

Java Básico

```
public class HolaMundo {  
    public static void main(String[] args) {  
        System.out.println("Hola Mundo");  
    }  
}
```



Anahí Salgado
@anncode

¿Java?

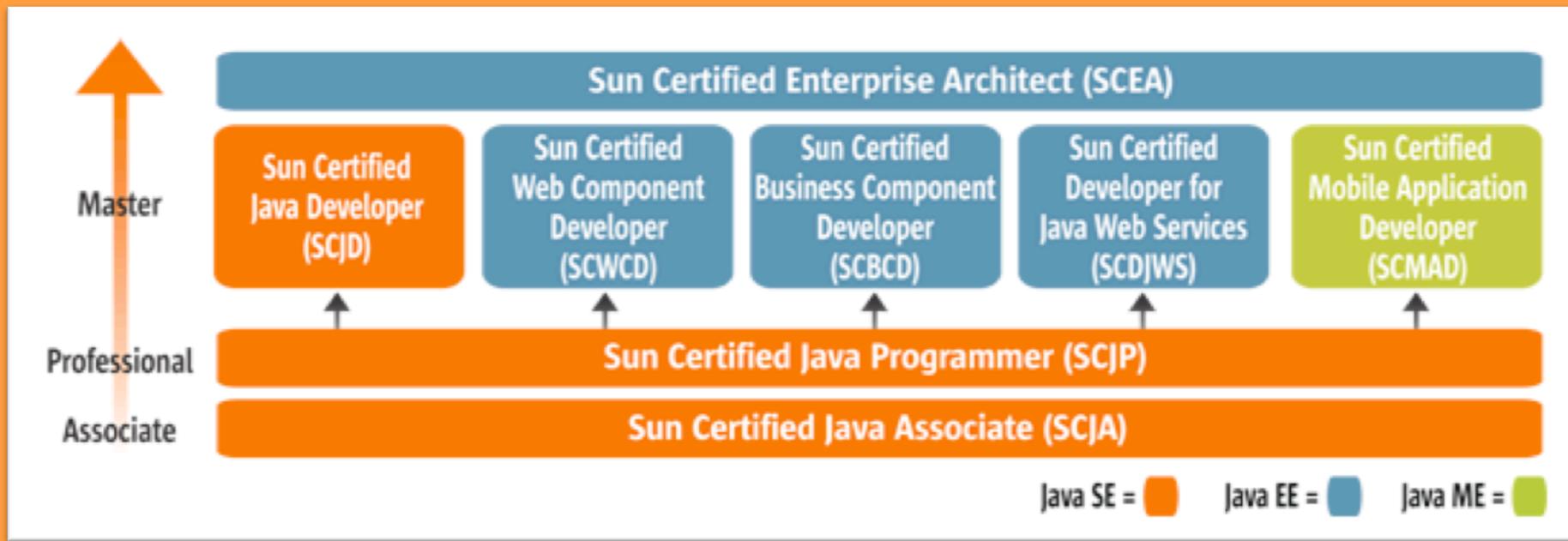
Anahí Salgado
@anncode



- Empezar rápidamente
 - C y C++
- Escribir menos código
 - POO – Reutilización
- Escribir mejor código
 - Buenas prácticas de codificación
- Desarrollar con mayor rapidez
 - Más simple que C++



- Lenguaje más utilizado a nivel mundial
- Google
- Amazon
- Empresas financieras
 - Incrementa tus posibilidades de conseguir trabajo
 - Aumentará tus aspiraciones profesionales



