
ORACLE
Academy

Java Foundations

5-1

Expresiones boolean y construcciones if/else

ORACLE
Academy

Objetivos

- En esta lección se abordan los siguientes objetivos:
  - Declarar, inicializar y utilizar variables boolean
  - Comparar expresiones boolean mediante operadores relacionales
  - Crear una sentencia if
  - Crear construcciones if/else
  - Comparar objetos String

Toma de decisiones

- Hasta el momento en las lecciones anteriores, hemos visto diferentes tipos de dato soportados en Java
- boolean es otro tipo de dato en Java que ayuda a agregar lógica a un programa
- Ayuda a tomar decisiones

Toma de decisiones

ORACLE Academy

Toma de decisiones

- Digamos que conduce hacia la escuela
- Para en una intersección
- Y ahora, debe tomar una decisión lógica:
  - Si giro a la izquierda, ¿me llevará hacia la escuela?
  - Si sigo recto, ¿me llevará hacia la escuela?
  - Si giro a la derecha, ¿me llevará hacia la escuela?
- Solo hay dos respuestas a cada una de estas preguntas:
  - sí o no

Tipos de dato boolean de Java

- Básicamente es lo mismo en Java, donde los valores boolean indicarán al programa cuál es la mejor acción a realizar
- En Java, los valores para los tipos de dato boolean son true y false, en lugar de sí y no
- Declare los tipos de dato boolean mediante la palabra clave boolean

Uso de los tipos de dato boolean de Java: Ejemplo

- Nota: El valor de una variable boolean se muestra como true o false

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

   boolean passed, largeVenue, grade;

   passed = true;
   largeVenue = false;
   grade = passed;

   System.out.println(passed);
   System.out.println(largeVenue);
   System.out.println(grade);
}//end method main
```

Declaración de variables booleanas

Asignación de valores a variables booleanas

Impresión de los valores de las variables booleanas

Recuerde que un tipo de dato `boolean` solo puede tener dos valores posibles: true y false.
Nota: True y false nunca se entrecomillan porque son valores `boolean`, no cadenas.

Salida del ejemplo de la diapositiva:
true
false
true

Tipo de dato boolean: Situación

- ¿Qué haría si estuviera conduciendo un coche con un sistema GPS instalado que se ejecuta con Java?
- Antes de salir, pida al sistema GPS que le lleve a la escuela
- ¿Qué sencillo código escribiría para que le ayudara a decidir hacia qué dirección girar?

Según este sencillo código, el coche se dirigirá hacia la dirección que tenga una variable `boolean` con un valor de true.
Nota: La mejora de este ejemplo con más código se tratará con más detalle en esta lección.

Tipo de dato boolean: Situación

- Comencemos

```
public static void main(String args[]) {
    String left = "museum";
    String straight = "gym";
    String right = "restaurant";
    boolean isLeft = false;
    boolean isStraight = true;
    boolean isRight = false;
    System.out.println("Go straight ahead");
}//end method main
```


Según este sencillo código, el coche se dirigirá hacia la dirección que tenga una variable `boolean` con un valor de true.
Nota: La mejora de este ejemplo con más código se tratará con más detalle en esta lección.

Expresiones y variables

- Las expresiones matemáticas se pueden...
  - Imprimir
  - Asignar a una variable int o double

```
System.out.println(2 + 2);
int x = 2 + 2;
```

Expresiones y variables

- Las expresiones boolean se pueden...
  - Imprimir
  - Asignar a una variable boolean

```
System.out.println(x == 5);
boolean isFive = x == 5;
```


Utilice el signo igual (=) para hacer una asignación y el signo == para establecer una comparación y devolver un `boolean`.

Igualdad y asignación

- == es un operador relacional
- Este operador comprueba si ambas partes de una expresión boolean son iguales entre sí
- Una expresión boolean devuelve un valor de true o false

```
x == 5
```

Igualdad y asignación

- = es un operador de asignación
- Este operador asigna un valor a una variable
- Una variable boolean se puede asignar sea cual sea el valor que devuelva la expresión boolean

```
int x = 4;
boolean isFive = x == 5;
```

Valores de las expresiones boolean

- Utilice == para probar la igualdad entre valores primitivos
- Las expresiones boolean pueden contener variables o valores codificados

