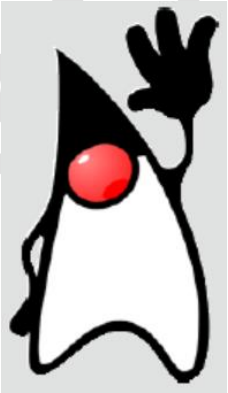
JAVA FOUNDATIONS 1Z0-811

ORACLE ACADEMY



2 DE SEPTIEMBRE DE 2025

https://academy.oracle.com/

HTTPS://GITHUB.COM/ISC-UPA/2025-3-TIID3C-POO

Contenido

1. Introduction 2
1.1. Technological Requirements: 2
1.2. Create Java Project: 3
1.3. Setting Up Java 4
2. Java Basics 6
2.1. The Software Development Process 6
2.2. What is my Program Doing? 7
2.3. Introduction to Object-Oriented Programming Concepts 7
3. Java Data Types 8
3.1. What is a Variable? 8
String x ="Sam"; 8
3.2. Numeric Data 8
Rules of Precedence 9
3.3. Textual Data 10
Primitives 10
Escape Sequence 11
3.4. Converting Between Data Types 11
3.5. Keyboard Input 13
4. Java Methods and Library Classes 14
5. Decision Statements 14
6. Loop Constructs 14
7. Creating Classes 14
8. Arrays and Exceptions 14
9. JavaFX 14

→

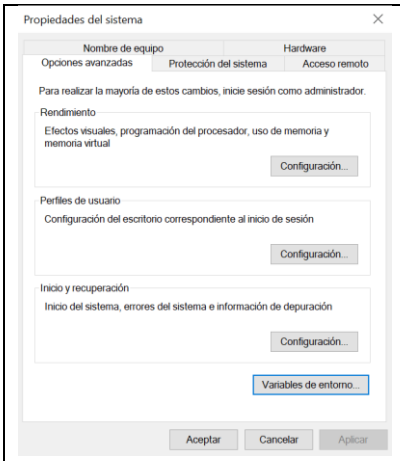
1. Introduction

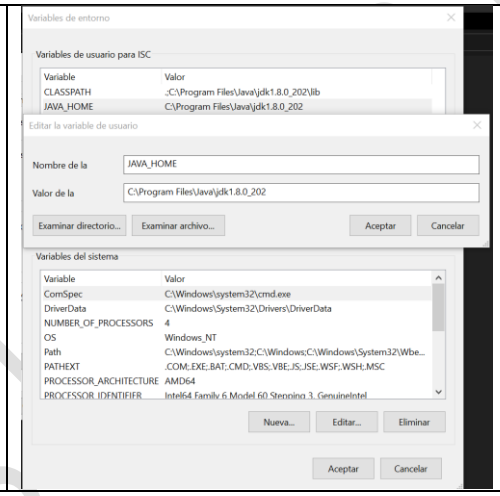
1.1. Technological Requirements:

Java JDK https://www.oracle.com/java/technologies/downloads/ jdk-8u202-windows-x64.exe
VS Code https://code.visualstudio.com/Download VSCodeSetup-x64-1.103.2.exe
Extensions: **Extension Pack for Java**

Integrated Development Environment (IDE)
Eclipse IDE: https://www.eclipse.org/downloads/packages/
NetBeans IDE https://netbeans.apache.org/download/index.html

Variables de entorno





Panel de control -> Sistema -> Configuracion avanzada del sistema
Opciones avanzadas -> Variables de entorno -> Variables de Usuario

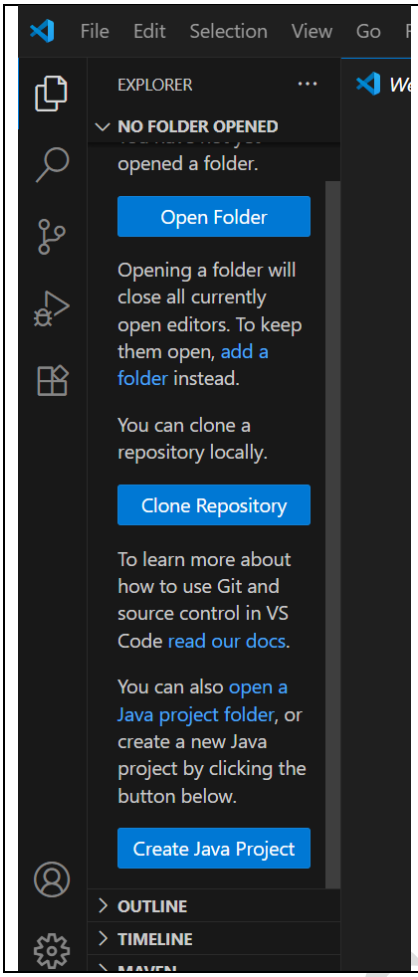
JAVA_HOME C:\Program Files\Java\jdk1.8.0_202	PATH %JAVA_HOME%\BIN
CLASSPATH **.;** %JAVA_HOME%\LIB	Probar Instalación desde CMD C:\>java -version (correr) C:\>javac -version (compilar)