Plan de Certificaciones

Anahí Salgado
@anncode



Java™

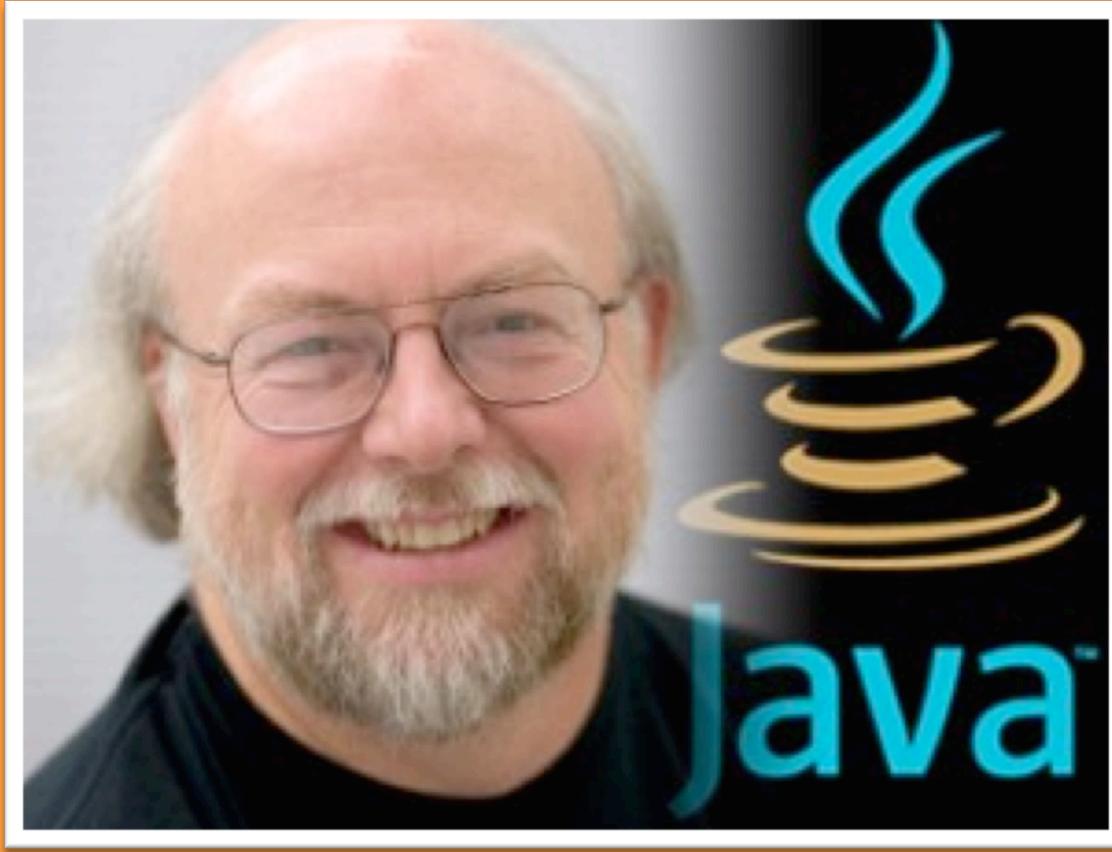
Anahí Salgado
@anncode

1991

Anahí Salgado
@anncode

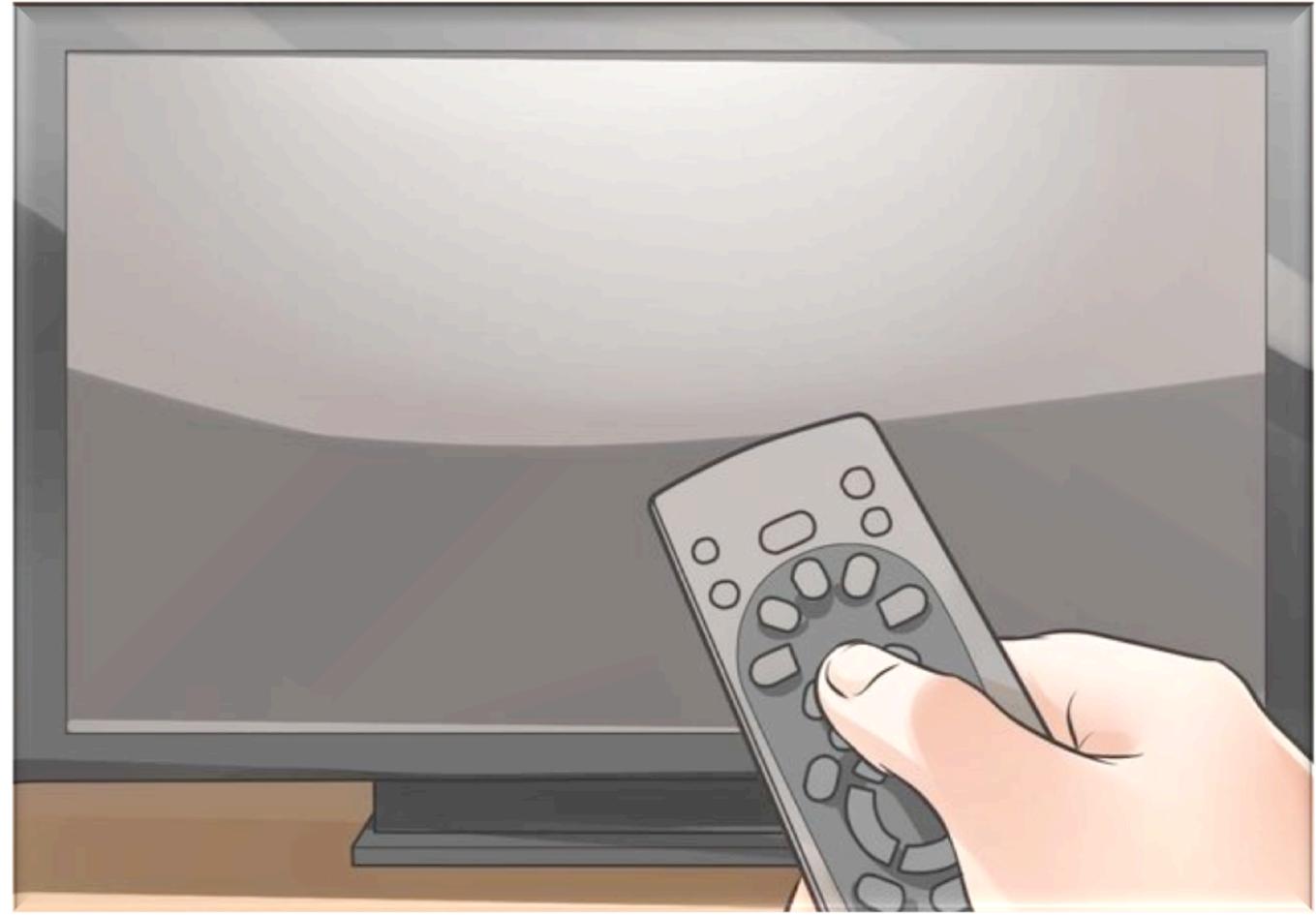


Anahí Salgado
@anncode



James Goslin

Anahí Salgado
@anncode



Comunicación entre dispositivos

Anahí Salgado

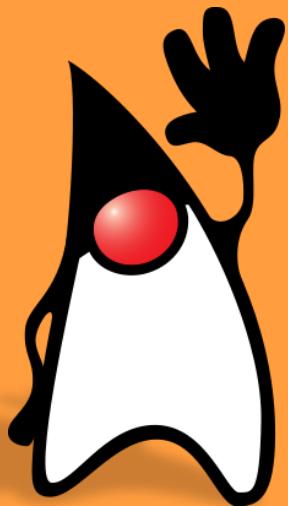
@anncode

2009

Anahí Salgado
@anncode

ORACLE®

Anahí Salgado
@anncode



- Java es un lenguaje de programación de **alto nivel**
 - Simple
 - Orientado a Objetos
 - Distribuido
 - Multihilo
 - Arquitectura Neutral
 - Portable
 - Alto desempeño
 - Seguro

Filosofía



Write Once
Run Anywhere

Anahí Salgado
@anncode



¿Qué versión elegir?

