SWITCH

La instrucción switch es una instrucción de múltiples vías. Proporciona una forma sencilla de enviar la ejecución a diferentes partes del código en función del valor de la expresión. Básicamente, la expresión puede ser tipos de datos primitivos byte, short, char e int. A partir de JDK7, también funciona con tipos enumerados (Enum en java), la clase String y las clases Wrapper.

Sintaxis de Switch-case:

```
// declaración de switch
        switch(expresión)
        {
          // declaración case
          // los valores deben ser del mismo tipo de la expresión
          case valor1 :
```

```
    // Declaraciones
    break; // break es opcional
  case valor2 :
    // Declaraciones
    break; // break es opcional
  // Podemos tener cualquier número de declaraciones de casos o case
  // debajo se encuentra la declaración predeterminada, que se usa cuando ninguno de los casos es verdadero.
  // No se necesita descanso en el case default
  default :
    // Declaraciones
}
```

Algunas reglas importantes para declaraciones switch:

- Los valores duplicados de los case no están permitidos.
- El valor para un case debe ser del mismo tipo de datos que la variable en el switch.
- El valor para un case debe ser una constante o un literal. Las variables no están permitidas.
- La declaración break se usa dentro del switch para finalizar una secuencia de instrucción.
- La declaración break es opcional. Si se omite, la ejecución continuará en el siguiente case.
- La instrucción default es opcional, y debe aparecer al final del switch.

Omitir la declaración break

Como la declaración break es opcional. Si omitimos el break, la ejecución continuará en el siguiente case. A veces es deseable tener múltiples case sin declaraciones break entre ellos.

Declaraciones anidadas de Switch Case

Podemos usar un switch como parte de la secuencia de la declaración de un switch externo. Esto se llama un switch anidado. Como una instrucción de switch define su propio bloque, no surgen conflictos entre las constantes de case en el switch interno y las del switch externo. Por ejemplo:

String en Switch Case

Desde JDK 7, podemos usar una cadena literal/constante para controlar una declaración switch, lo cual no es posible en C/C++. Usar un modificador basado en cadena/string es una mejora con respecto al uso de la secuencia equivalente if/else.

Puntos importantes:

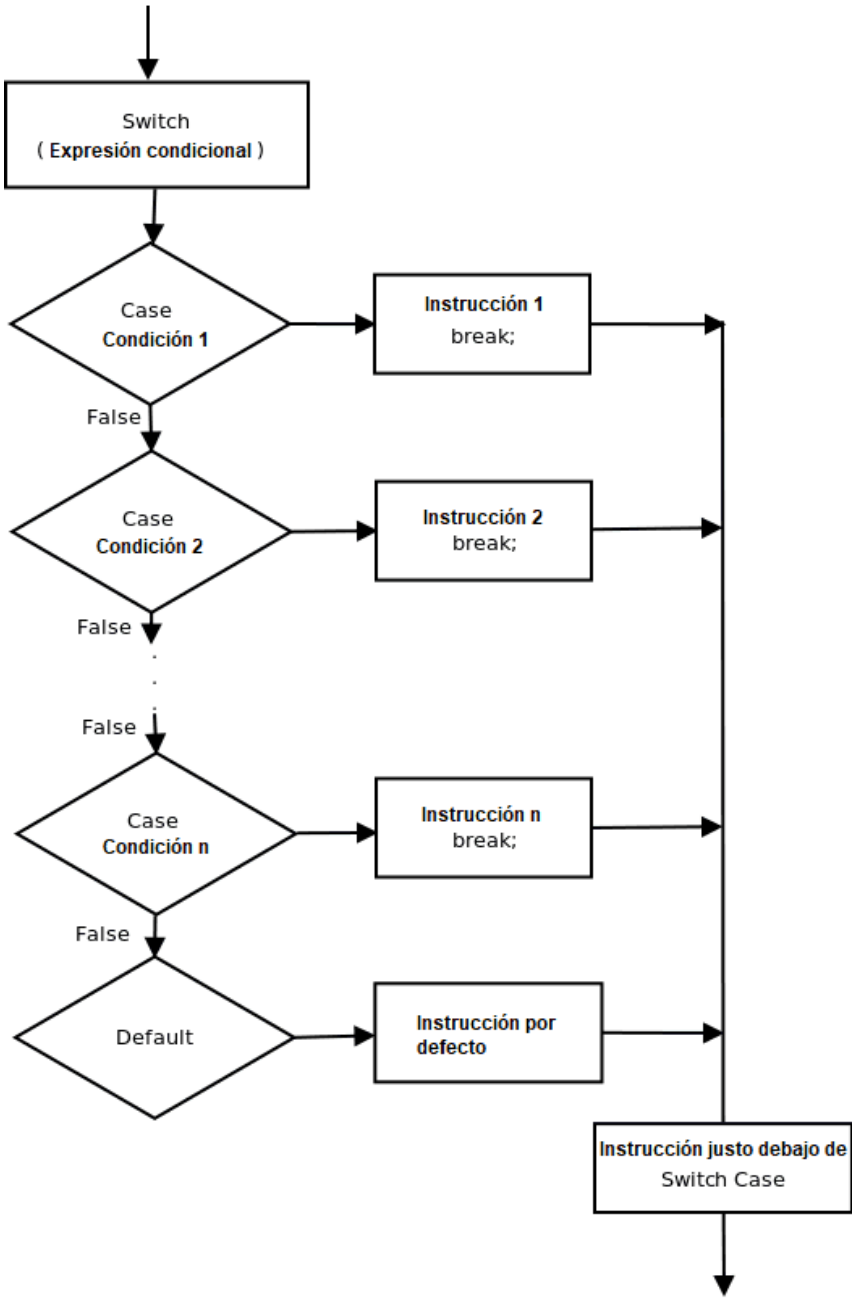
Operación costosa: el “switching” de strings puede ser más costosa en términos de ejecución que el switching de tipos de datos primitivos. Por lo tanto, es mejor activar el switch con strings solo en casos de que los datos de control ya estén en forma de cadena.

String no debe ser NULL: asegúrese de que la expresión en cualquier instrucción switch no sea nula mientras se trabaja con cadenas para evitar que una NullPointerException sea lanzada en tiempo de ejecución.

Case Sensitive – mayúsculas/minúsculas: la instrucción switch compara el objeto String en su expresión con las expresiones asociadas con cada etiqueta de case como si estuviera usando el método String.equals; en consecuencia, la comparación de objetos String en sentencias switch es sensible a mayúsculas y minúsculas.

Mejor que if-else: el compilador Java genera bytecode generalmente más eficiente a partir de sentencias switch que usan objetos String que de sentencias if-else anidadas.

Esquema de un SWITCH


Switch
(Expresión condicional)
Case
Condición 1
Instrucción 1
break;
False
Case
Condición 2
Instrucción 2
break;
False
False
Case
Condición n
Instrucción n
break;
False
Default
Instrucción por
defecto
Instrucción justo debajo de
Switch Case