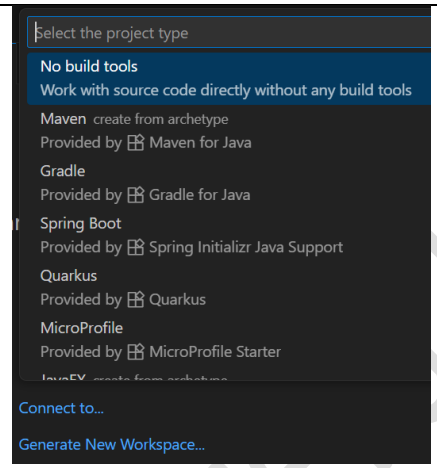
C:\dev>java -version
java version "1.8.0_202"
 	
C:\dev>javac -version
javac 1.8.0_202
 	
C:\dev\poo>javac Hola.java
 	
C:\dev\poo>java Hola
Hello World!
	`public class Hola {`
	
	`  public static void main(String[] args) {`
	`  System.out.println("Hello World!");`
	`  }`
	`}`

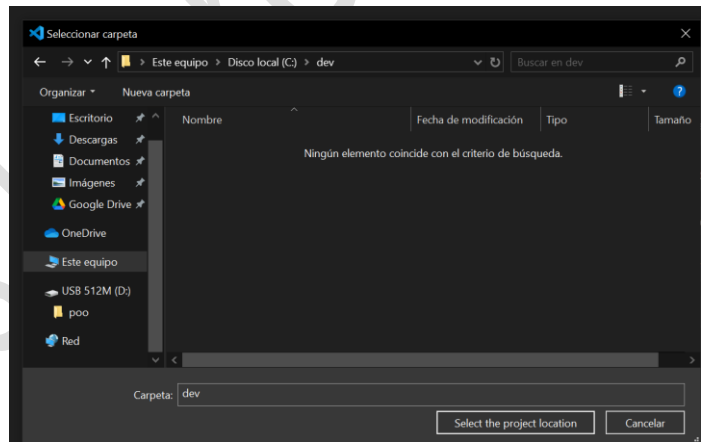
→

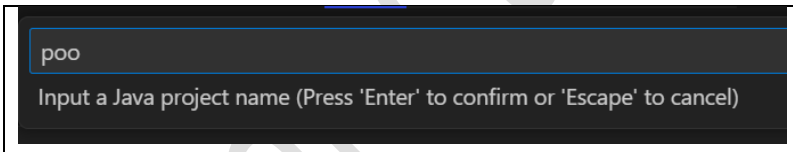
Crear un Directorio de trabajo
C:\> mkdir dev
C:\dev>

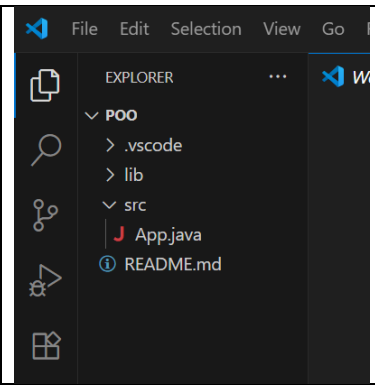
1.2. Create Java Project:

