Format specifier syntax:

%[arg\_index$][flags][width][.precision]conversion

* ***arg****\_****index***: (Optional) position of the argument in the list. First arg "1$", second "2$"…
* ***Flags***: (Optional) set of characters that modify format. flags depends on conversion.
* ***width***: (Optional) a non-negative integer, minimum number of characters to be written to output.

***precision***: (Optional) non-negative integer. For 'e', 'E', and 'f' conversions it is the number of digits **after** the decimal separator. For 'g' or 'G' conversions, it is all digits after rounding.

* ***Conversion:*** (Required) how argument s/b formatted. Conversions are data type specific.

# Conversions

|  |  |
| --- | --- |
| Conversion | Description – The Result is formatted |
| **'b', 'B'** | false if not a boolean true |
| **'s', 'S'** | "null" If argument *arg* is null, Otherwise arg.toString(). |
| **'c', 'C'** | as a Unicode character |
| **'d'** | as a decimal integer |
| **'o'** | as an octal integer |
| **'x', 'X'** | as a hexadecimal integer |
| **'e', 'E'** | **Fp:** as a decimal number in scientific notation |
| **'f'** | **fp** as a decimal number |
| **'g', 'G'** | **fp** using scientific notation or decimal format |
| **'a', 'A'** | as a hexadecimal **fp** number with significand and exponent |
| **'t', 'T'** | Prefix for date and time conversion characters. |
| **'%'** | as a literal '%' |
| **'n'** | OS specific line separator |

# Flags

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Flag | General | Char | Int | FP | Date/Time | The result will - |
| **'-'** | y | y | y | y | y | left-justified |
| **'+'** | - | - | y | y | - | include a sign! |
| **' '** | - | - | y | y | - | add space for positive #s |
| **'0'** | - | - | y | y | - | zero-padded |
| **','** | - | - | y | y | - | Locale specific |
| **'('** | - | - | y | y | - | (Negative #s) |

# Date/Time Conversions

Suffix characters for the 't' and 'T' conversions.

|  |  |
| --- | --- |
| Conver-sion | Description |
| 'H' | Hour: 24-hr, two digits zero padded, i.e. 01-12 |
| 'I' | Hour: 12-hr, 2 digits zero padded, i.e. 01-12 |
| 'k' | Hour: 24-hr, i.e. 0-23 |
| 'l' | Hour for the 12-hour clock, i.e. 1-12 |
| 'M' | Minute: two digits zero padded, i.e. 00-59 |
| 'S' | Seconds: two digits zero padded, i.e. 00-60 |
| 'L' | Millisecond: zeros padded, i.e. 000 – 999 |
| 'N' | Nanosecond: **nine** digits with zeros padded |
| 'p' | am/pm: lower case, Use 'T' to for upper case |
| 'z' | TZ: numeric offset from GMT, e.g. -0800. |
| 'Z' | time zone: abbrev. adjusted for DST |
| 's' | Seconds since the beginning of the epoch starting at 1 January 1970 00:00:00 UTC |
| 'Q' | Milliseconds since the beginning of the epoch |
| 'B' | Full month name: e.g. "January", "February". |
| 'b' | [abbreviated month name](https://docs.oracle.com/javase/7/docs/api/java/text/DateFormatSymbols.html#getShortMonths()):, e.g. "Jan", "Feb". |
| 'h' | Same as 'b'. |
| 'A' | [day of the week](https://docs.oracle.com/javase/7/docs/api/java/text/DateFormatSymbols.html#getWeekdays()), full e.g. "Sunday", "Monday" |
| 'a' | [day of the week](https://docs.oracle.com/javase/7/docs/api/java/text/DateFormatSymbols.html#getShortWeekdays()): short e.g. "Sun", "Mon" |
| 'C' | Century: two digits zero padded, i.e. 00 – 99 |
| 'Y' | Year: four digits zero padded |
| 'y' | Year: two digits, zero padded, i.e. 00 - 99. |
| 'j' | Day of year: 3 digit, zero padded, e.g. 001 – 366 |
| 'm' | Month: two digits zero padded, i.e. 01 - 13. |
| 'd' | Day of month: two digits zero padded, i.e. 01 - 31 |
| 'e' | Day of month, formatted as two digits, i.e. 1 - 31. |
| 'R' | 24-hour clock: "%tH:%tM" |
| 'T' | 24-hour clock: "%tH:%tM:%tS" |
| 'r' | 12-hour clock: "%tI:%tM:%tS %Tp" |
| 'D' | Date: "%tm/%td/%ty" |
| 'F' | Complete date: "%tY-%tm-%td". |
| 'c' | Date and time: "%ta %tb %td %tT %tZ %tY",  e.g. "Sun Jul 20 16:17:00 EDT 1969". |