



Introducción a Java

Conceptos básicos

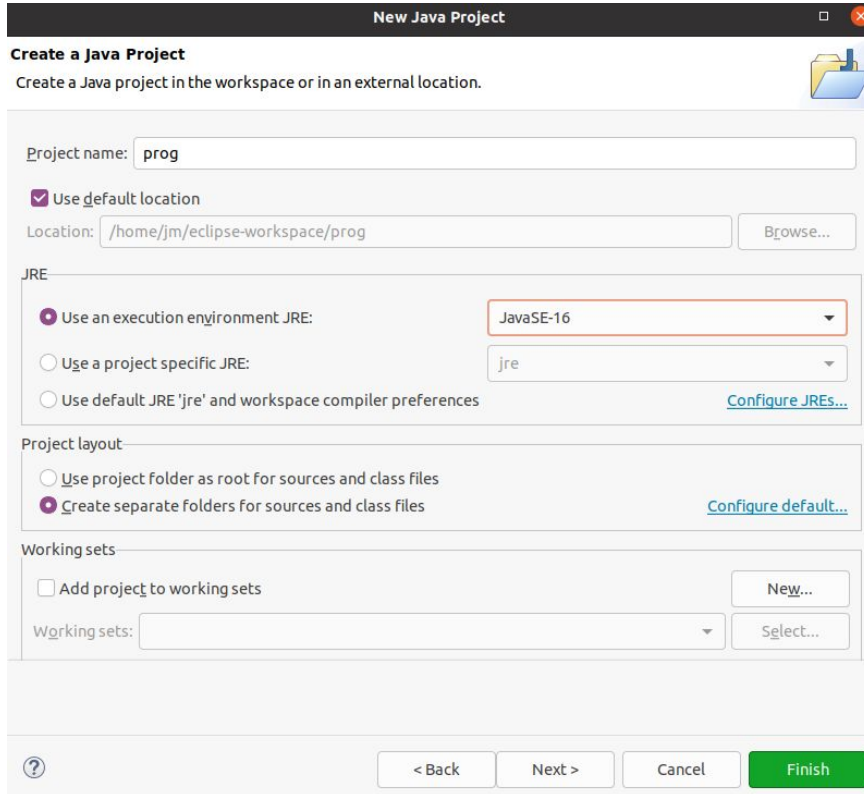


1. Características de Java

- James Gosling 1995
- Sun Microsystems
- Lenguaje de alto nivel
- Basado en clases
- Orientado a objetos
- Propósito general
- WORA - Write Once
Run Anywhere
- Compila a bytecode
- JVM
- Sintaxis similar a C y C++
- Uno de los lenguajes más populares en el desarrollo de aplicaciones web en entorno servidor (9M dev)



2. Primer programa en Java



New Java Project

Create a Java project in the workspace or in an external location.

Project name:

☒ Use default location
Location:

JRE:

☒ Use an execution environment JRE:

☐ Use a project specific JRE:

☐ Use default JRE 'jre' and workspace compiler preferences

Project layout:

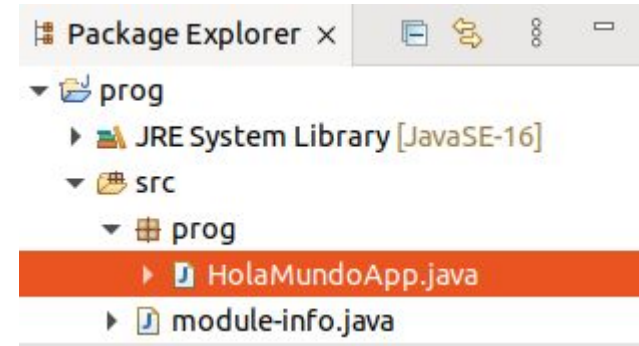
☐ Use project folder as root for sources and class files

☒ Create separate folders for sources and class files

Working sets

☐ Add project to working sets

Working sets:



2. Primer programa en Java

New Java Class

Create a new Java class.

Source folder: Browse...

Package: Browse...

☐ Enclosing type: Browse...

