

Python 2.7 Format String Reference Guide

[...] indicate possible characters

(...) indicates an optional subexpression

```
{(target)(!rsa)(:([fill character][alignment][sign])(#)(0)(width)(,)(.precision)(type))}
```

call repr(), str(), or ascii() prior to formatting
fill character

0-padding

comma thousands separator

alignment: left, right, center, or padded after the sign but before digits

sign: always, negative only, or negative only but with a space for positive numbers
include base prefix (non-decimal integers only)

target

A reference to an argument, or an attribute or indexed element of an argument. The argument may be specified by name (if a keyword argument) or index. By default, arguments are displayed in order.

width

The field width (by default, determined by content).

precision

For float values, the number of digits to display after the decimal point—unless the presentation type is g or G, in which case it is the number of significant figures to display before and after the decimal point. For string values, the maximum number of characters from the input to display (longer strings will be truncated).

type

Determines the presentation of the value (see table below).

Value Type	Presentation Types
string	string (default)
integer	decimal (default) number with localized separator characters character binary octal hex, HEX
integer or float	(default is to emulate str(), with default precision of 12 for floats but not displaying trailing 0s after the decimal point) fixed point : numeric (always with a digit before the decimal point) or nan or inf Fixed point : numeric or NAN or INF exponent, Exponent : scientific notation with e or E, respectively (exponent will have a sign and at least two digits) general, General : Chooses integer, fixed-point, or scientific notation based on the magnitude and precision (default: 6). The decimal point, if present, will be preceded by 0 if not another digit; no trailing 0s will be displayed after the decimal point. number : general format, but with localized separator characters % : percentage (fixed point value*100, with percent sign)

string examples (♦ marks the end of the output):

Value	{}	{:2}	{:5}	{:>5}	{:~<5}	{:~^5}	{!r}
u'th\u0259'	th♦	th♦	th ♦	th♦	th_♦	_th_♦	u'th\u0259'♦
-1	-1♦	-1♦	-1♦	-1♦	-1_♦	_-1_♦	-1♦
3.14159	3.14159♦						
float('inf')	inf♦	inf♦	inf♦	inf♦	inf_♦	_inf_♦	inf♦
False	False♦	0♦	0♦	0♦	0_♦	_0_♦	False♦
('a', 9)	('a', 9)♦						
None	None♦	None♦	None ♦	None♦	None_♦	None_♦	None♦

see reverse for numeric examples

Python 2.7 Format String Reference Guide

numeric examples:

Value	{:g}	{:04}	{: =4}	{:4,}	{:%}	{:.0%}	{:.0f}	{:.2f}	{: >5.2f}
-1	-1	-001	- 1	-1	-100%	-100%	-1	-1.00	-1.00
0	0	0000	0	0	0%	0%	0	0.00	0.00
1e-20	1e-20	1e-20	1e-20	1e-20	0%	0%	0	0.00	0.00
2/3	0.666667	0.666666 666667	0.66666 666667	0.66666 666667	66.666667%	67%	1	0.67	0.67
3.14159	3.14159	3.14159	3.14159	3.14159	314.159%	314%	3	3.14	3.14
999999	999999	999999	999999	999,999	99999900%	99999900%	999999	999999.00	999999.00
float('inf')	inf	0inf	inf	inf	inf%	inf%	inf	inf	inf