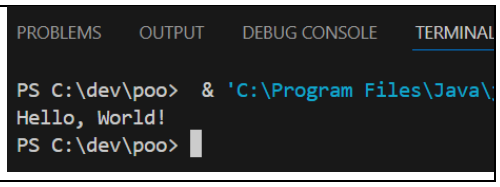












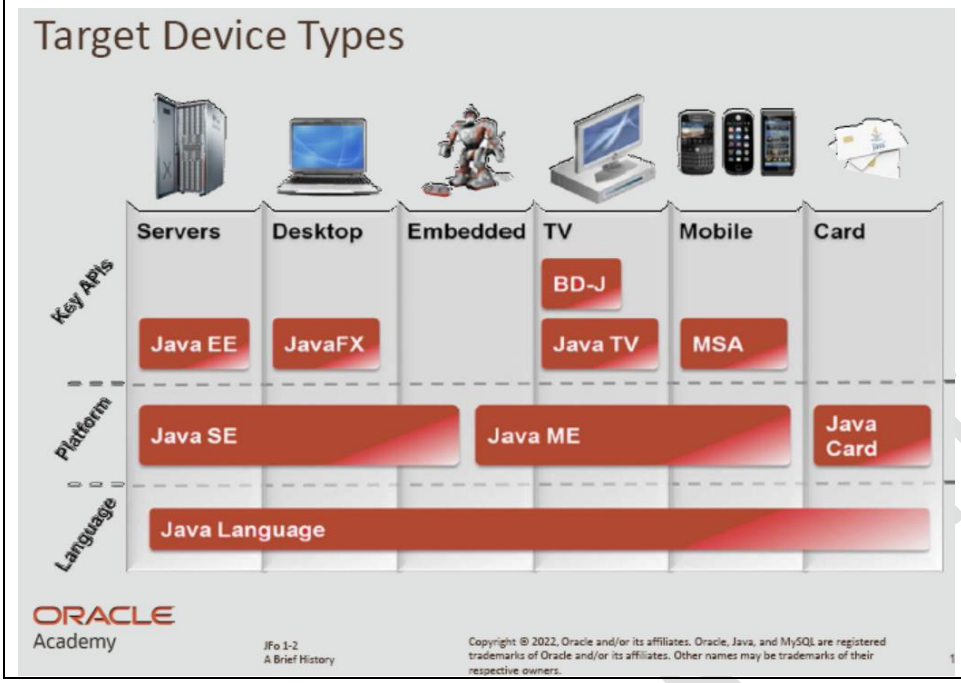
→

1.3. Setting Up Java

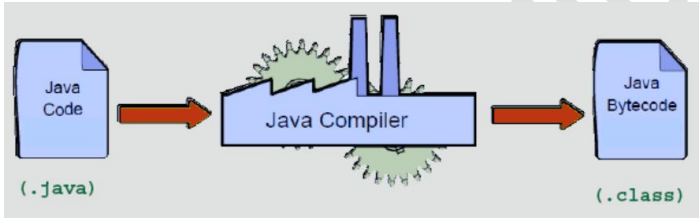
James Gosling is considered the “Father of Java”. Duke, the Java Mascot.

Oracle acquired Sun Microsystems in 2010, and released JDK 7 in 2011, and JDK 8 in 2014.

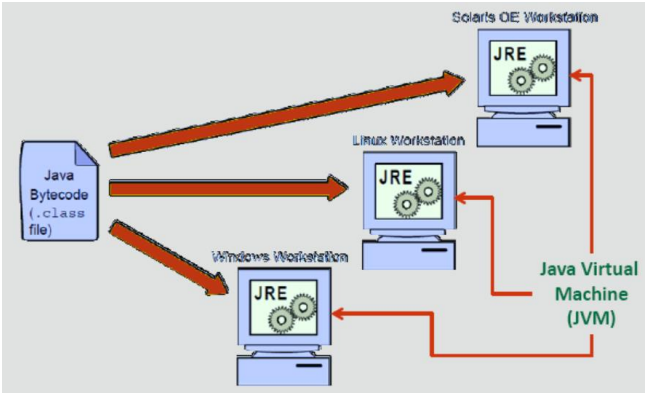
Jakarta EE Is used to create large enterprise, server-side, and client-side distributed applications