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

☒ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Cancel Finish



```
HolaMundoApp.java x
1 package prog;
2
3 public class HolaMundoApp {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7
8         System.out.println("Hola mundo");
9
10    }
11 }
12
13
```

Problems Javadoc Declaration Console x

<terminated> HolaMundoApp [Java Application] /home/jm/eclipse/plugin
Hola mundo

2. Primer programa en Java

HolaMundoApp.java x

```
1 package prog;
2
3 public class HolaMundoApp {
4
5     /**
6      * Programa principal que se ejecuta como punto de entrada a
7      * la aplicación que estemos creando
8      *
9      * @param args Como este programa se puede ejecutar desde consola podemos
10     * pasarle argumentos de entrada
11     */
12     public static void main(String[] args) {
13
14         System.out.println("Hola, " + args[0]);
15
16     }
17 }
18
19
```

```
jm@jm-L:~/eclipse-workspace/prog/src/prog$ java ./HolaMundoApp.java 'Jose Manuel'
Hola, Jose Manuel
jm@jm-L:~/eclipse-workspace/prog/src/prog$ java ./HolaMundoApp.java 'año 2022'
Hola, año 2022
jm@jm-L:~/eclipse-workspace/prog/src/prog$
```

2. 1. Estructura

```
1 // 1. Paquete donde está ubicado
2 package com.edu.mfp;
3
4 // 2. Importamos librerías
5 import java.util.Scanner;
6
7 // 3. Nombre de la clase
8 public class ClaseEjemplo {
9
10     // 4. Atributos y métodos
11
12     // Punto de entrada - método main
13     public static void main(String[] args) {
14         Scanner scanner = new Scanner(System.in);
15         System.out.println("Hello");
16     }
17
18
19
20 }
```

3. Tipos de datos básicos o primitivos

	<u>//nombre</u>	<u>#longitud</u>	<u>#rango de representación</u>	
numérico	byte z	= 8;	// -2 ⁷	2 ⁷ - 1
	short s	= 16;	// -2 ¹⁵	2 ¹⁵ - 1
	int entero	= 32;	// -2 ³¹	2 ³¹ - 1
	long l	= 64;	// -2 ⁶³	2 ⁶³ - 1
	float f	= 32;	// -2 ¹⁴⁹	(2 - 2 ²³)2 ¹²⁷ - 1
	double d	= 64;	// -2 ¹⁰⁷⁴	(2 - 2 ⁵²)2 ¹⁰²³ - 1
texto	char c	= 16;	// 0	2 ¹⁶ - 1
lógico	boolean b	= true ;	// 1	


```
entero = 1_000_000;  
s = 1_000;  
f = 1.967f;
```

4. Palabras reservadas

`abstract` `continue` `for` `new` `switch`
`assert` `default` `goto` `package` `synchronized`
`boolean` `do` `if` `private` `this`
`break` `double` `implements` `protected` `throw`
`byte` `else` `import` `public` `throws`
`case` `enum` `instanceof` `return` `transient`
`catch` `extends` `int` `short` `try`
`char` `final` `interface` `static` `void`
`class` `finally` `long` `strictfp` `volatile`
`const` `float` `native` `super` `while`



5. Identificadores y comentarios

- **Identificador:** nombre con el que se identifica cada variable
- **CaseSensitive**
- No existe longitud máxima
- Reglas:
 - empiezan por letra, _ o \$
 - los siguientes caracteres:
[A-Z][a-z][0-9] _ \$

// Comentario de una línea

/* Comentario

de longitud

variable */

6.1. Variables: declaración y definición

//Declaración

```
int nuevaVariable;
```

//Definición

```
nuevaVariable = 30;
```

//Declaración + definición:

```
int valor0 = 15;
```

//Declaración + definición múltiple:

```
int valor1 = 15, valor2 = 15;
```

- **Declarar** una variable implica reservar espacio en memoria para almacenar el valor futuro de ésta. Si el tipo utilizado es primitivo la variable queda inicializada automáticamente.
- **Definir** una variable conlleva dar un valor específico a la variable que ya ha sido declarada previamente.
- **Definir** y **declarar** pueden darse a la vez

6.2. Constantes

```
final float PI = 3.1415f;  
final int MAYORIA_EDAD = 18;  
final int REGION_ACTUAL = 1;
```

Similar a la declaración y definición de una variable, pero precedido por la palabra clave **final**.

