

---

## Java Primitive Data Types

Type Name	Description	Storage	Smallest Value	Largest Value
<code>boolean</code>	Logical values	1 bit	<code>false</code> (0)	<code>true</code> (1)
<code>char</code>	Single characters	2 bytes	NUL ( <code>\0</code> )	<i>depends</i> <sup>1</sup>
<code>byte</code>	Very small integers	1 byte	-128	127
<code>short</code>	Small integers	2 bytes	-32,768	32,767
<code>int</code>	Integers	4 bytes	-2,147,483,648	2,147,483,647
<code>long</code>	Large integers	8 bytes	-2 <sup>63</sup>	2 <sup>63</sup> - 1 <sup>2</sup>
<code>float</code>	Low-precision reals	4 bytes	1.4013e-45	3.4028e+38 <sup>3</sup>
<code>double</code>	Higher-precision reals	8 bytes	4.9407e-324	1.7977e+308 <sup>3</sup>

<sup>1</sup> Java encodes `char` using Unicode, and the maximum number of available symbols varies depending on the language being used. The first 128 characters of Unicode match the 7-bit ASCII standard.

<sup>2</sup> -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807

<sup>3</sup> same range in negative numbers

---

## Some Java Format Specifier Codes

These are part of the format string used within `System.out.printf()` to format output values. Commonly-used format specifiers have the form `%[width][.precision]code` (e.g. `%d` and `%5.2f`). Not shown here are format flags, which are used for effects such as left-justification. While these are borrowed from C, Java is more strict about how they are used.

Code	Formats	Example Use	Corresponding Output
<code>d</code>	Integers (Base 10)	<code>("%5d", 29)</code>	29
<code>x</code>	Integers (Base 16)	<code>("%x %x", 29, 32)</code>	1D 20
<code>o</code>	Integers (Base 8)	<code>("%o", 29)</code>	35
<code>f</code>	Floating-point	<code>("%7.2f", 874.9163)</code>	874.92
<code>e</code>	Exponential Floating-point	<code>("%e", 874.9163)</code>	8.749163e+02
<code>c</code>	Characters	<code>("%c", 'Y')</code>	Y
<code>s</code>	Strings	<code>("%10s", "Hi")</code>	Hi