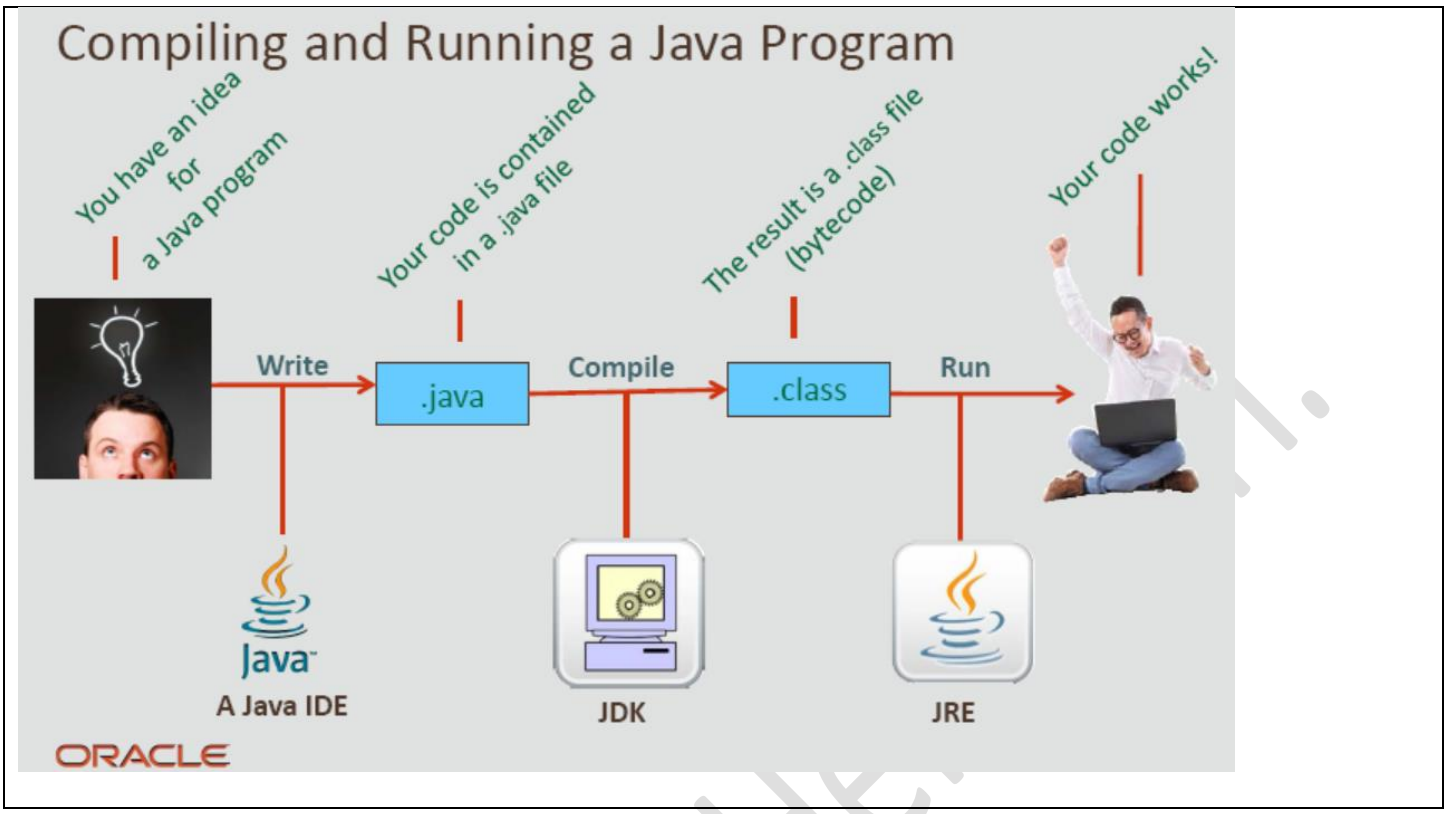
Java is Platform-Independent

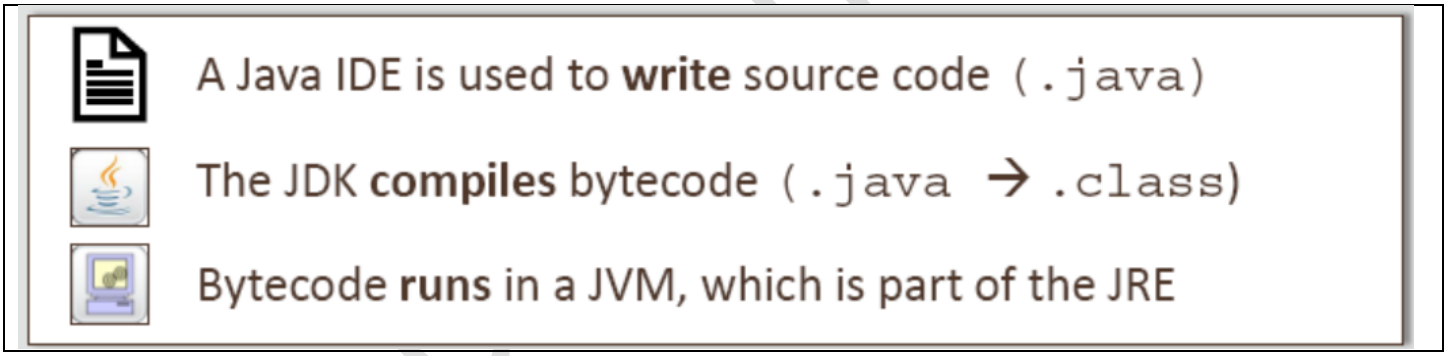


Java Programs Run in a JVM



Java Runtime Environment (JRE)	Java Development Kit (JDK)
Includes: • The Java Virtual Machine (JVM) • Java class libraries Purpose: • Read bytecode (.class) • Run the same bytecode anywhere with a JVM	Includes: • JRE Java Compiler • Additional tools Purpose: Compile bytecode (.java ⍰.class)