Java EE

Java SE

Java ME



Java SE

Java Standard Edition

Anahí Salgado
@anncode



Componentes

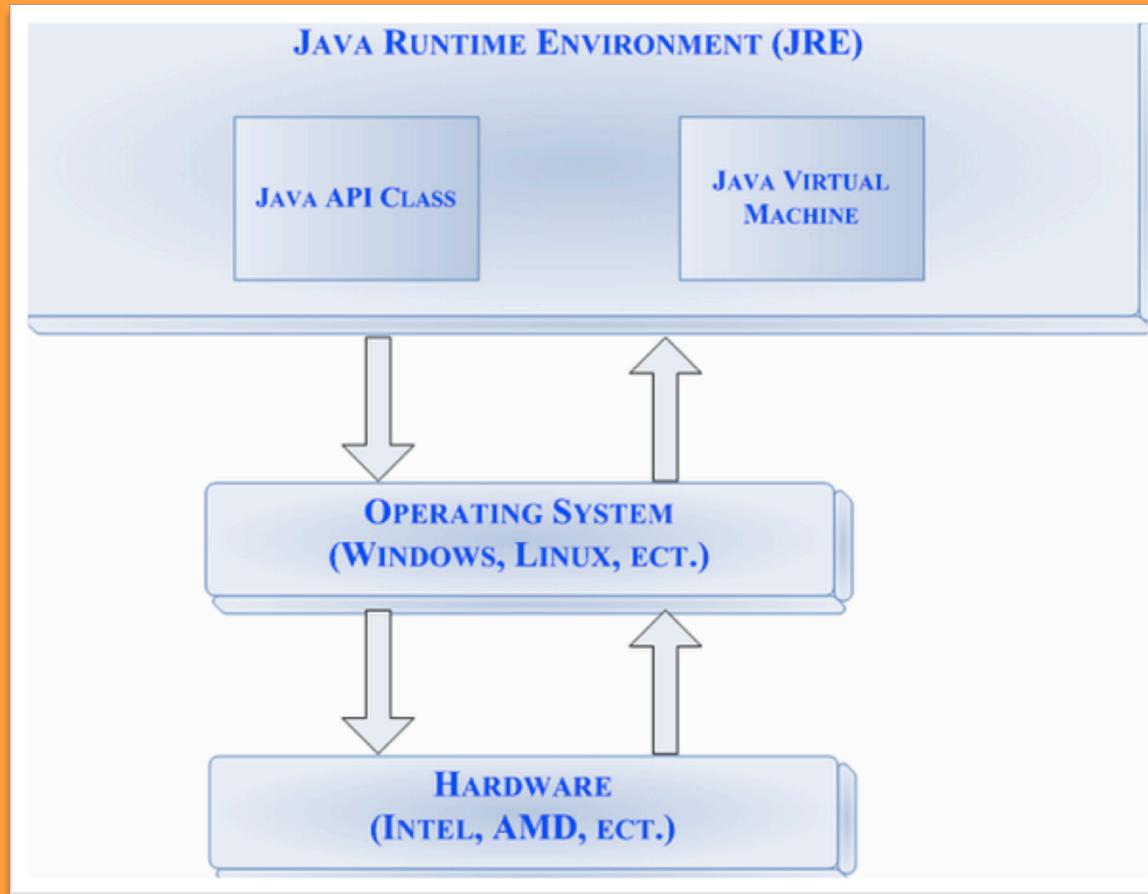
JDK
Java
Development Kit



JRE
Java Runtime
Environment



Java Virtual Machine



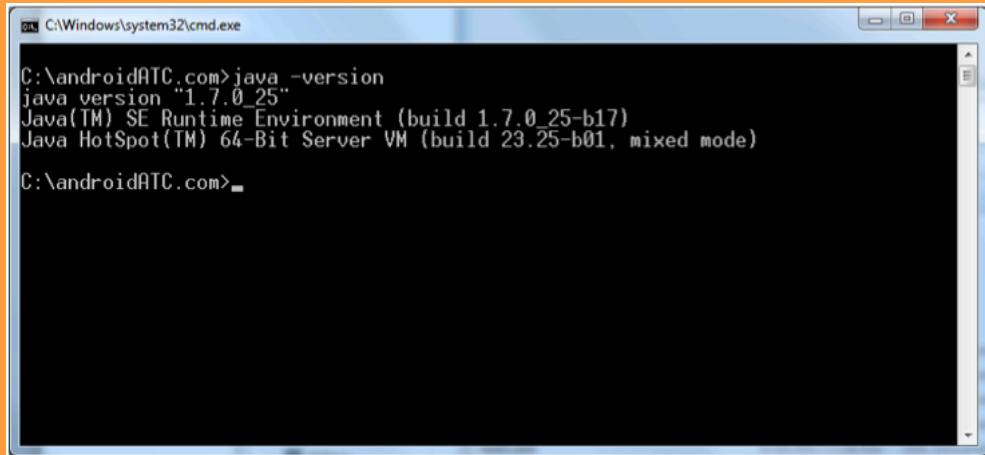
Anahí Salgado
@anncode



Programando con Java

Verifica que lo tengas instalado y configurado

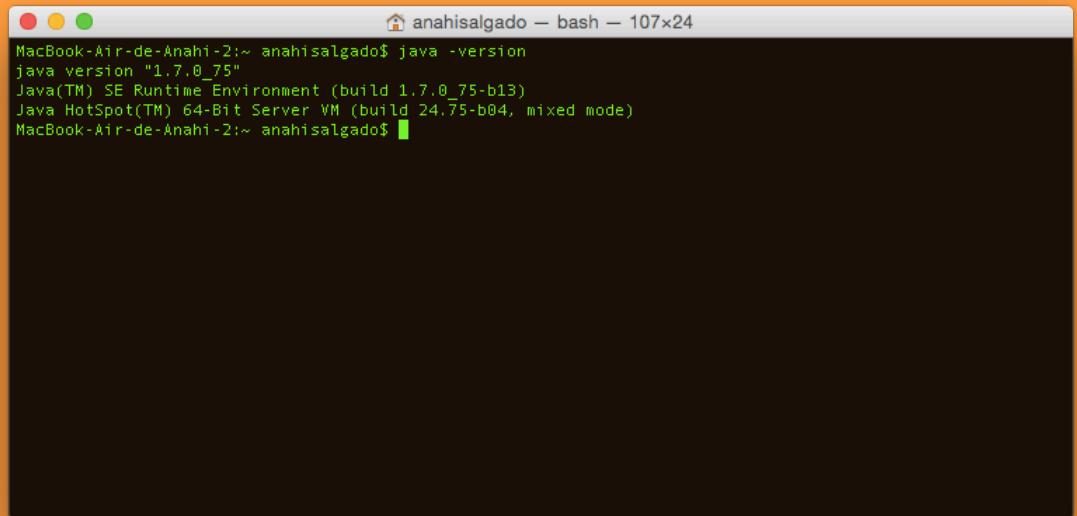
```
$ java -version  
$ javac
```



A screenshot of a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The command 'java -version' is entered, and the output shows Java version 1.7.0_25, Java(TM) SE Runtime Environment (build 1.7.0_25-b17), and Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode). The prompt 'C:\androidATC.com>' is visible at the bottom.

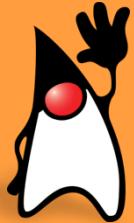
```
C:\androidATC.com>java -version
java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

C:\androidATC.com>
```



A screenshot of a Mac OS X Terminal window titled 'anahisalgado — bash — 107x24'. The command 'java -version' is entered, and the output shows Java version 1.7.0_75, Java(TM) SE Runtime Environment (build 1.7.0_75-b13), and Java HotSpot(TM) 64-Bit Server VM (build 24.75-b04, mixed mode). The prompt 'MacBook-Air-de-Anahi-2:~ anahisalgado\$' is visible at the bottom.

```
MacBook-Air-de-Anahi-2:~ anahisalgado$ java -version
java version "1.7.0_75"
Java(TM) SE Runtime Environment (build 1.7.0_75-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.75-b04, mixed mode)
MacBook-Air-de-Anahi-2:~ anahisalgado$
```



Instalando JDK

Sign In/Register Help Country ▾ Communities ▾ I am a... ▾ I want to... ▾ Search C