```
boolean res1 = 24 == 15;
System.out.println("res1: " + res1);
int value1 = 15;
int value2 = 24;
boolean res2 = value1 == value2;
System.out.println("res2: " + res2);
```

Valores de las expresiones boolean

- Las dos expresiones siguientes devuelven el mismo valor:
  - Si value1 y value2 cuentan con el mismo valor, la expresión devuelve un resultado true
  - De lo contrario, la expresión devuelve false

```
boolean res1 = 24 == 15;
System.out.println("res1: " + res1);
int value1 = 15;
int value2 = 24;
boolean res2 = value1 == value2;
System.out.println("res2: " + res2);
```

Operadores relacionales

- Utilice operadores relacionales en expresiones boolean que se utilizan para evaluar las sentencias if/else

A continuación se muestra una lista más completa de operadores relacionales. En la tabla se muestran las diferentes condiciones que puede probar mediante operadores relacionales. El resultado de todos los operadores relacionales es un valor `boolean`. Todos los ejemplos producen un resultado `boolean` true.

Operadores relacionales

Condición	Operador	Ejemplo
Es igual a	==	int i=1; (i == 1)
Es distinto de	!=	int i=2; (i != 1)
Es menor que	<	int i=0; (i < 1)
Es menor o igual que	<=	int i=1; (i <= 1)
Es mayor que	>	int i=2; (i > 1)
Es mayor o igual que	>=	int i=1; (i >= 1)

A continuación se muestra una lista más completa de operadores relacionales. En la tabla se muestran las diferentes condiciones que puede probar mediante operadores relacionales. El resultado de todos los operadores relacionales es un valor `boolean`. Todos los ejemplos producen un resultado `boolean` true.

Operadores relacionales: Ejemplo

- Nota Utilice el signo igual (=) para hacer una asignación y utilice el signo == para establecer una comparación y devolver un boolean

```
public static void main(String args[]) {
    int a = 10;
    int b = 20;
    System.out.println(a == b);
    System.out.println(a != b);
    System.out.println(a > b);
    System.out.println(a < b);
    System.out.println(b >= a);
    System.out.println(b <= a);
}//end method main
```

Para valores primitivos == comprueba la existencia de pruebas de igualdad

ORACLE Academy

JFo 5-1
Expresiones boolean y construcciones if/else

Pruebas de igualdad:
Para los objetos, utilice el método equals para la prueba de igualdad.
Resultado:

```
false
true
false
true
true
false
```

Ejercicio 1

- Cree un nuevo proyecto y agréguele el archivo `AgeValidity.java`
- Modifique `AgeValidity.java` para implantar lo siguiente:
  - Pedir a los usuarios introduzcan la edad
  - Declarar una variable boolean, drivingUnderAge
  - Inicializar drivingUnderAge en false
  - Escribir una expresión boolean para comprobar si la edad introducida por el usuario es menor o igual a 18 y, a continuación, definir drivingUnderAge en true
  - Imprimir el valor de drivingUnderAge

Sentencias Condicionales

- Las sentencias condicionales nos dejan elegir qué sentencia se ejecuta a continuación
- Estas decisiones se basan en expresiones boolean (o condiciones) que se evalúan como true o false
- Las sentencias condicionales en Java son:
  - Sentencia if
  - Sentencia if/else
  - Sentencia switch

Descripción de la sentencia if

- Una sentencia if consta de una expresión boolean seguida de una o varias sentencias
- Sintaxis:

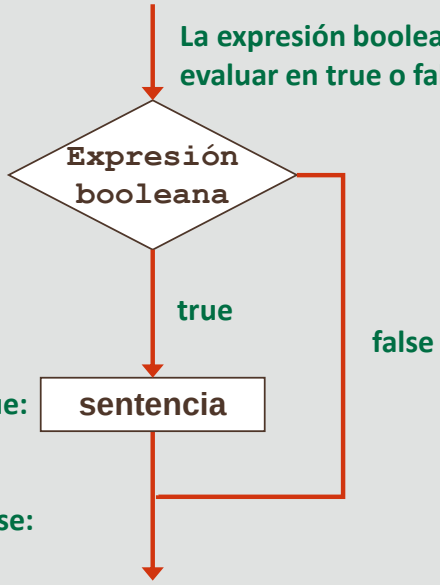
expresión booleana

