public class Main {

public static void main(String[] args) {

System.out.println("Hello World");

System.out.println(3 + 3);

}

}

Declaring (Creating) Variables

*type variableName = value;*

String name = "John";

System.out.println(name);

2) int myNum;

myNum = 15;

System.out.println(myNum);

3) int x, y, z;

x = y = z = 50;

System.out.println(x + y + z);

## Identifiers

All Java **variables** must be **identified** with **unique names**.

These unique names are called **identifiers**.

The general rules for naming variables are:

* Names can contain letters, digits, underscores, and dollar signs
* Names must begin with a letter
* Names should start with a lowercase letter, and cannot contain whitespace
* Names can also begin with $ and \_
* Names are case-sensitive ("myVar" and "myvar" are different variables)
* Reserved words (like Java keywords, such as int or boolean) cannot be used as names

int minutesPerHour = 60;

## Calculate the Area of a Rectangle

int length = 4;

int width = 6;

int area;

// Calculate the area of a rectangle

area = length \* width;

// Print variables

System.out.println("Length is: " + length);

System.out.println("Width is: " + width);

System.out.println("Area of the rectangle is: " + area);

## Java Data Types

As explained in the previous chapter, a [variable](https://www.w3schools.com/java/java_variables.asp) in Java must be a specified data type:

int myNum = 5; // Integer (whole number)

float myFloatNum = 5.99f; // Floating point number

char myLetter = 'D'; // Character

boolean myBool = true; // Boolean

String myText = "Hello"; // String

* Primitive data types - includes byte, short, int, long, float, double, boolean and char
* Non-primitive data types - such as [String](https://www.w3schools.com/java/java_strings.asp), [Arrays](https://www.w3schools.com/java/java_arrays.asp) and [Classes](https://www.w3schools.com/java/java_classes.asp) (you will learn more about these in a later chapter)
* Primitive types are predefined (already defined) in Java. Non-primitive types are created by the programmer and is not defined by Java (except for String).
* Non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.
* A primitive type always has a value, while non-primitive types can be null.
* A primitive type starts with a lowercase letter, while non-primitive types starts with an uppercase letter

## Java Type Casting

Type casting is when you assign a value of one primitive data type to another type.

**Widening Casting** (automatically) - converting a smaller type to a larger type size  
byte -> short -> char -> int -> long -> float -> double

**Narrowing Casting** (manually) - converting a larger type to a smaller size type  
double -> float -> long -> int -> char -> short -> byte

## java Operators

Operators are used to perform operations on variables and values.

1) int x = 100 + 50;

2) int x = 10;

x += 5;

## Java Strings

Strings are used for storing text.

A String variable contains a collection of characters surrounded by double quotes:

String greeting = "Hello";

2) String x = "10";

int y = 20;

String z = x + y;

int myAge = 25;

int votingAge = 18;

if (myAge >= votingAge) {

System.out.println("Old enough to vote!");

} else {

System.out.println("Not old enough to vote.");

}

### Syntax

if (*condition*) {

*// block of code to be executed if the condition is true*

}

### Syntax --else:

if (*condition*) {

*// block of code to be executed if the condition is true*

} else {

*// block of code to be executed if the condition is false*

}

### Syntax---else if

if (*condition1*) {

*// block of code to be executed if condition1 is true*

} else if (*condition2*) {

*// block of code to be executed if the condition1 is false and condition2 is true*

} else {

*// block of code to be executed if the condition1 is false and condition2 is false*

}

Syntax---short if else(ternary op)

int time = 20;

String result = (time < 18) ? "Good day." : "Good evening.";

System.out.println(result);

### Syntax --switch

switch(expressionexpression) {

case x xx::

// code block

break;

case yyy:

// code block

break;

default:

// code block

}

### Syntax

while (*condition*) {

*// code block to be executed*

}

int i = 0;

while (i < 5) {

System.out.println(i);

i++;

}

Syntax—do while loop

do {

*// code block to be executed*

}

while (*ccondondition*);

ex: int i = 0;  
do {

System.out.println(i);

i++;

}

while (i < 5);

### Syntax:for

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

*// code block to be executed*

}

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

System.out.println(i);

}

[Try it Yourself »](https://www.w3schools.com/java/tryjava.asp?filename=demo_for_loop_even)

even or odd:

### Example

int myNum = 5;

if (myNum % 2 == 0) {

System.out.println(myNum + " is even");

} else {

System.out.println(myNum + " is odd");

}

For each loop

int[] myNumbers = {1, 2, 3, 4, 5};  
for (int i : myNumbers) {  
  System.out.println(i);  
}

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

if (i == 4) {

break;

}

System.out.println(i);

}

0,1,2,3

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

if (i == 4) {

continue;

}

System.out.println(i);

}

0,1,2,3,5,6,7,8,9

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

String [ ]cars;

## Multidimensional Arrays Multidimensional arrays are useful when you want to store data as a tabular form, like a table with rows and columns.

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

A **method** is a block of code which only runs when it is called.

Methods are used to perform certain actions, and they are also known as **functions**.

static means that the method belongs to the Main class a

void means that this method