The l3str-format package: formatting strings of characters

The LaTeX Project[*]

Released 2021-08-27

1 Format specifications

In this module, we introduce the notion of a string ⟨*format*⟩. The syntax follows that of Python's `format` built-in function. A ⟨*format specification*⟩ is a string of the form

⟨*format specification*⟩ = [[⟨*fill*⟩]⟨*alignment*⟩][⟨*sign*⟩][⟨*width*⟩][.⟨*precision*⟩][⟨*style*⟩]

where each [. . .] denotes an independent optional part.

- ⟨*fill*⟩ can be any character: it is assumed to be present whenever the second character of the ⟨*format specification*⟩ is a valid ⟨*alignment*⟩ character.

- ⟨*alignment*⟩ can be < (left alignment), > (right alignment), ^ (centering), or = (for numeric types only).

- ⟨*sign*⟩ is allowed for numeric types; it can be + (show a sign for positive and negative numbers), - (only put a sign for negative numbers), or a space (show a space or a -).

- ⟨*width*⟩ is the minimum number of characters of the result: if the result is naturally shorter than this ⟨*width*⟩, then it is padded with copies of the character ⟨*fill*⟩, with a position depending on the choice of ⟨*alignment*⟩. If the result is naturally longer, it is not truncated.

- ⟨*precision*⟩, whose presence is indicated by a period, can have different meanings depending on the type.

- ⟨*style*⟩ is one character, which controls how the given data should be formatted. The list of allowed ⟨*styles*⟩ depends on the type.

The choice of ⟨*alignment*⟩ = is only valid for numeric types: in this case the padding is inserted between the sign and the rest of the number.

---

[*]E-mail: latex-team@latex-project.org

2   Formatting various data-types

`\tl_format:Nn` ⋆
`\tl_format:cn` ⋆
`\tl_format:nn` ⋆

`\tl_format:nn {`⟨*token list*⟩`} {`⟨*format specification*⟩`}`

Converts the ⟨*token list*⟩ to a string according to the ⟨*format specification*⟩. The ⟨*style*⟩, if present, must be `s`. If ⟨*precision*⟩ is given, all characters of the string representation of the ⟨*token list*⟩ beyond the first ⟨*precision*⟩ characters are discarded.

`\seq_format:Nn` ⋆
`\seq_format:cn` ⋆

`\seq_format:Nn {`⟨*sequence*⟩`} {`⟨*format specification*⟩`}`

Converts each item in the ⟨*sequence*⟩ to a string according to the ⟨*format specification*⟩, and concatenates the results.

`\int_format:nn` ⋆

`\int_format:nn {`⟨*intexpr*⟩`} {`⟨*format specification*⟩`}`

Evaluates the ⟨*integer expression*⟩ and converts the result to a string according to the ⟨*format specification*⟩. The ⟨*precision*⟩ argument is not allowed. The ⟨*style*⟩ can be `b` for binary output, `d` for decimal output (this is the default), `o` for octal output, `X` for hexadecimal output (using capital letters).

`\fp_format:nn` ⋆

`\fp_format:nn {`⟨*fpexpr*⟩`} {`⟨*format specification*⟩`}`

Evaluates the ⟨*floating point expression*⟩ and converts the result to a string according to the ⟨*format specification*⟩. The ⟨*style*⟩ can be

- `e` for scientific notation, with one digit before and ⟨*precision*⟩ digits after the decimal separator, and an integer exponent, following `e`;

- `f` for a fixed point notation, with ⟨*precision*⟩ digits after the decimal separator and no exponent;

- `g` for a general format, which uses style `f` for numbers in the range $[10^{-4}, 10^{\langle precision \rangle})$ and style `e` otherwise.

When there is no ⟨*style*⟩ specifier nor ⟨*precision*⟩ the number is displayed without rounding. Otherwise the ⟨*precision*⟩ defaults to 6.

3   Possibilities, and things to do

- Provide a token list formatting ⟨*style*⟩ which keeps the last ⟨*precision*⟩ characters rather than the first ⟨*precision*⟩.

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.