Compiling and Running a Java Program
You have an idea for a Java program
Your code is contained in a .java file
The result is a .class file (bytecode)
Your code works!
Write
.java
Compile
.class
Run
Java
A Java IDE
JDK
JRE
ORACLE

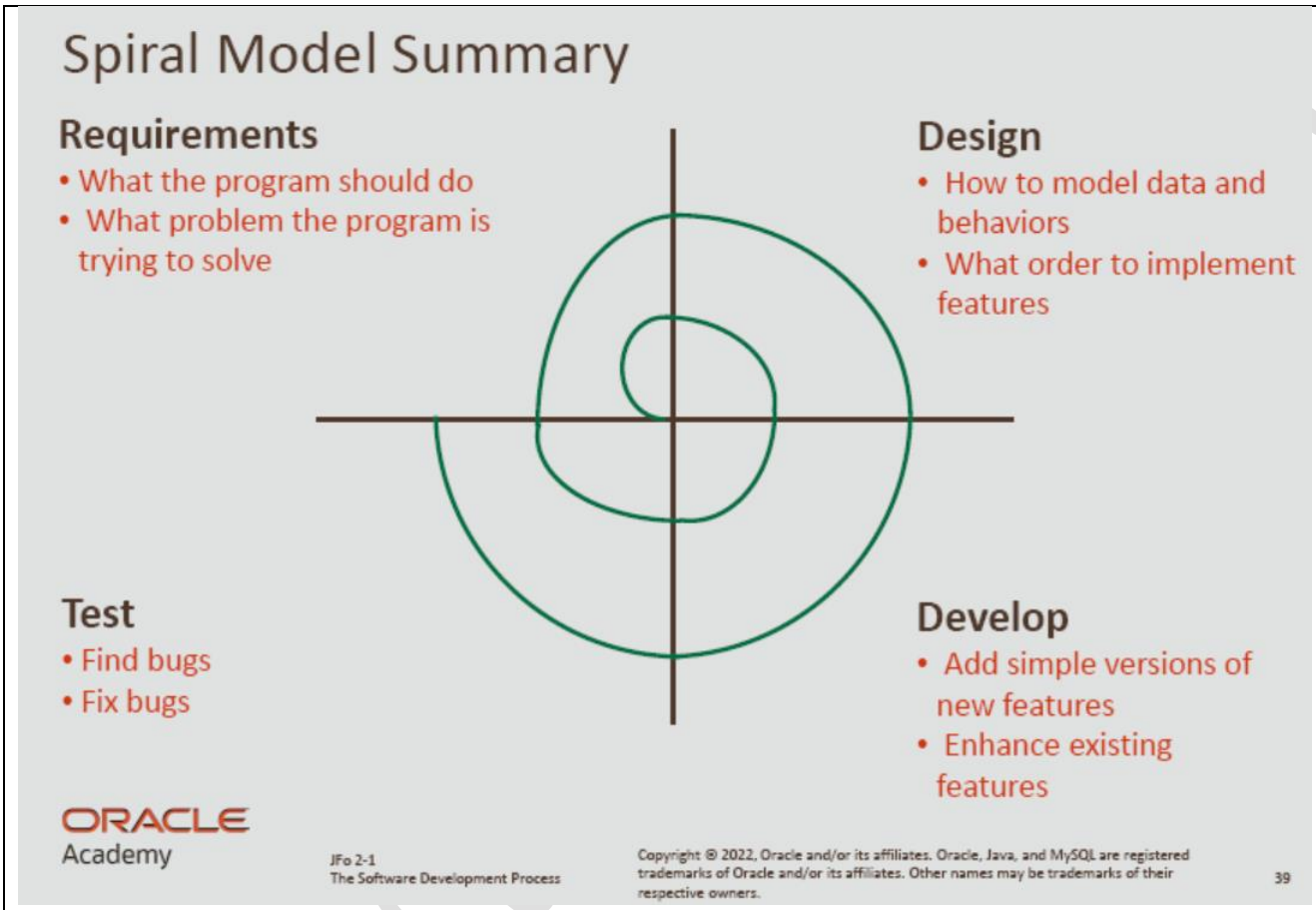

A Java IDE is used to **write** source code (.java)
The JDK **compiles** bytecode (.java → .class)
Bytecode **runs** in a JVM, which is part of the JRE