Su valor no se modifica durante la ejecución del programa. Es **constante**.

6.3. Operadores

```
/* Operadores aritméticos: */
```

```
valor0 = valor1 + valor2;  
valor0 = valor1 - valor2;  
valor0 = valor1 / valor2;  
valor0 = valor1 * valor2;  
valor0 = valor1 % valor2;  
valor0++;  
valor0--;
```

```
valor0 += valor2;  
valor0 -= valor2;  
valor0 /= valor2;  
valor0 *= valor2;
```

```
/* Operadores relacionales */
```

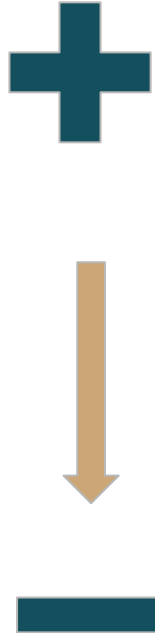
```
boolean varLogica;  
varLogica = valor0 == valor1;  
varLogica = valor0 < valor1;  
varLogica = valor0 <= valor1;  
varLogica = valor0 > valor1;  
varLogica = valor0 >= valor1;  
varLogica = valor0 != valor1;
```

```
varLogica = true && true;  
varLogica = true || false;  
varLogica = !varLogica;
```

6.3. Operadores. Precedencia

Operador

Tipo



++ --	Unario, notación postfija/prefija
* / %	Aritméticos
+ -	Aritméticos
< <= > >=	Relacionales
== !=	Relacionales
&&	Lógico
	Lógico
?:	Ternario
= += -= *= /= %=	Asignación

7. Instrucciones de Entrada/Salida

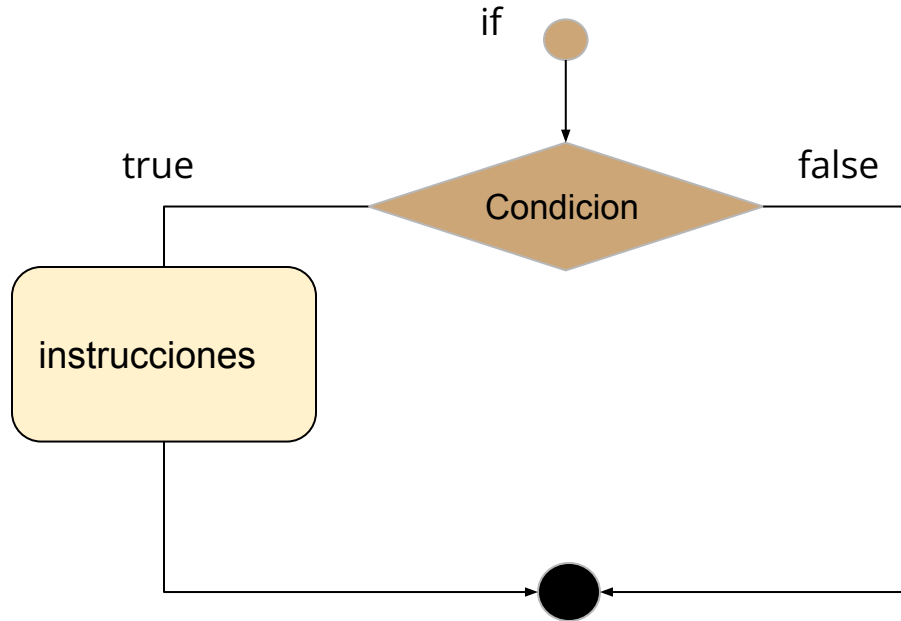
```
Scanner sc = new Scanner(System.in);  
String nombre = sc.next();
```

```
System.out.println("Hola "+ nombre);  
System.out.print("Hola mundo sin retorno");
```