Products Solutions Downloads Store Support Training Partners

Oracle Technology Network > Java > Java SE > Downloads

Java SE Downloads

Next Releases (Early Access) Embedded Use Previous Releases

 DOWNLOAD ↴  DOWNLOAD ↴

Java Platform (JDK) 7u51 JDK 7u51 & NetBeans 7.4

Java Platform Standard Edition

Java SE Downloads

Overview Downloads Documentation Community Technologies Training

Java SE

Java EE

Java ME

Java SE Support

Java SE Advanced & Suite

Java Embedded

JavaFX

Java DB

Web Tier

Java Card

Java TV

New to Java

Community

Java Magazine

Java SE Development Kit 7u51

You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux ARM v6/v7 Hard Float ABI	67.7 MB	jdk-7u51-linux-arm-vfp-hflt.tar.gz
Linux ARM v6/v7 Soft Float ABI	67.68 MB	jdk-7u51-linux-arm-vfp-sflt.tar.gz
Linux x86	115.65 MB	jdk-7u51-linux-i586.rpm
Linux x86	132.98 MB	jdk-7u51-linux-i586.tar.gz
Linux x64	116.96 MB	jdk-7u51-linux-x64.rpm
Linux x64	131.8 MB	jdk-7u51-linux-x64.tar.gz
Mac OS X x64	179.49 MB	jdk-7u51-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.02 MB	jdk-7u51-solaris-i586.tar.Z
Solaris x86	95.13 MB	jdk-7u51-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.53 MB	jdk-7u51-solaris-x64.tar.Z
Solaris x64	16.28 MB	jdk-7u51-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	139.39 MB	jdk-7u51-solaris-sparc.tar.Z
Solaris SPARC	98.19 MB	jdk-7u51-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	23.94 MB	jdk-7u51-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.33 MB	jdk-7u51-solaris-sparcv9.tar.gz
Windows x86	123.64 MB	jdk-7u51-windows-i586.exe
Windows x64	125.46 MB	jdk-7u51-windows-x64.exe

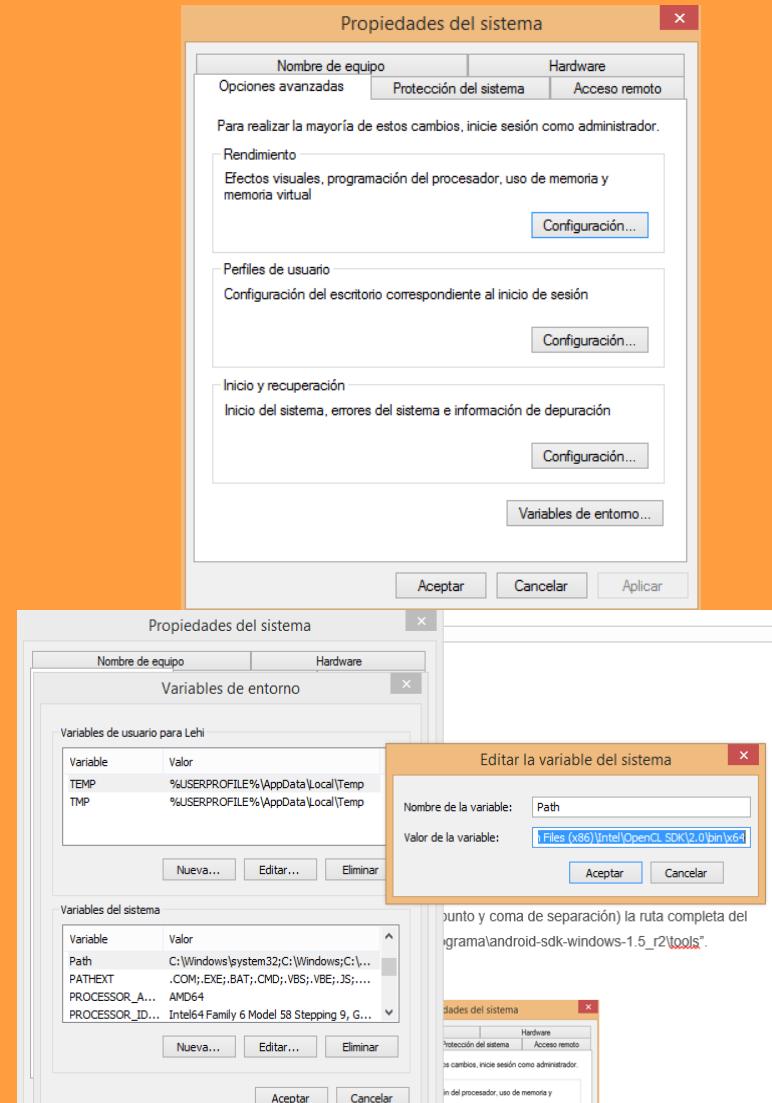
Instalando JDK

Variables de entorno



Vamos a Mi Pc -> Propiedades,

- Pestaña Opciones avanzadas, seleccionamos Variables de entorno -> Path
- Hacemos click en Modificar y añadimos la ruta completa del directorio
- “C:\Program Files\Java\jdk1.7.0_51\bin”.



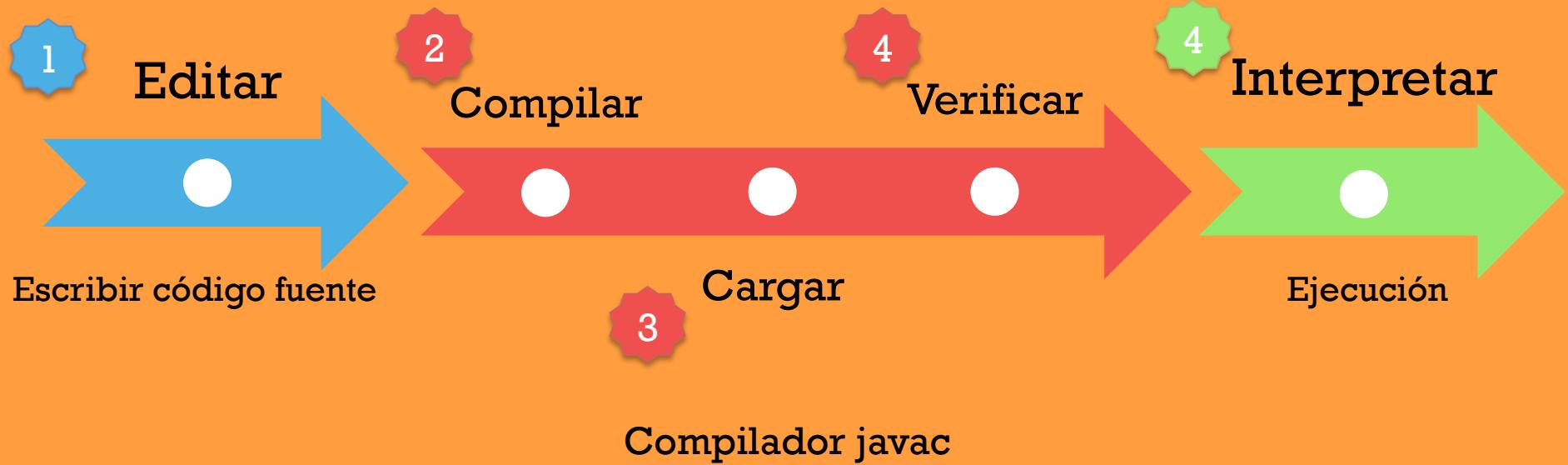


Hola Mundo

- Editor de Textos
- Consola de comandos (CMD)

