Estructura Switch en Java

Estructura `switch` en Java

La estructura `switch` en Java permite seleccionar una de varias posibles acciones a realizar, basándose en el valor de una expresión. Es una alternativa más legible y manejable a las múltiples sentencias `if-else if-else`.

Sintaxis Clásica de `switch`

```
switch (expresion) {
    case valor1:
        // Bloque de código para el caso valor1
        break;
    case valor2:
        // Bloque de código para el caso valor2
        break;
        // Puedes tener tantos casos como necesites
    default:
        // Bloque de código para el caso por defecto
        break;
}
```

Ejemplo 1: Días de la Semana

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

        var dia = 3; // Suponiendo que 1 es Lunes, 2 es Martes, etc.
```

```
        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("Día inválido");
                break;
        }
    }
}
```

Mejoras en las Versiones Recientes de Java

En Java 12 y versiones posteriores, se introdujeron las expresiones `switch`, que permiten utilizar una sintaxis más compacta y limpia. También soportan `switch` con múltiples etiquetas de caso y bloques de texto.

Ejemplo 2: Uso de Expresiones `switch`

```
public class DiasSemanaExpresiones {
    public static void main(String[] args) {
        int dia = 3;
        String nombreDia;

        nombreDia = switch (dia) {
            case 1 -> "Lunes";
            case 2 -> "Martes";
            case 3 -> "Miércoles";
            case 4 -> "Jueves";
            case 5 -> "Viernes";
            case 6 -> "Sábado";
            case 7 -> "Domingo";
            default -> "Día inválido";
        };

        System.out.println(nombreDia);
    }
}
```

Ejercicios Prácticos

Ejercicio 1: Calificaciones

```
import java.util.Scanner;

public class SistemaCalificaciones {
    public static void main(String[] args) {
        Scanner consola = new Scanner(System.in);
        System.out.print("Proporciona una calificación entre 0 y 10: ");
        var calificacion = Integer.parseInt(consola.nextLine());
        var calificacionLetra;

        calificacionLetra = switch (calificacion) {
            case 9, 10 -> "A";
            case 8 -> "B";
            case 7 -> "C";
            case 6 -> "D";
            case 0, 1, 2, 3, 4, 5 -> "F";
            default -> "Calificación incorrecta";
        };

        System.out.printf("Calificación %.1f es equivalente a %s%n",
            calificacion, calificacionLetra);
    }
}
```

Ejercicio Final

Ejercicio: Estación del Año utilizando `switch`

```
import java.util.Scanner;

public class EstacionAnioSwitch {
    public static void main(String[] args) {
        Scanner consola = new Scanner(System.in);
        System.out.print("Proporciona el valor del mes (1-12): ");
        int mes = Integer.parseInt(consola.nextLine());
        String estacion;

        switch (mes) {
            case 1, 2, 12 -> estacion = "Invierno";
            case 3, 4, 5 -> estacion = "Primavera";
            case 6, 7, 8 -> estacion = "Verano";
            case 9, 10, 11 -> estacion = "Otoño";
            default -> estacion = "Estación desconocida";
        }

        System.out.printf("La estación para el mes %d es %s%n", mes, estacion);
    }
}
```

Significado de `yield`

En la sintaxis de `switch` mejorado de Java (introducido en Java 12 y posterior, y formalizado en Java 14), el uso de `yield` es una característica clave para las expresiones `switch`. Aquí te explico qué significa y cómo se usa.

El `yield` se utiliza dentro de una expresión `switch` para devolver un valor desde un caso específico. En la sintaxis clásica de `switch`, los casos no devuelven valores, solo ejecutan código. Con las nuevas expresiones `switch`, cada caso puede devolver un valor utilizando `yield`, lo que permite asignar el resultado del `switch` a una variable.

Ejemplo y Explicación

Veamos el ejemplo específico del código que has pedido y otros ejemplos adicionales.

Ejemplo 1: Sistema de Envíos

```
import java.util.Scanner;

public class SistemaEnvios {
    public static void main(String[] args) {
        System.out.println("*** Sistema de Envíos ***");

        final double TARIFA_NACIONAL = 10;
        final double TARIFA_INTERNACIONAL = 20;

        Scanner consola = new Scanner(System.in);

        System.out.print("Ingresa el destino del paquete (nacional/internacional): ");
        String destino = consola.nextLine().strip().toLowerCase();

        System.out.print("Ingresa el peso del paquete (en kg): ");
        double peso = Double.parseDouble(consola.nextLine());

        // Cálculo del envío del paquete usando switch con yield
        Double costoEnvio = switch (destino) {
            case "nacional" -> peso * TARIFA_NACIONAL;
            case "internacional" -> peso * TARIFA_INTERNACIONAL;
            default -> {
                System.out.println("Destino no válido. Ingresa el valor de nacional o internacional");
                yield null;
            }
        };

        if (costoEnvio != null) {
            System.out.printf("El costo de envío del paquete es: $%.2f%n", costoEnvio);
        }
    }
}
```

Explicación del `yield`

1. **Uso en el `default`**:
   - En el `default` case, `yield null;` se utiliza para devolver `null` cuando el destino no es válido.
   - `yield` actúa como una forma de devolver un valor desde el `switch` hacia la variable `costoEnvio`.
2. **Asignación del Resultado**:
   - El `switch` ahora puede ser utilizado como una expresión que devuelve un valor.
   - Cada `case` puede devolver un valor utilizando `yield`, y ese valor se asigna a `costoEnvio`.

Ejemplo Adicional: Días de la Semana

```
public class DiasSemana {
    public static void main(String[] args) {
        int dia = 3; // Suponiendo que 1 es Lunes, 2 es Martes, etc.
        String nombreDia = switch (dia) {
            case 1 -> "Lunes";
            case 2 -> "Martes";
            case 3 -> "Miércoles";
            case 4 -> "Jueves";
            case 5 -> "Viernes";
            case 6 -> "Sábado";
            case 7 -> "Domingo";
            default -> {
                yield "Día inválido"; // Devolver un valor con yield
            }
        };

        System.out.println(nombreDia);
    }
}
```

Beneficios de `yield` en las Expresiones `switch`

1. **Concisión y Claridad**:
   - Permite escribir `switch` más concisos y legibles, eliminando la necesidad de múltiples variables temporales.
2. **Menos Errores**:
   - Reduce la posibilidad de olvidar `break` en los casos del `switch` clásico, lo que podría causar errores lógicos.
3. **Flexibilidad**:
   - Permite devolver valores directamente desde los casos, facilitando la asignación de resultados a variables.

Resumen

La estructura `switch` en Java es una herramienta poderosa para realizar selecciones múltiples. Con la sintaxis clásica, puedes manejar múltiples casos de forma clara, y con las mejoras recientes en Java, puedes hacer que tu código sea aún más conciso y legible. Estos ejemplos y ejercicios prácticos te ayudarán a entender y aplicar el `switch` en tus propios programas.

El uso de `yield` en las expresiones `switch` en Java proporciona una forma elegante y eficiente de devolver valores desde los casos del `switch`. Esto hace que el `switch` sea más potente y flexible, permitiendo su uso en contextos donde se necesita devolver valores, no solo ejecutar bloques de código.

¡Espero que esta lección sea útil para ti!

Saludos!

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx