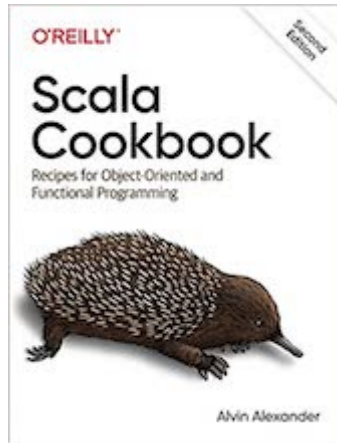




## 2021 cookbook



#1 new release!



Buy Alvin A Coffee

A  
`printf`  
formatreference page (cheat  
sheet) (C, Java, Scala,  
etc.)

By Alvin Alexander. Last updated: January 29, 2022

Does your  
business

have a small

Scala or Flutter side project?

You can now hire Alvin Alexander.

Contact me at (al) at  
valleyprogramming (dot) com for  
details.

Summary: This page is a *printf* formatting cheat sheet. I originally created this cheat sheet for my own purposes, and then thought I would share it here.

A great thing about the `printf` formatting syntax is that the format specifiers you can use are very similar — if not identical — between different languages, including C, C++, Java, Perl, PHP, Ruby, Scala, and others. This means that your `printf` knowledge is reusable, which is a good thing.

## `printf` formatting with Perl and Java

In this cheat sheet I'll show all the examples using Perl, but at first it might help to see one example using both Perl and Java. Therefore, here's a simple Perl `printf` example to get things started:

```
printf("the %s jumped over the %s, %d times", "cow", "moon", 2);
```

And here are three different Java `printf` examples, using different string formatting methods that are available to you in the Java programming language:

```
System.out.format("the %s jumped over the %s, %d times", "cow", "moon", 2);
System.err.format("the %s jumped over the %s, %d times", "cow", "moon", 2);
String result = String.format("the %s jumped over the %s, %d times", "cow", "moon",
```

As you can see in that last `String.format` example, that line of code doesn't print any output, while the first line prints to standard output, and the second line prints to standard error.

In the remainder of this document I'll use Perl examples, but again, the actual format specifier strings can be used in many different languages.

## A summary of printf format specifiers

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

%c	character
%d	decimal (integer) number (base 10)
%e	exponential floating-point number
%f	floating-point number
%i	integer (base 10)
%o	octal number (base 8)
%s	a string of characters
%u	unsigned decimal (integer) number
%x	number in hexadecimal (base 16)
%%	print a percent sign
\%	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 '
		'00010 '

At least five-wide, zero-filled	<code>printf("%05d", 10);</code>	
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>	'10.3'
Two positions after the decimal	<code>printf("%.2f", 10.3456);</code>	'10.35'
Eight-wide, two positions after the decimal	<code>printf("%8.2f", 10.3456);</code>	' 10.35'
Eight-wide, four positions after the decimal	<code>printf("%8.4f", 10.3456);</code>	' 10.3456'
Eight-wide, two positions after the decimal, zero-filled	<code>printf("%08.2f", 10.3456);</code>	'00010.35'
Eight-wide, two positions after the decimal, left-justified	<code>printf("%-8.2f", 10.3456);</code>	'10.35 '
Printing a much larger number with that same format	<code>printf("%-8.2f", 101234567.3456);</code>	'101234567.35'

## 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
<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\

## Related printf content

Here are a few links to other “printf” tutorials on my website:

- [Perl printf formatting examples](#)
- [Java printf - formatting Java printing with System.out.format](#)

And here’s a link to [the Java formatter javadoc](#).

category: [java](#)