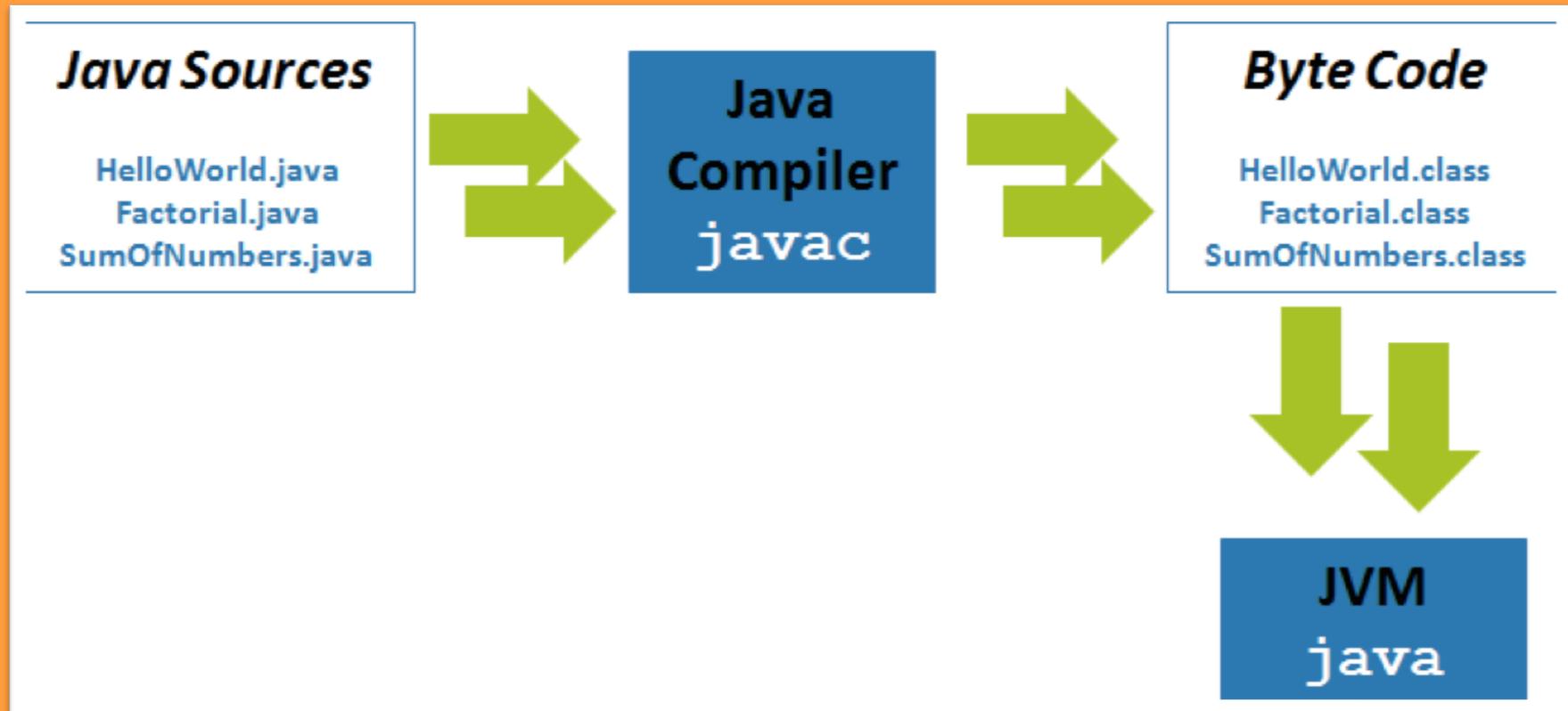


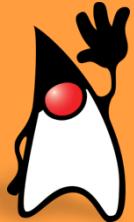
Fases de programación





Fases de programación





El método Main

Método que es el punto de entrada de una aplicación Java.



El método Main

- Declara todas las acciones realizadas por tu aplicación

```
public static void main (String[ ] args) {  
    // acciones  
}
```

- Sin él, la aplicación no se ejecutará, regresando el siguiente error: **In class NoMain: void main(String args[])** no está definido.



El método Main

- El método **main** contiene dos modificadores requeridos, **public** y **static**.
- No devuelve ningún valor, por lo que tiene un tipo de retorno de vacío.
- El método principal tiene un identificador método (nombre) de "main".
- Acepta cero o más objetos de tipo String (String [] args). Esta sintaxis le permite escribir en los valores de la línea de comandos para ser utilizado por el programa mientras se está ejecutando.

Anahí Salgado

@anncode

Hola Mundo



- IDE (Integrated Development Environment):
- Es un entorno de programación que ha sido empaquetado como un programa de aplicación.
 - Editor de código
 - Compilador
 - Depurador
 - Constructor de interfaz Gráfica

Usando un IDE (Eclipse)



Tipos de Datos

Anahí Salgado
@anncode



Tipos de Datos

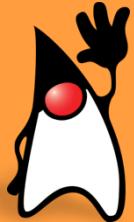
- DEFINICIÓN:
 - Un espacio de memoria al que le asignamos un contenido, puede ser un valor numérico, de tipo carácter o cadena de caracteres.
- Por ejemplo:
 - $a = 8$
 - $a = 56$
 - $a = b$