→

2. Java Basics

2.1. The Software Development Process

Spiral Model of Development



https://objectstorage.uk-london-1.oraclecloud.com/n/lrvrlgaqj8dd/b/Games/o/JavaPuzzleBall/index.html

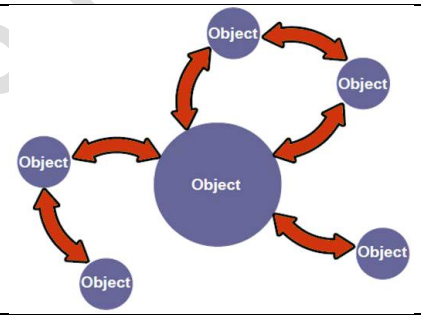
→

2.2. What is my Program Doing?

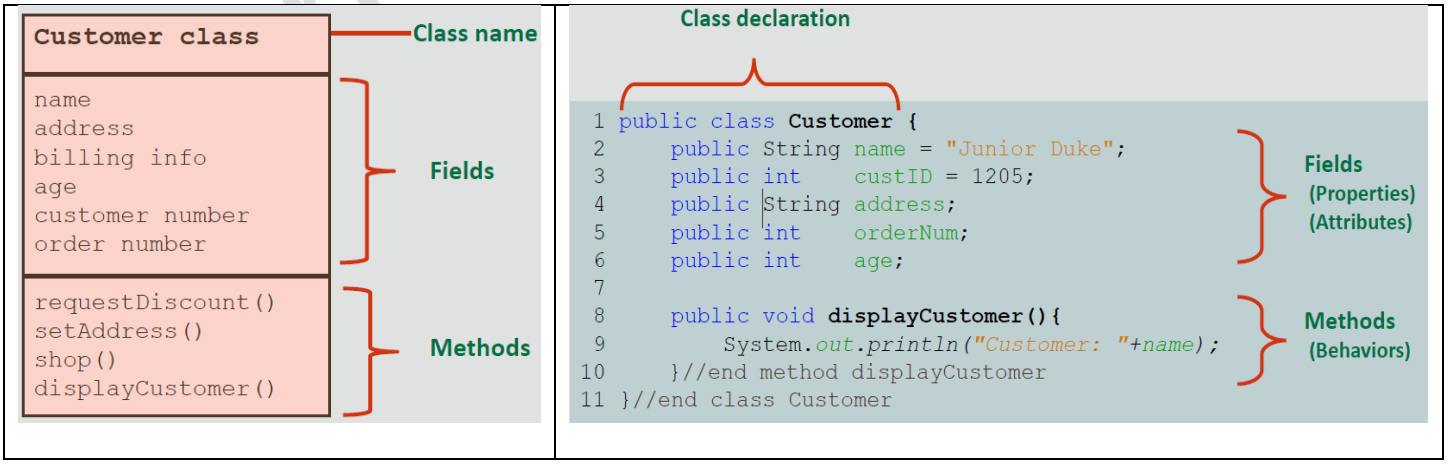
Code within curly braces is called a block of code Indentation before a line of code (4 spaces) Whitespace End statements with semicolons (;) // Single-line comments Multi-line comments /* Bievenidos a poo */	public static void NombreMetodo() { . . } NombreMetodo(); // llamar al método Debug To set a breakpoint Press Step Over

2.3. Introduction to Object-Oriented Programming Concepts

Procedural languages … • Read one line at a time • The C language is procedural	Object-oriented languages… • Read one line at a time • Model objects through code • Emphasize object interaction • Allow interaction without a prescribed order • Java and C++ are object-oriented languages

Object-Oriented Programming • Interaction of objects • No prescribed sequence	

Modeling Properties and Behaviors



→

3. Java Data Types

3.1. What is a Variable?

```
String x ="Sam";
System.out.println("My name is " + x);
```

Variables03.java (There are 6 mistakes)

Type	Keyword	Example Values
Boolean	boolean	true, false
Integer	int	1, -10, 20000, 123_456_789
Double	double	1.0, -10.0005, 3.141
String	String	"Alex", "I ate too much dinner."