```
if ( <some condition is true> ){

   //Statements will execute if the boolean
   //expression is true
}
```


La condición que se va a evaluar está entre paréntesis. Se hace referencia a ella como una expresión `boolean`, ya que debe evaluar en true o false.

Descripción de la sentencia if



Si la expresión booleana es true:
La sentencia se ejecuta

Si la expresión booleana es false:
La sentencia se omite

Uso de expresiones boolean en las sentencias if

```
public static void main(String args[]) {
    String left = "museum";
    String straight = "gym";
    String right = "restaurant";

    if (left == "gym") {
        System.out.println("Turn Left");
    }//endif

    if (straight== "gym") {
       System.out.println("Drive Straight");
    }//endif

    if (right == "gym") {
       System.out.println("Turn Right");
    }//endif
}//end method main
```

Este bloque se ejecuta

En el ejemplo de la diapositiva, la expresión `boolean` en la segunda sentencia `if` devuelve true. Por lo tanto, la opción "Drive Straight" se imprime en la consola.

Ejecución de un bloque de código

1. Un bloque de código no es necesario para que una sentencia if ejecute una sentencia
   - Por ejemplo:

1

```
daysInFeb = 28;
if(isLeapYear)
   daysInFeb = 29;
   System.out.println(year + "is a leap year");
```

Solo se ejecuta esta sentencia

En el ejemplo 1, desea que las dos sentencias se ejecuten cuando la condición es true. Como no hay bloque de código, solo se ejecuta la primera sentencia.
En el ejemplo 2, las dos sentencias se ejecutan cuando la condición es true, ya que hay un bloque de código para la sentencia `if`.
Para evitar este error, debe utilizar los bloques de código incluso `si` solo hay una sentencia que se va a ejecutar en el bloque `if`.

Ejecución de un bloque de código

2. Sin embargo, siempre se recomienda utilizar los bloques de código, incluso si solo hay una sentencia que se va a ejecutar en el bloque

2

```
daysInFeb = 28;
if(isLeapYear){
   daysInFeb = 29;
   System.out.println(year + " is a leap year");
} //endif
```

Este bloque se ejecuta

ORACLE Academy

JFo 5-1
Expresiones boolean y construcciones if/else

En el ejemplo 1, desea que las dos sentencias se ejecuten cuando la condición es true. Como no hay bloque de código, solo se ejecuta la primera sentencia.
En el ejemplo 2, las dos sentencias se ejecutan cuando la condición es true, ya que hay un bloque de código para la sentencia `if`.
Para evitar este error, debe utilizar los bloques de código incluso si solo hay una sentencia que se va a ejecutar en el bloque `if`.

Sentencia if: Ejemplos

```
public static void main(String args[]) {
   int grade = 85;

    if (grade > 88) {
        System.out.println("You made the Honor Roll.");
    } //endif

    if (grade <=88) {
        System.out.println("You are eligible for tutoring.");
    }//endif

}//end method main
```

Segunda sentencia if

- Resultado:

```
You are eligible for tutoring.
```

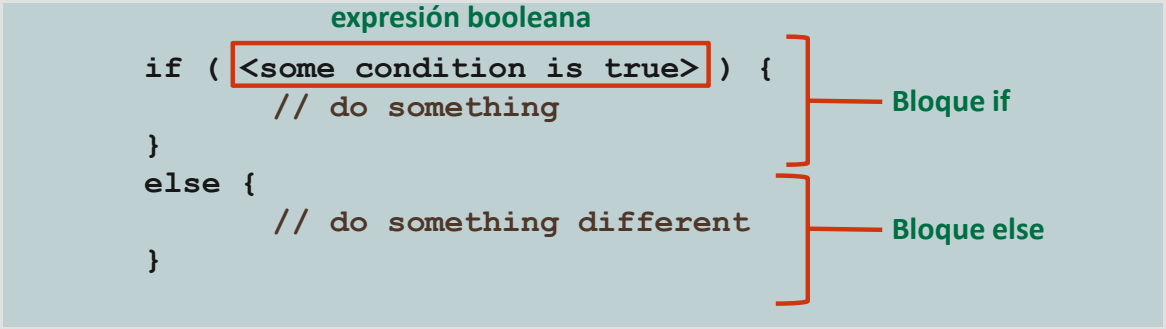

En el ejemplo de la diapositiva, hay dos sentencias `if`: La primera comprueba los valores superiores a 88. La segunda comprueba los valores inferiores o iguales a 88.
Las dos sentencias `if` se evalúan, incluso si la primera es true.

Ejercicio 2

- Agregue el archivo `ChkOddEven.java` al proyecto creado para el ejercicio 1
- Modifique `ChkOddEven.java` para implantar las siguientes acciones:
  - Introducir un número entre 1 y 10
  - Utilizar sentencias if
  - Comprobar si un número es par o impar
- El programa debe generar la siguiente salida:
  - Enter a number: 7
  - The num is odd 7

Elección entre dos alternativas

- Si desea elegir entre dos alternativas, utilice la sentencia if/else
- Sintaxis:



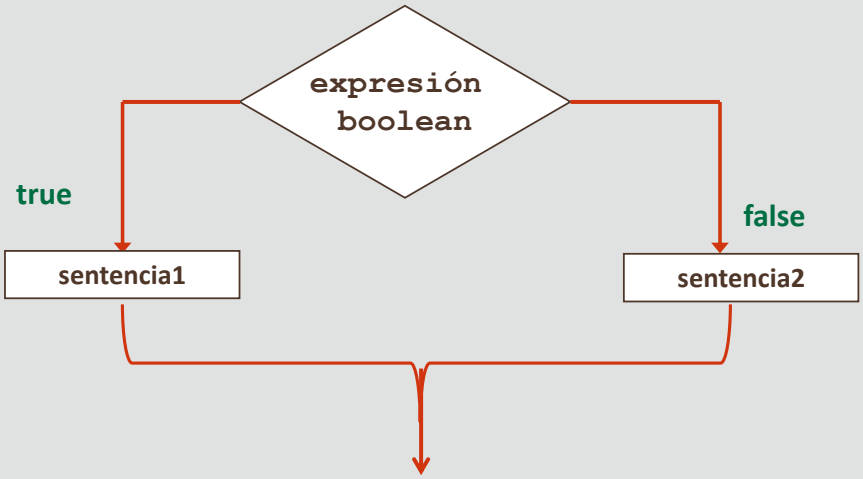
ORACLE Academy

JFo 5-1
Expresiones boolean y construcciones if/else

La sentencia `if/else` es una forma de bifurcación del código según determinadas condiciones. Utiliza dos las palabras clave de Java, `if` y `else`.

- Si alguna condición es true, ejecute el código en el bloque `if`.
- Si esa condición es false, ejecute el código en el bloque `else`.

Descripción de sentencias if/else



Si la expresión boolean es true: Se ejecuta la sentencia1
Si la expresión boolean es false: Se omite la sentencia2

Sentencias if/else: Ejemplo 1

```
String forecast;
double temperature = getTemperature();

if (temperature <= 32.0) {
   forecast = "SNOW";
}
else {
   forecast = "RAIN";
}//endif
```

Este bloque se ejecuta

30,3 °F

ORACLE Academy

JFo 5-1
Expresiones boolean y construcciones if/else

Esta diapositiva demuestra un ejemplo de `if/else`. El método `getTemperature()` devuelve la temperatura como 30,3. Ya que la temperatura es inferior a 32 grados, la expresión booleana `(temperature <= 32.0)` devuelve un valor true y se ejecuta el bloque `if`.

Sentencias if/else: Ejemplo 2

```
String forecast;
double temperature = getTemperature();

if (temperature <= 32.0) {
   forecast = "SNOW";
}
else {
   forecast = "RAIN";
}//endif
```

Este bloque se ejecuta

40,2 °F

ORACLE Academy

Esta diapositiva demuestra un ejemplo de `if/else`, donde el método `getTemperature()` devuelve la temperatura como 40.2. En la sentencia `if/else`, ya que `temperature <= 32` grados, la expresión `boolean temperature <= 32.0` devuelve un valor false y el bloque `else` se ejecuta.

Sentencias if/else: Ejemplo 3

- Puede sustituir las dos sentencias if por una sentencia if/else
- La sentencia if/else es más eficaz porque solo se realiza una comparación