Tipos de Datos

Tipo Primitivo

Tipo Objeto



byte

Rango
-128 a 127

1
byte

int

Rango
-2,147,483,648 to
2,147,483,647

short

Rango
-32,768 a 32,7676

2
bytes

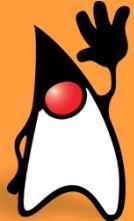
long

Rango
-9,223,372,036,85
4,775,808
to
+9,223,372,036,85
4,775,807

4
bytes

8
bytes

Tipos ENTEROS



float

Rango

1.40129846432481707e-45

to

3.40282346638528860e+38

4
byte

double

Rango

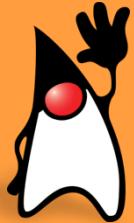
4.94065645841246544e-324d

to

1.79769313486231570e+308d

8
bytes

Tipos PUNTO FLOTANTE



char

Rango
Unicode

2
byte

Tipos TEXTO

Anahí Salgado
@anncode



boolean

Rango
true o false

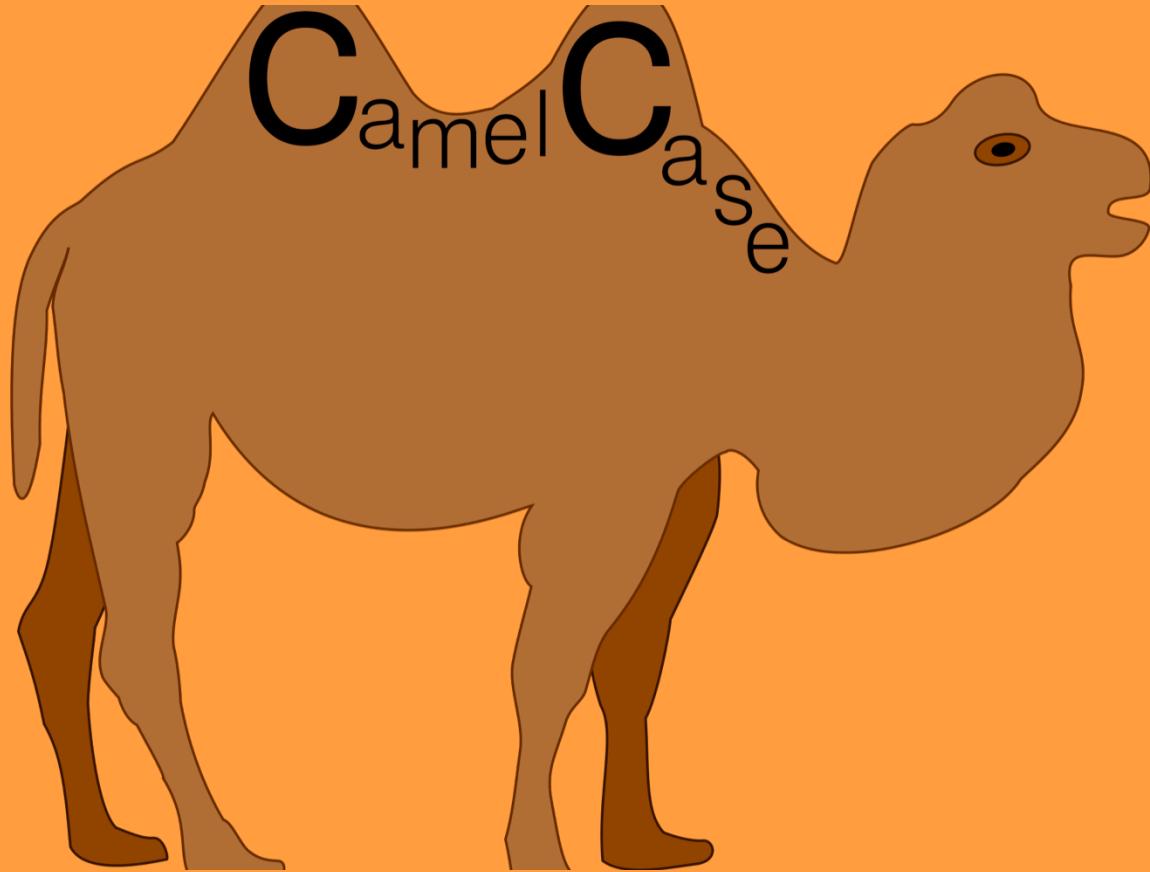
1
bit

Tipos LOGICOS



Nombres en Java

- Java sigue la siguiente convención para nombrar variables
 - Es sensible al uso de mayúsculas y minúsculas
 - Debe comenzar con una letra, se permite usar \$ y “_”
 - Las letras posteriores pueden ser letras, números, \$ y “_”
 - Por convención se debe usar la técnica “camello”
 - También por convención, las constantes se escriben en mayúsculas y contienen “_”.



Upper Camel Case
Lower Camel Case

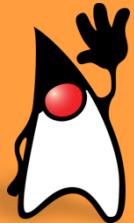
CAST

En la programación hay situaciones donde
se necesita cambiar el tipo de dato



Cast

- Un Cast es una operación en Java que:
 - Da como resultado una variable con un tipo de datos diferente a su fuente.
 - Puede usarse entre tipos de datos primitivos, instancias de una clase y tipos de objetos primitivos.



```
double d = 10;  
  
int i;  
  
i = (int) d
```



Cast



Cast a tipos primitivos

- Se puede realizar el cast para todos los tipos de datos primitivos, con excepción de boolean.
- A menudo, el tipo cast de tipos primitivos se realiza en situaciones donde el tipo del resultado es más grande que su tipo original.



Cast a tipos primitivos

- Por lo tanto, a menudo se puede usar un byte o char como un int, un int como un long, un int como un float y como un doble también.

Arrays

Anahí Salgado
@anncode



Arrays

- Los arreglos se pueden definir como objetos en los que podemos guardar mas de una variable



Arrays

- La estructura de declaración de un arreglo es la siguiente:

```
tipo_dedato[ ] nombre_variable;
```

```
tipo_dedato nombre_variable[ ];
```



Arrays. Definir tamaño

- Para asignar a un arreglo su tamaño o capacidad, se hace de la siguiente forma:

```
array = new tipo_dedato[capacidad];
```



Arrays. Asignar valores

- Una vez se tiene declarado un arreglo, y al mismo se le ha asignado un tamaño o capacidad, podemos accesar a los datos dentro del mismo y asignarle valores.

```
array[indicador] = valor;
```

OPERADORES

Una vez que el código fuente de Java tienen variables, las podemos usar para crear y formar expresiones que regresen valores.



Operadores aritméticos

- Son los símbolos que se usan para realizar aritmética básica en el lenguaje de programación java

Operador	Significado	Ejemplo
+	Adición	$x + y$
-	Substracción	$x - y$
*	Multiplicación	$x * y$
/	División	x / y
%	Módulo	$x \% y$



Concatenación de cadenas

- El operador + puede usarse para agregar o concatenar cadenas
- Unión de dos elementos.

```
System.out.println ("El balance de la cuenta es: " +  
                    balance );
```



Operadores de Asignación

=

+=

-=

/=

%=

x += 2; x = x + 2;

Anahí Salgado

@anncode

Operadores de incremento y decrecimiento



- **Incremento:** Se usan para agregar un 1 al valor de la expresión
++
- **Decremento:** Se usan para substrair un 1 del valor de la expresión.
--



Prefijo y postfijo

- Un prefijo se refiere a colocar un operador antes del operando

`++i`

`--i`

- Un posfijo se refiere a colocar un operador después del operando.

`i++`

`i--`



Incremento

Escribiendo expresiones.

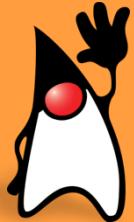
1. Indique cual es la salida de la siguiente clase

```
class EjemploPrePost {  
    public static void main(String[] args){  
        int i = 3;  
        i++;  
        System.out.println(i);  
        ++i;  
        System.out.println(i);  
        System.out.println(++i);  
        System.out.println(i++);  
        System.out.println(i);  
    }  
}
```

Equidad y operadores relacionales



- Todas las expresiones creadas con equidad y operadores relacionales regresaran un valor booleano, dependiendo si la comparación se realiza o no.



Equidad y operadores relacionales

- Hace uso de dos operandos, uno en cada lado del operador.
- Los operadores de equidad se describen a continuación:

Operador	Significado	Ejemplo
<code>==</code>	Igual	<code>a == b</code>
<code>!=</code>	Distinto	<code>a != b</code>



Equidad y operadores relacionales

- Los operadores relacionales se describen a continuación:

Operador	Significado	Ejemplo
<	Menor que	$a < b$
>	Mayor que	$a > b$
\leq	Menor o igual a	$a \leq b$
\geq	Mayor o igual a	$a \geq b$



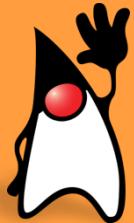
Operadores lógicos

- Combinan expresiones que regresan un valor boolean

AND &&

OR ||

NOT !



