2.4.1 String literals

String literals are described by the following lexical definitions:

```
stringliteral:   shortstring | longstring
shortstring:     "'" shortstringitem* "'" | '"' shortstringitem* '"'
longstring:      "'''" longstringitem* "'''" | '"""' longstringitem* '"""'
shortstringitem: shortstringchar | escapeseq
longstringitem:  longstringchar | escapeseq
shortstringchar: <any ASCII character except "\" or newline or the quote>
longstringchar:  <any ASCII character except "\">
escapeseq:       "\" <any ASCII character>
```

In plain English: String literals can be enclosed in matching single quotes (') or double quotes ("). They can also be enclosed in matching groups of three single or double quotes (these are generally referred to as *triple-quoted strings*). The backslash (\) character is used to escape characters that otherwise have a special meaning, such as newline, backslash itself, or the quote character. String literals may optionally be prefixed with a letter `r' or `R'; such strings are called raw strings and use different rules for backslash escape sequences.

In triple-quoted strings, unescaped newlines and quotes are allowed (and are retained), except that three unescaped quotes in a row terminate the string. (A ``quote'' is the character used to open the string, i.e. either ' or ".)

Unless an `r' or `R' prefix is present, escape sequences in strings are interpreted according to rules similar to those used by Standard C. The recognized escape sequences are:

Escape Sequence	Meaning
newline	Ignored
\\	Backslash (\)
\'	Single quote (')
\"	Double quote (")
\a	ASCII Bell (BEL)
\b	ASCII Backspace (BS)
\f	ASCII Formfeed (FF)
\n	ASCII Linefeed (LF)
\r	ASCII Carriage Return (CR)
\t	ASCII Horizontal Tab (TAB)
\v	ASCII Vertical Tab (VT)
ooo	ASCII character with octal value *ooo*
\x*hh...*	ASCII character with hex value *hh...*

In strict compatibility with Standard C, up to three octal digits are accepted, but an unlimited number of hex digits is taken to be part of the hex escape (and then the lower 8 bits of the resulting hex number are used in 8-bit implementations).

Unlike Standard C, all unrecognized escape sequences are left in the string unchanged, i.e., *the backslash is left in the string.* (This behavior is useful when debugging: if an escape sequence is mistyped, the resulting output is more easily recognized as broken.)

When an `r' or `R' prefix is present, backslashes are still used to quote the following character, but *all backslashes are left in the string*. For example, the string literal `r"\n"` consists of two characters: a backslash and a lowercase `n'. String quotes can be escaped with a backslash, but the backslash remains in the string; for example, `r"\""` is a valid string literal consisting of two characters: a backslash and a double quote; `r"\"` is not a value string literal (even a raw string cannot end in an odd number of backslashes). Specifically, *a raw string cannot end in a single backslash* (since the backslash would escape the following quote character). Note also that a single backslash followed by a newline is interpreted as those two characters as part of the string, *not* as a line continuation.

See About this document... for information on suggesting changes.