



Java Numbers

[< Previous](#)[Next >](#)

Numbers

Primitive number types are divided into two groups:

Integer types stores whole numbers, positive or negative (such as 123 or -456), without decimals. Valid types are `byte`, `short`, `int` and `long`. Which type you should use, depends on the numeric value.

Floating point types represents numbers with a fractional part, containing one or more decimals. There are two types: `float` and `double`.

Even though there are many numeric types in Java, the most used for numbers are `int` (for whole numbers) and `double` (for floating point numbers). However, we will describe them all as you continue to read.

Integer Types

Byte

The **byte** data type can store whole numbers from -128 to 127. This can be used instead of **int** or other integer types to save memory when you are certain that the value will be within -128 and 127:

Example

```
byte myNum = 100;  
System.out.println(myNum);
```

Try it Yourself »

Short

The **short** data type can store whole numbers from -32768 to 32767:

Example

```
short myNum = 5000;  
System.out.println(myNum);
```

Try it Yourself »

Int

The **int** data type can store whole numbers from -2147483648 to 2147483647. In general, and in our tutorial, the **int** data type is the preferred data type when we create variables with a numeric value.

Example

```
int myNum = 100000;  
System.out.println(myNum);
```

Try it Yourself »

Long

The `long` data type can store whole numbers from -9223372036854775808 to 9223372036854775807. This is used when `int` is not large enough to store the value. Note that you should end the value with an "L":

Example

```
long myNum = 15000000000L;  
System.out.println(myNum);
```

Try it Yourself »

Floating Point Types

You should use a floating point type whenever you need a number with a decimal, such as 9.99 or 3.14515.

The `float` and `double` data types can store fractional numbers. Note that you should end the value with an "f" for floats and "d" for doubles:

Float Example

```
float myNum = 5.75f;  
System.out.println(myNum);
```

Try it Yourself »

Double Example

```
double myNum = 19.99d;  
System.out.println(myNum);
```

Try it Yourself »

Use `float` or `double` ?

The **precision** of a floating point value indicates how many digits the value can have after the decimal point. The precision of `float` is only six or seven decimal digits, while `double` variables have a precision of about 15 digits. Therefore it is safer to use `double` for most calculations.

Scientific Numbers

A floating point number can also be a scientific number with an "e" to indicate the power of 10:

Example

```
float f1 = 35e3f;  
double d1 = 12E4d;  
System.out.println(f1);  
System.out.println(d1);
```

Try it Yourself »

< Previous

Next >