Operadores lógicos

A	B	A OR B
F	F	F
F	V	V
V	F	V
V	V	V

F: Falso

V: Verdadero

A	B	A AND B
F	F	F
F	V	F
V	F	F
V	V	V

A	NOT A
F	V
V	F

CONTROL FLUJO

Las sentencias de código en java son ejecutadas secuencialmente desde arriba hasta abajo en el orden en que van apareciendo.

Sin embargo podemos controlar el flujo usando sentencias condicionales, ciclos, etc.

Anahí Salgado

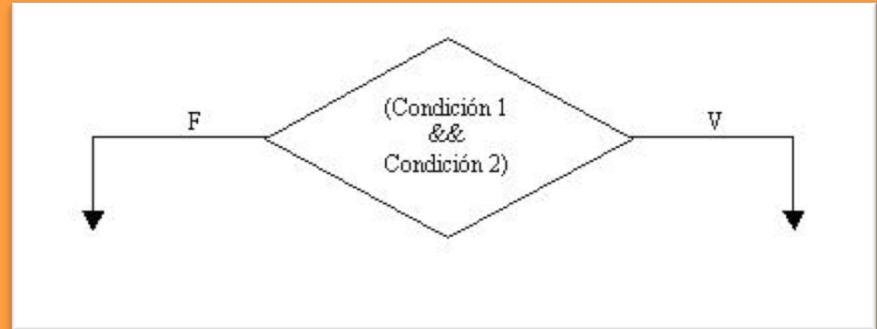
@anncode



If/Else

- Un condicional es una expresión booleana.
- La sentencia se ejecuta solamente si la expresión booleana es verdadera.

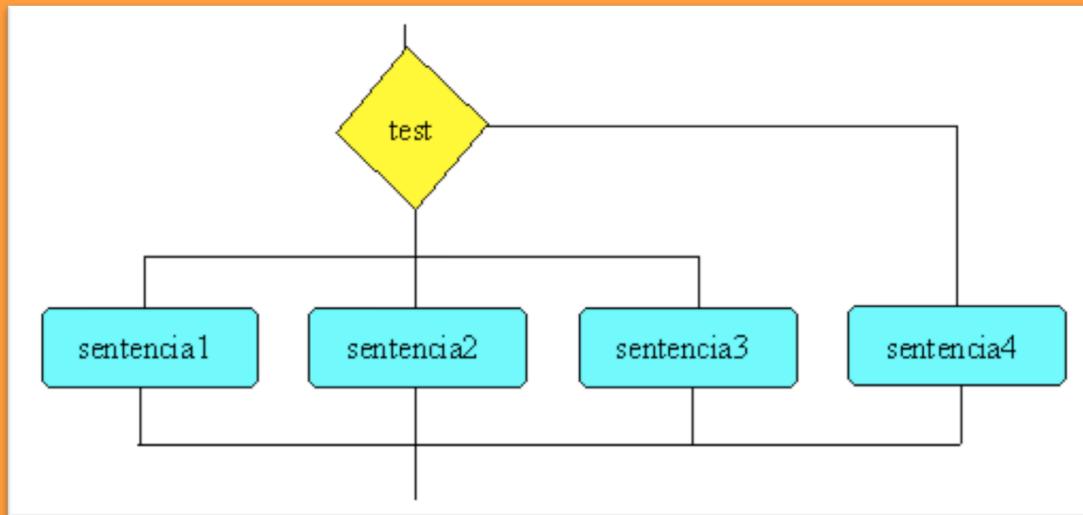
```
if (condición) {  
    instrucciones  
} else {  
    instrucciones  
}
```





Switch

- A diferencia de sentencias if / else, la sentencia switch puede tener un número de posibles rutas de ejecución

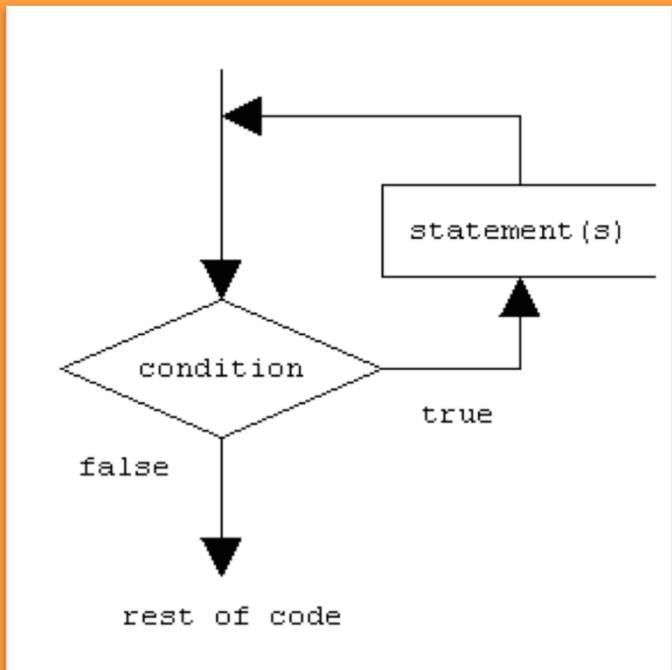




Ciclo While

- Se ejecuta continuamente un bloque de código mientras una condición particular, es cierto. Su sintaxis se puede expresar como:

```
while (condicion) {  
    //instrucciones  
}
```





Ciclo For

- La sentencia proporciona una forma compacta para iterar sobre un rango de valores.

```
for (inicializa; fin-condicion; incremento) {  
    instrucciones  
}
```

Ciclo For extendido foreach



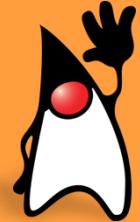
- Es más fácil para recorrer colecciones de datos sin necesidad de conocer o definir el número de elementos a recorrer

```
for ( TipoDato elemento : colección ) {  
    Instrucciones  
}
```

PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

Anahí Salgado
@anncode

Programación Orientada a Objetos

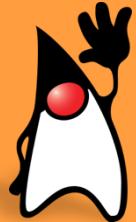


Una nueva forma de pensar



Anahí Salgado
@anncode

Programación Orientada a Objetos



Se trata de descomponer el problema
en subproblemas y más
subproblemas

Programación Orientada a Objetos



Definir un Dominio del Problema
PROBLEM DOMAIN

Recopilación de requisitos del cliente y
tener por escrito un alcance

¿Qué queremos lograr?