9. Estructuras de control

9.1. Condicionales

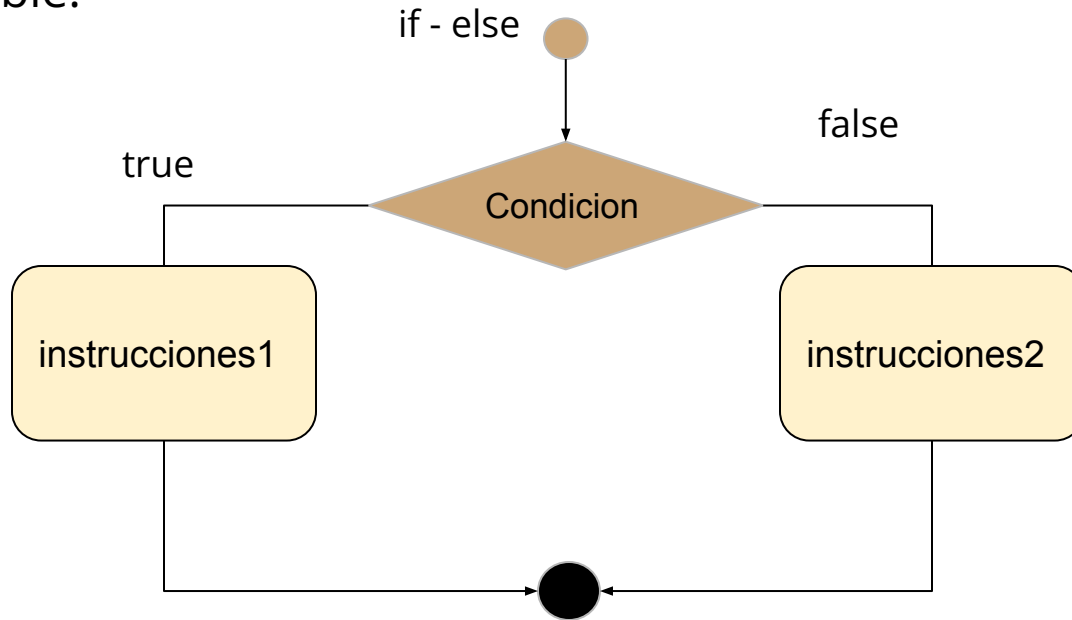
Conditional simple:



```
if (edad >= 18) {  
    System.out.println("Puede realizar la compra");  
}
```


9.1. Condicionales

Condicional doble:



9.1. Condicionales

```
boolean abierto = true;

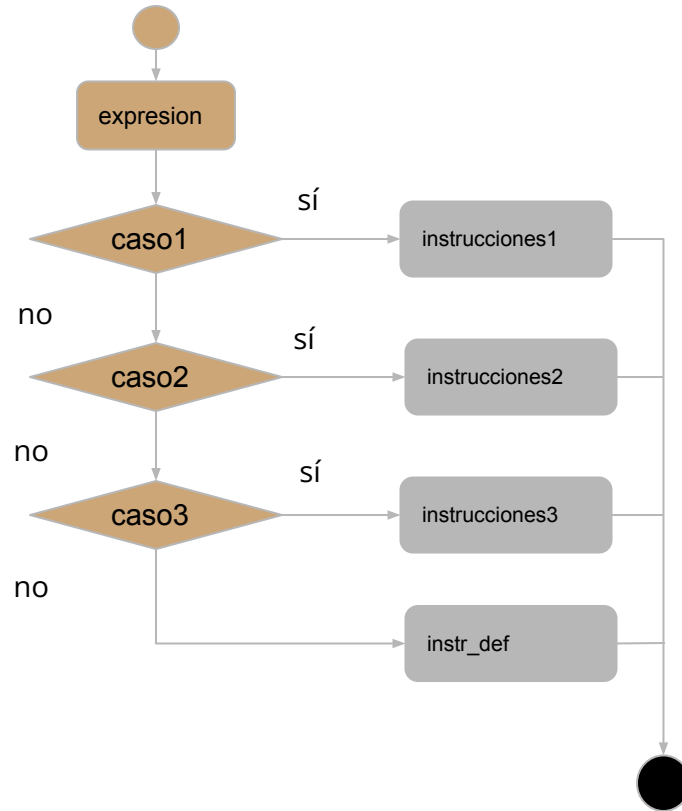
if (abierto) {
    System.out.println("Puede pasar");
}else {
    System.out.println("En otra ocasión");
}
```

```
if (dia == 1) {
    System.out.println("Lunes");
}else if(dia==2) {
    System.out.println("Martes");
}else if(dia==3) {
    System.out.println("Miércoles");
}else if(dia==4) {
    System.out.println("Jueves");
}else if(dia==5) {
    System.out.println("Viernes");
}else if(dia==6) {
    System.out.println("Sábado");
}else if(dia==7) {
    System.out.println("Domingo");
}else {
    System.out.println("El día introducido no es válido");
}
```

