

A summary of printf format specifiers

Here's a quick summary of the available `printf` format specifiers:

<code>%c</code>	character
<code>%d</code>	decimal (integer) number (base 10)
<code>%e</code>	exponential floating-point number
<code>%f</code>	floating-point number
<code>%i</code>	integer (base 10)
<code>%o</code>	octal number (base 8)
<code>%s</code>	a string of characters
<code>%u</code>	unsigned decimal (integer) number
<code>%x</code>	number in hexadecimal (base 16)
<code>%%</code>	print a percent sign
<code>\%</code>	print a percent sign

Controlling integer width with printf

The `%3d` specifier is used with integers, and means a minimum width of three spaces, which, by default, will be right-justified:

<code>printf("%3d", 0);</code>	0
<code>printf("%3d", 123456789);</code>	123456789
<code>printf("%3d", -10);</code>	-10
<code>printf("%3d", -123456789);</code>	-123456789

Left-justifying printf integer output

To left-justify integer output with `printf`, just add a minus sign (-) after the % symbol, like this:

<code>printf("%-3d", 0);</code>	0
<code>printf("%-3d", 123456789);</code>	123456789
<code>printf("%-3d", -10);</code>	-10
<code>printf("%-3d", -123456789);</code>	-123456789

The printf integer zero-fill option

To zero-fill your printf integer output, just add a zero (0) after the %symbol, like this:

<code>printf("%03d", 0);</code>	000
<code>printf("%03d", 1);</code>	001
<code>printf("%03d", 123456789);</code>	123456789
<code>printf("%03d", -10);</code>	-10
<code>printf("%03d", -123456789);</code>	-123456789

printf integer formatting

As a summary of `printf` integer formatting, here's a little collection of integer formatting examples. Several different options are shown, including a minimum width specification, left-justified, zero-filled, and also a plus sign for positive numbers.

Description	Code	Result
At least five wide	<code>printf("%5d", 10);</code>	' 10 '
At least five-wide, left-justified	<code>printf("%-5d", 10);</code>	'10 '
At least five-wide, zero-filled	<code>printf("%05d", 10);</code>	'00010 '
At least five-wide, with a plus sign	<code>printf("%+5d", 10);</code>	' +10 '
Five-wide, plus sign, left-justified	<code>printf("%-+5d", 10);</code>	'+10 '

formatting floating point numbers with printf

Here are several examples showing how to format floating-point numbers with `printf`:

Description	Code	Result
Print one position after the decimal	<code>printf("%.1f", 10.3456);</code>	<code>'10.3'</code>
Two positions after the decimal	<code>printf("%.2f", 10.3456);</code>	<code>'10.35'</code>
Eight-wide, two positions after the decimal	<code>printf("%8.2f", 10.3456);</code>	<code>' 10.35'</code>
Eight-wide, four positions after the decimal	<code>printf("%8.4f", 10.3456);</code>	<code>' 10.3456'</code>
Eight-wide, two positions after the decimal, zero-filled	<code>printf("%08.2f", 10.3456);</code>	<code>'00010.35'</code>
Eight-wide, two positions after the decimal, left-justified	<code>printf("%-8.2f", 10.3456);</code>	<code>'10.35 '</code>
Printing a much larger number with that same format	<code>printf("%-8.2f", 101234567.3456);</code>	<code>'101234567.35'</code>

printf string formatting

Here are several examples that show how to format string output with `printf`:

Description	Code	Result
A simple string	<code>printf("%s", "Hello");</code>	'Hello'
A string with a minimum length	<code>printf("%10s", "Hello");</code>	' Hello'
Minimum length, left-justified	<code>printf("%-10s", "Hello");</code>	'Hello '

printf special characters

The following character sequences have a special meaning when used as `printf` format specifiers:

<code>\a</code>	audible alert
<code>\b</code>	backspace
<code>\f</code>	form feed
<code>\n</code>	newline, or linefeed
<code>\r</code>	carriage return
<code>\t</code>	tab

printf Format Specifiers

<code>\v</code>	vertical tab
<code>\\</code>	backslash

As you can see from that last example, because the backslash character itself is treated specially, you have to print two backslash characters in a row to get one backslash character to appear in your output.

Here are a few examples of how to use these special characters:

Description	Code	Result
Insert a tab character in a string	<code>printf("Hello\tworld");</code>	Hello world
Insert a newline character in a string	<code>printf("Hello\nworld");</code>	Hello world
Typical use of the newline character	<code>printf("Hello world\n");</code>	Hello world
A DOS/Windows path with backslash characters	<code>printf("C:\\Windows\\System32\\");</code>	C:\windows\System32\