## Java For Loop

When you know exactly how many times you want to loop through a block of code, use the for loop instead of a while loop:

### Syntax

for (*statement 1*; *statement 2*; *statement 3*) {

*// code block to be executed*

}

****Statement 1**** is executed (one time) before the execution of the code block.

****Statement 2**** defines the condition for executing the code block.

****Statement 3**** is executed (every time) after the code block has been executed.

The example below will print the numbers 0 to 4:

### Example

for (int i = 0; i < 5; i++) {

System.out.println(i);}

#### Example explained

Statement 1 sets a variable before the loop starts (int i = 0).

Statement 2 defines the condition for the loop to run (i must be less than 5). If the condition is true, the loop will start over again, if it is false, the loop will end.

Statement 3 increases a value (i++) each time the code block in the loop has been executed.

## Another Example

This example will only print even values between 0 and 10:

### Example

for (int i = 0; i <= 10; i = i + 2) {

System.out.println(i);}

## For-Each Loop

There is also a "**for-each**" loop, which is used exclusively to loop through elements in an ****array****:

### Syntax

for (*type* *variableName* : *arrayName*) {

*// code block to be executed*}

The following example outputs all elements in the ****cars**** array, using a "**for-each**" loop:

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};for (String i : cars) {

System.out.println(i);}

# Java Break and Continue

## Java Break

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

The break statement can also be used to jump out of a ****loop****.

This example jumps out of the loop when i is equal to 4:

### Example

for (int i = 0; i < 10; i++) {

if (i == 4) {

break;

}

System.out.println(i);}

## Java Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

This example skips the value of 4:

### Example

for (int i = 0; i < 10; i++) {

if (i == 4) {

continue;

}

System.out.println(i);}

## Break and Continue in While Loop

You can also use break and continue in while loops:

### Break Example

int i = 0;while (i < 10) {

System.out.println(i);

i++;

if (i == 4) {

break;

}}

### Continue Example

int i = 0;while (i < 10) {

if (i == 4) {

i++;

continue;

}

System.out.println(i);

i++;}

# Java Arrays

[❮ Previous](https://www.w3schools.com/java/java_break.asp)[Next ❯](https://www.w3schools.com/java/java_methods.asp)

## Java Arrays

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

To declare an array, define the variable type with ****square brackets****:

String[] cars;

We have now declared a variable that holds an array of strings. To insert values to it, we can use an array literal - place the values in a comma-separated list, inside curly braces:

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

To create an array of integers, you could write:

int[] myNum = {10, 20, 30, 40};

## Access the Elements of an Array

You access an array element by referring to the index number.

This statement accesses the value of the first element in cars:

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};System.out.println(cars[0]);// Outputs Volvo

****Note:**** Array indexes start with 0: [0] is the first element. [1] is the second element, etc.

## Change an Array Element

To change the value of a specific element, refer to the index number:

### Example

cars[0] = "Opel";

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

cars[0] = "Opel";System.out.println(cars[0]);// Now outputs Opel instead of Volvo

## Array Length

To find out how many elements an array has, use the length property:

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};System.out.println(cars.length);// Outputs 4

## Loop Through an Array

You can loop through the array elements with the for loop, and use the length property to specify how many times the loop should run.

The following example outputs all elements in the ****cars**** array:

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};for (int i = 0; i < cars.length; i++) {

System.out.println(cars[i]);}

## Loop Through an Array with For-Each

There is also a "**for-each**" loop, which is used exclusively to loop through elements in arrays:

### Syntax

for (*type* *variable* : *arrayname*) {

...}

The following example outputs all elements in the ****cars**** array, using a "**for-each**" loop:

### Example

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};for (String i : cars) {

System.out.println(i);}

The example above can be read like this: ****for each**** String element (called ****i**** - as in ****i****ndex) in ****cars****, print out the value of****i****.

If you compare the for loop and ****for-each**** loop, you will see that the **for-each** method is easier to write, it does not require a counter (using the length property), and it is more readable.

## Multidimensional Arrays

A multidimensional array is an array containing one or more arrays.

To create a two-dimensional array, add each array within its own set of ****curly braces****:

### Example

int[][] myNumbers = { {1, 2, 3, 4}, {5, 6, 7} };

****myNumbers**** is now an array with two arrays as its elements.

To access the elements of the ****myNumbers**** array, specify two indexes: one for the array, and one for the element inside that array. This example accesses the third element (2) in the second array (1) of myNumbers:

### Example

int[][] myNumbers = { {1, 2, 3, 4}, {5, 6, 7} };int x = myNumbers[1][2];System.out.println(x); // Outputs 7

We can also use a for loop inside another for loop to get the elements of a two-dimensional array (we still have to point to the two indexes):

### Example

public class MyClass {

public static void main(String[] args) {

int[][] myNumbers = { {1, 2, 3, 4}, {5, 6, 7} };

for (int i = 0; i < myNumbers.length; ++i) {

for(int j = 0; j < myNumbers[i].length; ++j) {

System.out.println(myNumbers[i][j]);

}  
 }

}}