Actividad

1. Diseña un programa que pregunte por la edad de una persona e imprima si es mayor de edad.
2. Modifica el programa anterior para que indique el grupo de edad al que pertenece (niñ@ < 18, adulto, ancian@ > 65).
3. Realiza un programa (días del mes) que pregunte por un mes y año y devuelva el número de días que tiene ese mes para el año especificado.

9.2. Condicionales: switch



9.2. Condicionales: switch

```
switch (dia) {  
  case 1: {  
    System.out.println("Lunes");  
    break;  
  }  
  case 2: {  
    System.out.println("Martes");  
    break;  
  }  
  case 3: {  
    System.out.println("Miércoles");  
    break;  
  }  
  case 4: {  
    System.out.println("Jueves");  
    break;  
  }  
  case 5: {  
    System.out.println("Viernes");  
    break;  
  }  
  case 6: {  
    System.out.println("Sábado");  
    break;  
  }  
  case 7: {  
    System.out.println("Domingo");  
    break;  
  }  
  default:  
    System.out.println("El día introducido no es válido");  
}
```



```
switch (dia) {  
  case 1: {  
    System.out.println("Lunes");  
  
  }  
  case 2: {  
    System.out.println("Martes");  
  
  }  
  case 3: {  
    System.out.println("Miércoles");  
  
  }  
  case 4: {  
    System.out.println("Jueves");  
  
  }  
  case 5: {  
    System.out.println("Viernes");  
  
  }  
  case 6: {  
    System.out.println("Sábado");  
  
  }  
  case 7: {  
    System.out.println("Domingo");  
  
  }  
  default:  
    System.out.println("El día introducido no es válido");  
}
```

Actividad

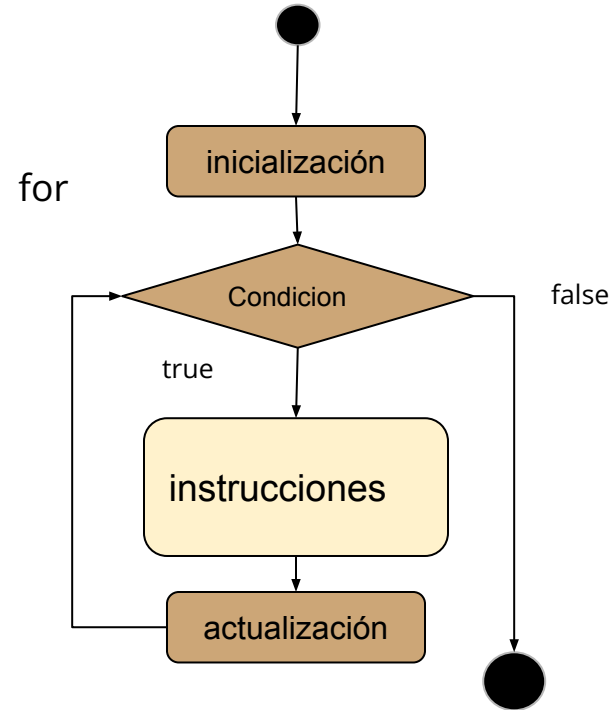
Modifica el programa realizado anteriormente (días del mes) para que en lugar de una estructura condicional doble (if else if) utilice un switch.