Variable Naming Conventions

- Begin each variable with a lowercase letter
- Subsequent words should be capitalized: myVariable
- Choose names that are mnemonic and that indicate the intent of the variable to the casual observer
- Remember that …
- Names are case-sensitive
- Names can't include white space

```
Int studentAge = 20;
String myCatchPhrase = "Enjoy Alex Appreciation Day!";
```

3.2. Numeric Data

Integral Primitive Types

Type	Length	Number of Possible Values	Minimum Value	Maximum Value
Byte	8 bits	2^8, or… 256	-2^7, or… −128	2^7-1, or… 127
short	16 bits	2^{16}, or… 65,535	-2^{15}, or… −32,768	$2^{15}-1$, or… 32,767
int	32 bits	2^{32},or… 4,294,967,296	-2^{31}, or… −2,147,483,648	$2^{31}-1$, or… 2,147,483,647
long	64 bits	2^{64}, or… 18,446,744,073,709,551,616	-2^{63}, or… −9,223,372,036, 854,775,808L	$2^{63}-1$, or… 9,223,372,036, 854,775,807L

+= -= *= /= %= ++ -- Pre/Post a+=b a = a + (b)

// pre y post incremento y decremento

```
int players = 0;
System.out.println("players online: " + players++);
System.out.println("The value of players is " + players);
System.out.println("The value of players is now " + ++players);
System.out.println("The value of players is " + players);
```

Floating Point Primitive Types

Type	Float Length	When will I use this?
`float`	32 bits	Never
`double`	64 bits	Often

double x = 9/2; double x = 9/2.0;

final double PI = 3.141592;

Rules of Precedence

- Operators within a pair of parentheses
- Increment and decrement operators (++or --)
- Multiplication and division operators, evaluated from left to right
- Addition and subtraction operators, evaluated from left to right
- If operators of the same precedence appear successively, the operators are evaluated from left to right

```
int x = (((25 - 5) * 4) / (2 - 10)) + 4;
int y = 25 - 5 * 4 / 2 - 10 + 4;
```

→

3.3. Textual Data

Use the char data type
Use Strings
Concatenate Strings
Understand escape sequences
Understand print statements better

Char is used for a single character (16 bits) char shirtSize= 'M';	A String can handle multiple characters String greeting = "Hello World!";

Primitives

Type	Length	Data
boolean	1 bit	true / false
byte	8 bits	Integers
short	16 bits	Integers
int	32 bits	Integers
long	64 bits	Integers
float	32 bits	Floating point numbers
double	64 bits	Floating point numbers
char	16 bits	Single characters

Where are Strings?

String is capitalized

- Strings are an object, not a primitive
- Object types are capitalized by convention

Combining multiple Strings is called concatenation
String totalPrice = "Total: $" +3 +2 +1;
String totalPrice = 3 +2 + 1 + "Total: $";
String totalPrice = "Total: $" +(3 +2 +1);

Escape Sequence

Escape Sequence	Description
\t	Insert a new tab
\b	Insert a backspace
\n	Insert a new line
\r	Insert a carriage return
\f	Insert a formfeed
\'	Insert a single quote character
\"	Insert a double quote character
\\	Insert a backslash character

System.out.print**ln**("The cat said \"Meow!\" to me.");
println() vs. print()

System.out.println("1\t2\t3\t\"Hola\" mundo");
1 2 3 "Hola" mundo

System.out.println("Hola\nAdios");
Hola
Adios

```
System.out.println("This is the first line."
                + "This is NOT the second line.");
```

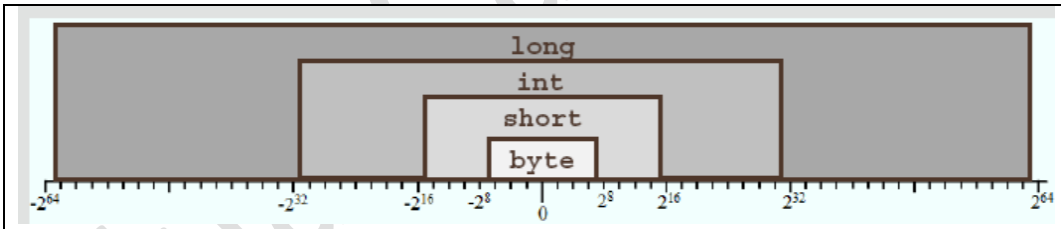
sout tab "Metodo abreviado"

3.4. Converting Between Data Types

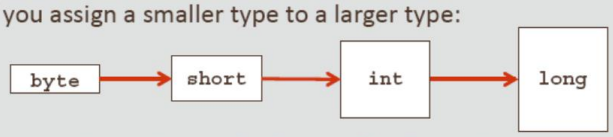
```
double x = 9 / 2; // Should be 4.5
System.out.println(x); // prints 4.0

double y = 4;
System.out.println(y); //prints 4.0
```