```
public static void main(String args[]) {
    int grade = 85;
    if (grade > 88) {
        System.out.println("You made the Honor Roll.");
    }
    else {
        System.out.println("You passed.");
    }//endif
}//end method main
```

Ejercicio 3

- Agregue el archivo `AgeCheck.java` al proyecto creado para el ejercicio 1
- Examine `AgeCheck.java`:
  - El programa tiene un problema de lógica
  - Para algunos valores, imprime la respuesta incorrecta
  - Busque los problemas y corríjalos (Puede que necesite ejecutar el programa unas cuantas veces y probar distintos valores para ver los que fallan)
  - Sustituya las dos sentencias if por una sentencia if/else

Ejercicio 4

- Agregue el archivo `ShoppingCart.java` al proyecto creado para el ejercicio 1
- Examine `ShoppingCart.java`
- Utilice una sentencia if/else para implantar lo siguiente:
  - Declarar e inicializar una variable boolean, outOfStock
  - Si la cantidad > 1, cambiar la variable de mensaje para indicar plural
  - Si un elemento está agotado, informar al usuario de que el artículo no está disponible Imprimir además el mensaje y el costo total

Comparación de variables

- Al comparar valores mediante expresiones boolean, deberá conocer los matices de ciertos tipos de dato
- Los operadores relacionales como == son...
  - Ideales para la comparación de valores primitivos
  - Malos para la comparación de objetos String (y otros objetos)
- Vamos a examinar el motivo

Comparación de valores primitivos

- El valor z se define para que sea la suma de x + y
- Cuando una expresión boolean comprueba la igualdad entre z y la suma de x + y, el resultado es true

```
int x = 3;
int y = 2;
int z = x + y;

boolean test = (z == x + y);
System.out.println(test);           //true
```


Nota para los instructores: el cuadro código de esta diapositiva debería estar en la misma posición para que las ligeras diferencias se hagan más evidentes.

Comparación de objetos String

- El valor z se define para que sea la concatenación de x + y
- Cuando una expresión boolean comprueba la igualdad entre z y la concatenación de x + y, el resultado es false

```
String x = "Ora";
String y = "cle";
String z = x + y;

boolean test = (z == x + y);
System.out.println(test);          //false
```

¿Por qué hay resultados contradictorios?

- Los primitivos y los objetos se almacenen de manera diferente en la memoria
  - A los objetos String se les da un tratamiento especial.
  - Esto se trata más adelante en el curso
- Como resultado...
  - == compara los valores de primitivos
  - == compara las ubicaciones de los objetos en la memoria
- Es mucho más probable que necesite comparar el contenido de los objetos String y no sus ubicaciones en la memoria

¿Cómo se deben comparar los objetos String?

- Casi nunca se deben comparar los objetos String mediante ==
- En su lugar, compare los objetos String mediante el método equals()
  - Este método es parte de la clase String
  - Acepta un argumento String, comprueba si el contenido de los objetos String son iguales y, a continuación, devuelve un boolean
  - También hay un método similar, equalsIgnoreCase()

```
String x = "Ora";
String y = "cle";
String z = x + y;
boolean test = z.equals(x + y);
System.out.println(test);                    //true
```


¿Cómo escribiría la sentencia
boolean test = z.equals(x + y);

de modo que el valor de la prueba sea true independientemente de si se usan mayúsculas o minúsculas?

Respuesta:
boolean test = z.equalsIgnoreCase(x + y);

Ejercicio 5

- Agregue el archivo `StringEquality.java` al proyecto creado para el ejercicio 1
- Examine `StringEquality.java`
- Utilice una sentencia if y una sentencia if/else:
  - Declare un nombre de la variable String
  - Pida al usuario que introduzca un valor para el nombre
  - Compruebe si el nombre es "Moe" y, a continuación, imprima “You are the king of rock and roll”
  - De lo contrario, imprima “You are not the king”
  - No utilice ==

Resumen

- En esta lección, debe haber aprendido lo siguiente:
  - Declarar, inicializar y utilizar variables boolean
  - Comparar valores primitivos mediante operadores relacionales
  - Crear una sentencia if
  - Crear construcciones if/else
  - Comparar objetos String


ORACLE
Academy