2 to 7
```

fi "he dog"

```
"the dog is blue" from 7 for 5
```

fi "g is b"

```
s has c
returns true if the string s has the character c
```

The test looks for any occurrence of the character c (a one character string) in s:

```
"aeiou" has "e"
```

fi

true

```
"aeiou" has "q"
```

fi

false

Comparison

Strings can be compared with the normal comparison operators. Strings compare in the order that they would appear in the dictionary: "cat" comes after "california" and before "catatonic". When strings are compared, case makes a difference, and all uppercase letters come before all lowercase ones.

s == t
equality
s != t
inequality
s < t
s comes before t
s <= t
s comes before or is same as t
s > t
s comes after t
s >= t
s comes after or is same as t

Examples:

"cat" < "dog"

fi
true

"cat" < "california"

fi
false

"cat" >= "catatonic"

fi
false

Entering and Displaying

Strings can be directly typed in a script by enclosing the string between double quotes. When typed in a script, strings follow slightly different rules than the rest of the text of the script:

- Case distinctions matter: the string "abc" is different than "ABC".

-

Formatting is ignored: the string "abc" is the same as "abc". This is because ASCII has no way to encode formatting.

•
The string cannot have any line breaks in it. If you want to have a line break so the script formats nicely, enter two strings and use the & operator. If you want to have a line break character in the string, use the escape sequence '\r' (see below).

•
The string must be less than 250 characters. If you need to have a string which is longer than 250 characters, enter two strings and use the & operator.

Some characters that you may want to have in a string are difficult to enter. For example, if you just enter a double quote, it will end the string rather than include a double quote in the string. Certain characters can be entered by writing and escape sequence. This is a two character sequence starting with a backslash that represents one character in the string. The escape sequences are:

\n

a new line (control-j in ASCII)

\r

a carriage return (control-m in ASCII)

\t

a tab character

\\

a backslash

\"

a double quote character

»
PenPoint's text editor interprets \r as a new break character and ignores \n. Other systems (including MS-DOS, Macintosh, and Unix) interpret these characters differently from PenPoint and from each other. If you are working with strings from or destined for these other systems, you'll have to learn how they handle these characters.