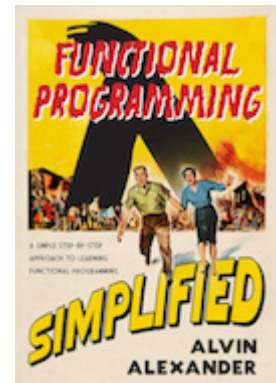
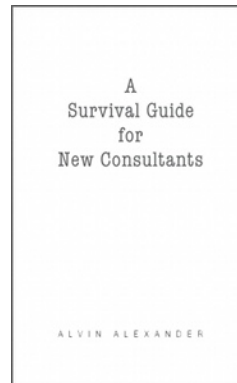
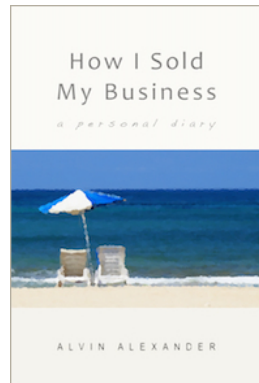
tags: [c](#) [c++](#) [format](#) [java](#) [perl](#) [printf](#) [reference](#) [ruby](#) [source](#) [sprintf](#) [string](#) [cheatsheet](#) [scala](#)

### related

- [Java String formatting with the String.format method \(like ‘sprintf’\)](#)
- [Java ‘printf’ - formatting output with System.out.format](#)
- [An example of Scala’s `f` string interpolator](#)

- Does Scala have a String variable substitution syntax like Ruby?
- Java SimpleDateFormat example: How to convert a Date to a formatted String

### books i've written



### what's new

- Functional Programming, Simplified (a Scala book)
- Scala: GraalVM native-image: Warning: Image 'ffx' is a fallback image that requires a JDK
- What is it on this planet that needs doing that I know something about?
- Dart: How to compile/run a project with a pubspec.yaml file (dependencies)
- The Rise, by Lisa Guyer and Sully Erna

## Links:

[front page](#) [photo gallery](#) [scala](#) [linux/unix](#) [alvin on twitter](#) [privacy terms & conditions](#)

alvinalexander.com  
is owned and operated by  
[Valley Programming, LLC](#)

In regards to links to Amazon.com, As an Amazon Associate  
I (Valley Programming, LLC) earn from qualifying purchases

This website uses cookies: [learn more](#)

This website uses cookies: [learn more](#)

alvinalexander.com is owned and operated by [Valley Programming, LLC](#)

In regards to links to Amazon.com, “As an Amazon Associate  
I (Valley Programming, LLC) earn from qualifying purchases”

## java

- [java tutorials](#)
- [android tutorials](#)
- [java applets](#)
- [java faqs](#)
- [java source code](#)
- [intellij idea](#)
- [eclipse ide](#)
- [jdbc](#)
- [jsp's](#)
- [java servlets](#)
- [jfc-swing](#)
- [kotlin](#)

## perl

- [perl tutorials](#)
- [perl faqs](#)
- [perl recipes](#)
- [perl tutorials](#)

## Unix

- [linux tutorials](#)
- [unix by example](#)
- [tutorials](#)
- [man \(help\) pages](#)

## programming

- [scala tutorials](#)
- [drupal](#)

- [flutter/dart](#)
- [git](#)
- [haskell](#)
- [oop](#)
- [php](#)
- [mysql](#)
- [postgresql](#)
- [ruby](#)
- [sencha](#)
- [best practices](#)
- [programming](#)
- [software dev](#)
- [testing](#)
- [uml](#)

## **misc**

- [privacy policy](#)
- [terms & conditions](#)
- [apple/mac/iphone](#)
- [career](#)
- [design](#)
- [funny/humor](#)
- [function point](#)
- [analysis \(fpa\)](#)
- [fpa tutorial](#)
- [gadgets](#)
- [technology](#)
- [LaTeX](#)
- [alaska](#)
- [colorado](#)
- [cvs tutorials](#)
- [wincvs tutorial](#)

## **other**

- [contact me](#)
- [rss feed](#)
- [my photos](#)
- [inspirational quotes](#)
- [investing](#)
- ['personal' posts](#)
- [science](#)
- [zen, mindfulness](#)
- [life in alaska](#)
- [living in talkeetna, alaska](#)