9.3.1. Control por contador

```
int i =1;  
  
while(i<=50) {  
    System.out.println(i*2);  
    i++;  
}
```

```
for(int i=1; i <=50; i++ ) {  
    System.out.println(i*2);  
}
```

El conjunto de instrucciones no se ejecuta si no se cumple la condición.

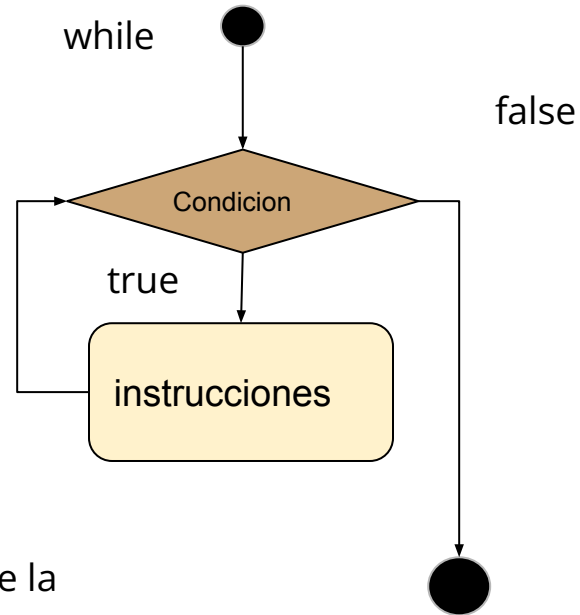


9.3.2. Control por condición

```
Scanner scanner = new Scanner(System.in);
int numero = 0;

while(numero>0) {
    System.out.println("Introduzca un número");
    numero = Integer.valueOf(scanner.next());
    if (numero%2==0){
        System.out.println("El número es par");
    }else {
        System.out.println("El número es impar");
    }
};
```

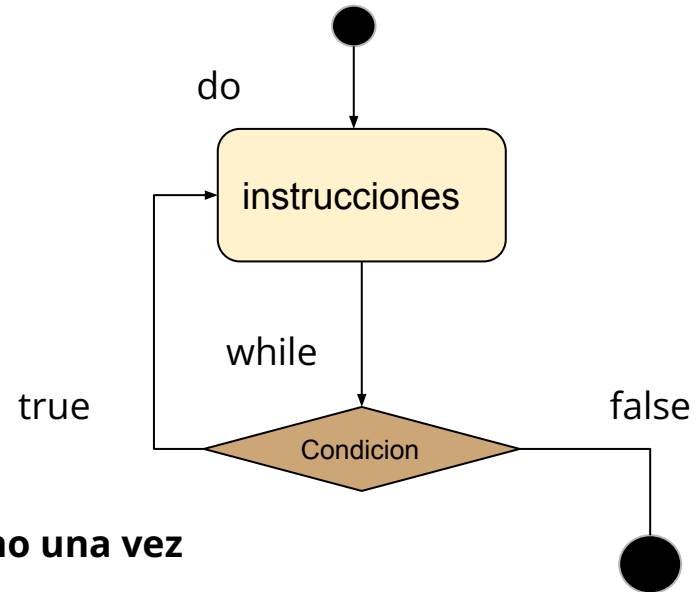
El conjunto de instrucciones no se ejecuta si no se cumple la condición.



9.3.2. Control por condición

```
Scanner scanner = new Scanner(System.in);  
int numero = 0;  
  
do {  
    System.out.println("Introduzca un número");  
    numero = Integer.valueOf(scanner.next());  
    if (numero%2==0){  
        System.out.println("El número es par");  
    }else {  
        System.out.println("El número es impar");  
    }  
}while(numero>0);
```

El conjunto de instrucciones se ejecuta como mínimo una vez



9.3.2. Control por condición

En un for tradicional además de la inicialización y actualización del contador, tenemos una condición a cumplir. Si eliminamos los primeros tenemos una estructura similar al while.

```
Scanner scanner = new Scanner(System.in);
int numero = 0;

for( ;numero>0; ) {
    System.out.println("Introduzca un número");
    numero = Integer.valueOf(scanner.next());
    if (numero%2==0){
        System.out.println("El número es par");
    }else {
        System.out.println("El número es impar");
    }
}
```