Programación Orientada a Objetos



Fijarnos en el
escenario del problema
y tratar de simularlo con objetos

Programación Orientada a Objetos



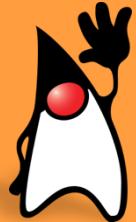
Identificar mis objetos

- Pueden ser **Físicos** o **Conceptuales**
- Los objetos tienen **atributos** (características)
 - tamaño
 - nombre
 - forma
 - representan el estado del objeto
- Los objetos tienen **operaciones** (las cosas que puede hacer el objeto)

Anahí Salgado

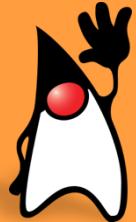
@anncode

Programación Orientada a Objetos



- Los nombres de los objetos por lo general son **sustantivos**
cuenta, cliente
- Los atributos de los objetos también
- Las operaciones suelen ser **verbos** o sustantivo y verbo
mostar, Enviar Pedido

Programación Orientada a Objetos



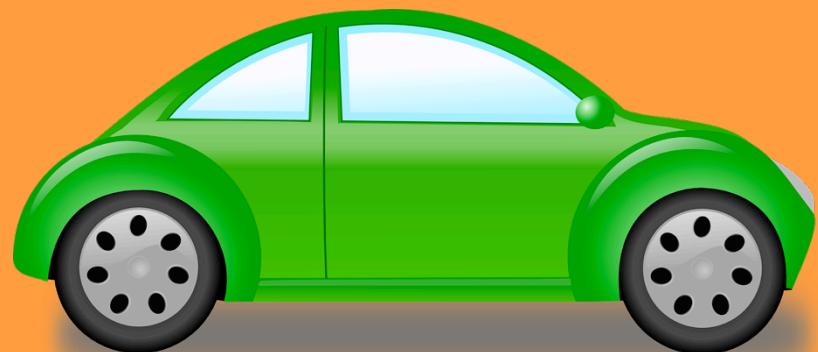
- Vehiculo

atributos:

- matricula
- marca
- modelo
- año

comportamiento:

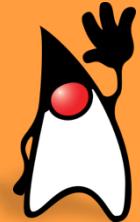
- arrancar
- frenar
- reversa



Anahí Salgado

@anncode

Programación Orientada a Objetos



- Diseñando un **modelo** de Clase
- Una **Clase** es la forma en como defines tu objeto
- Las **Clases** son descriptivas – plantillas

Programación Orientada a Objetos



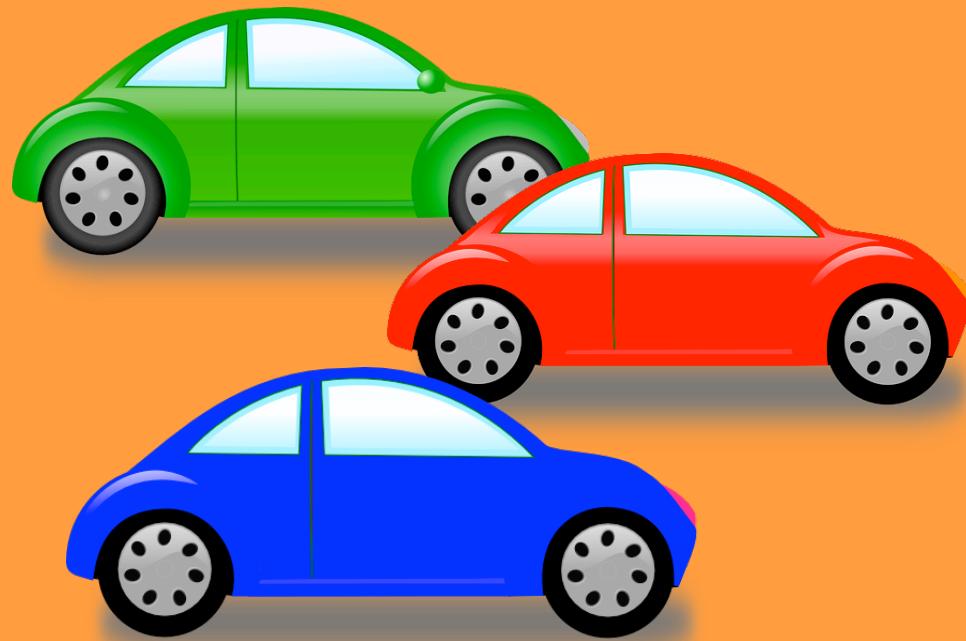
- Clase Vehiculo

atributos:

- matricula
- marca
- modelo
- año

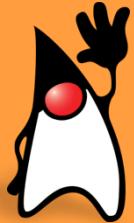
comportamiento:

- arrancar
- frenar
- reversa



Anahí Salgado

@anncode



Tipos Datos Objeto

Byte

Short

Integer

Long

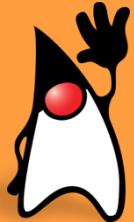
Float

Double

Characer

Boolean

String



Variables ≠ Objetos

- Variables son entidades elementales (muy sencillas)
 - Un número
 - Un carácter
 - Un valor verdadero falso
- Objetos son entidades complejas que pueden estar formadas por la agrupación de muchas variables y métodos.

CODIGO

Anahí Salgado
@anncode



Declaración de métodos

- Una declaración de un método es un elemento de código en Java que:
 - Consiste de cuatro partes: tipo de datos de regreso, nombre, argumentos y cuerpo entre llaves.

mod. acceso

valor regreso

nombre

argumentos

public

int

suma

(int a int b)



Declaración de métodos

- Tiene un valor de regreso explícitamente invocado en su cuerpo usando la palabra reservada return.
- No regresa ningún valor si es declarado void.
- No puede declararse dentro de otro método.

```
public int suma(int a int b){  
    return a+b;  
}
```

Anahí Salgado

@anncode



Constructor

- Un constructor es un conjunto de sentencias que:
 - Crea nuevas instancias de una clase.
 - Tiene el mismo nombre que la clase que inicializa.
 - Usa la palabra reservada new para invocarlo.
 - Usa cero o más argumentos contenidos dentro de los paréntesis que siguen al nombre.
 - No regresa un valor.
- La sintaxis para llamarlo es:

```
TipoClase variable = new TipoClase(argumentos);
```



Control de acceso

Alcance	Visibilidad
public	Accesible en cualquier lugar que una clase es accesible
protected	Accesible en subclases, en clases que residen en el mismo paquete y en la clase en donde esta definido.
private	Accesible solo en la clase en donde esta definido
default	Accesible por clases que residen en el mismo paquete y en la clase que en donde esta definido

Access Levels				
Modifier	Class	Package	Subclass	All Other
public	Y	Y	Y	Y
protected	Y	Y	Y	N
Default	Y	Y	N	N
private	Y	N	N	N



Getters y Setters

- Un conjunto de métodos se crean por lo general en una clase para leer/escribir específicamente los valores de las variables miembro.
- Estos se llaman getters - se utilizan para obtener los valores
- Y setters - se utilizan para cambiar los valores de las variables miembro.



Getters y Setters

- Los getters y setters son cruciales en las clases de Java, ya que se utilizan para gestionar el estado de un objeto.

Java Básico

```
public class HolaMundo {  
    public static void main(String[] args) {  
        System.out.println("Hola Mundo");  
    }  
}
```



Anahí Salgado
@anncode

- <https://docs.oracle.com/javase/tutorial/jdbc/basics/sqlstructured.html>