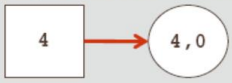
```
int    num1 = 7;
double num2 = 2;
double num3;
num3 = num1 / num2; // num3 is 3.5
```



- Automatic promotions:
  - If you assign a smaller type to a larger type:

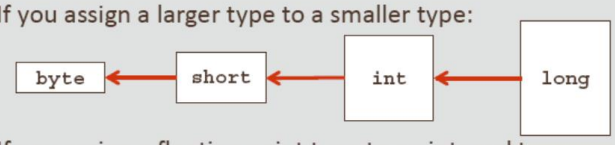


  - If you assign an integral value to a floating-point type:

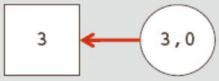


- Examples of automatic promotions:
  - `long intToLong = 6;`
  - `double intToDouble = 4;`

- When to cast:
  - If you assign a larger type to a smaller type:



  - If you assign a floating point type to an integral type:



- Examples of casting:
  - `int longToInt = (int)20L;`
  - `short doubleToShort = (short)3.0;`

double pi = 3.1416
int entero = (int) pi

127 in binary is 01111111; 128 in binary is 10000000.
Java uses the first bit in a number to indicates sign (+/-)

byte, short, and char values are automatically promoted to int prior to an operation

- Solution using larger data type:

```
int num1 = 53;
int num2 = 47;
int num3;
num3 = (num1 + num2);
```

Changed from byte to int

- Solution using casting:

```
int num1 = 53;            // 32 bits of memory to hold the value
int num2 = 47;            // 32 bits of memory to hold the value
byte num3;                // 8 bits of memory reserved
num3 = (byte)(num1 + num2); // no data loss
```

Automatic Promotion

- Example of a potential problem:

```
short a, b, c;
a = 1 ;
b = 2 ;
c = a + b ; //compiler error
```

a and b are automatically promoted to integers

- Example of potential solutions:
  - Declare c as an `int` type in the original declaration:
    - `int c;`
  - Type cast the (a+b) result in the assignment line:
    - `c = (short)(a+b);`

int x = 123_456_789;
int x = 123456789;

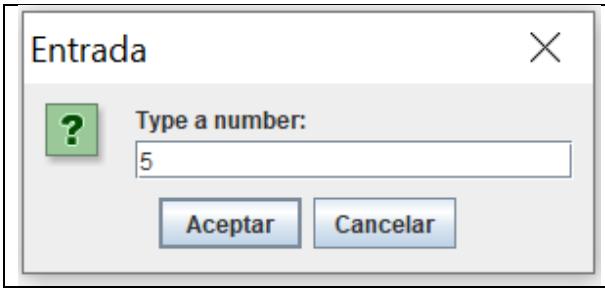
intintVar1 = Integer.parseInt("100");
doubledoubleVar2 = Double.parseDouble("2.72");

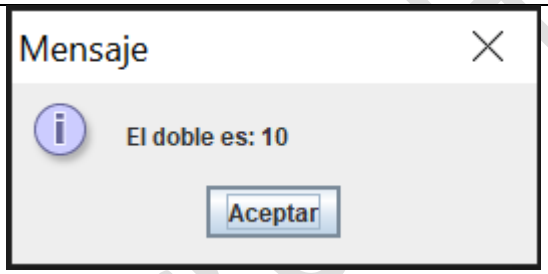
→

3.5. Keyboard Input

```
System.out.println("\033[H\033[2J"); // limpiar pantalla

String input = JOptionPane.showInputDialog(null, "Type a number:");
int number = Integer.parseInt(input);
number *= 2;
JOptionPane.showMessageDialog(null, "El doble es: " + number);
```





The Scanner searches for tokens

A few useful Scanner methods …

- nextInt() reads the next token as an int
- nextDouble() reads the next token as a double
- next() reads the next token as a String

```
Scanner sc = new Scanner(System.in);
```

Reading from a File

- nextLine() advances this Scanner past the current line and returns the input that was skipped
- findInLine("StringToFind") Attempts to find the next occurrence of a pattern constructed from the specified String, ignoring delimiters

```
Scanner sc = new Scanner(MyClase.class.getResourceAsStream("texto.txt"));
```

```
Scanner sc = new Scanner(System.in);
int  x = sc.nextInt();
double y = sc.nextDouble();
String z = sc.next();
String linea = sc.nextLine();
int numero = Integer.parseInt(z);
sc.close();
```

→

4. Java Methods and Library Classes
5. Decision Statements
6. Loop Constructs
7. Creating Classes
8. Arrays and Exceptions
9. JavaFX