

Hello, world!

```
/* HelloWorld.java
 * Purpose:
 * The classic "Hello, world!" program.
 * It prints a message to the screen.
 * Author: Jane Programmer
 * as derived from Kernighan and Richie
 */
class HelloWorld {
    public static void main (String[] args) {
        System.out.println("Hello, world!");
    }
}
```

Compiling a program

- Source code - HelloWorld.java
 - viewed with an editor
 - understandable by a human
- Object code - HelloWorld.class
 - for Java, this is machine independent byte code
 - compilers for other languages produce machine code
 - this is also called the binary form or executable

Compiling

- Create HelloWorld.java with an editor
- Execute the command:

```
javac HelloWorld.java
```

HelloWorld.java → Java Compiler → HelloWorld.class
(source) (bytecode)

Running your Java program

- Once it compiles with no errors, type:

```
java HelloWorld
```
- Notice it is not HelloWorld.class.
- The name here must be the name found after the keyword `class` in your programs source file. In general it should be the same as the name of the file, minus the extension.

Keywords vs Identifiers

- Keywords cannot be used for any other purpose. Examples include: `class`, `int`, `public`, `static`, `void`
- Identifiers are the names for things that you get to make up. They must start with a letter and then may include digits. `$` and `_` (underscore) can be used but should be avoided.

Literals

These are strings of symbols that represent “literal” data values.

123 is an integer literal

1.23 is a floating point literal

"123" is a String literal as is "class"

but `class` is a keyword and `Class` is an identifier

Operators and punctuation

- Operators are symbols like: +, -, / (division), and * (multiply)
- Punctuation includes symbols like: (,), {, }, and ; (semicolon)

Data types and variables

- Data types - simple to complex
 - int - for integers or whole numbers
 - double - for numbers with fractional parts
 - String - for text
 - Button - a button on a GUI
 - Point - for representing points in a plane
- Variables store data in named locations
 - every variable must have a declared type

Primitive types vs Classes

- Java has eight primitive types: byte, short, int, long, float, double, char, boolean
- Primitive types have literal values and can be manipulated with built-in operators. E.g.
 $2 + 3$
- Class type values are created with the operator new.

```
new Button("Quit")
```

Declaring Variables

```
int count, total;  
String sentence;  
boolean done;  
Button clickToExit;
```

Initializing Variables

```
int count = 10, total = 0;  
String sentence = "Hello there.";  
boolean done = false;  
Button clickToExit =  
    new Button("Exit");
```

```
// HelloWorld2.java - variable declarations  
class HelloWorld2 {  
    public static void main (String[] args) {  
        String word1, variable;  
        String word2, sentence;  
  
        word1 = "Hello, ";  
        word2 = "world!";  
        sentence = word1.concat(word2);  
        System.out.println(sentence);  
    }  
}
```

```
// StringVsId.java
// contrast strings and identifiers
class StringVsId {
    public static void main(String[] args) {
        String hello = "Hello, world!";
        String stringVary;
        stringVary = hello;
        System.out.println(stringVary);
        stringVary = "hello";
        System.out.println(stringVary);
    }
}
```

User Input

- Dissect SimpleInput.java
 - tio
 - use + to break up long string literals
 - be sure an include a prompt
 - use meaningful variable names
 - * is multiplication

```
// SimpleInput.java-reading numbers from the keyboard
import tio.*; // use the package tio

class SimpleInput {
    public static void main (String[] args) {
        int width, height, area;

        System.out.println("type two integers for" +
            " the width and height of a box");
        width = Console.in.readInt();
        height = Console.in.readInt();
        area = width * height;
        System.out.print("The area is ");
        System.out.println(area);
    }
}
```

Calling predefined methods

- method - a group of instructions with a name. E.g. main(), println(), readInt().
- Some methods require some data values upon which to operate.
E.g `System.out.println(width);`
- These data values are called *parameters*.
- Parameters are *passed* to methods.
- Some also return a value.
E.g. `x = Math.sqrt(y);`

print() and println()

```
System.out.println("type two integers for" +
    " the width and height.");

System.out.print("type two integers for the");
System.out.println(" width and height.");

        Illegal
System.out.println("type two integers for
    the width and height.");
```

Inserting newlines in a string

```
System.out.println("One\nword\nper\nline.");
```

Output

```
One
word
per
line.
```

Numeric Types

- byte - 8 bits
- short - 16 bits
- char - 16 bits (no sign)
- int - 32 bits - +/-2 billion
- long - 64 bits - 19 decimal digits
- float - 32 bits - +/-10⁻⁴⁵ to +/-10⁺³⁸ - 7 digits
- double - 64 bits - +/-10⁻³²⁴ to +/-10⁺³⁰⁸ - 15 digits

Numbers vs Strings

- The bit pattern in the computers memory is different for "1234" and 1234.
- The computer needs to know how to interpret the bits, hence the type for variables.
- This is just like needing to know what language is being used. Does "pie" mean something good to eat (English), or foot (Spanish)?

Integer arithmetic

- Dissect MakeChange.java
 - variable declaration
 - user input
 - simple expression
 - integer division
 - remainder or modulo

```
// MakeChange.java - change in dimes and pennies
import tio.*; // use the package tio

class MakeChange {
    public static void main (String[] args) {
        int price, change, dimes, pennies;

        System.out.println("type price (0:100):");
        price = Console.in.readInt();
        change = 100 - price; //how much change
        dimes = change / 10; //number of dimes
        pennies = change % 10; //number of pennies
        System.out.print("The change is : ");
        System.out.println(dimes + " dimes " +
            pennies + " pennies");
    }
}
```

Precedence and associativity

- * and / have higher precedence than + and -
 - What is the value of $7 + 5 * 3$?
- For equal precedence operators, most are left associative.
 - What is the value of $100 / 5 * 2$?
- Use parentheses can be used to override the normal rules. E.g. $100 / (5 * 2)$