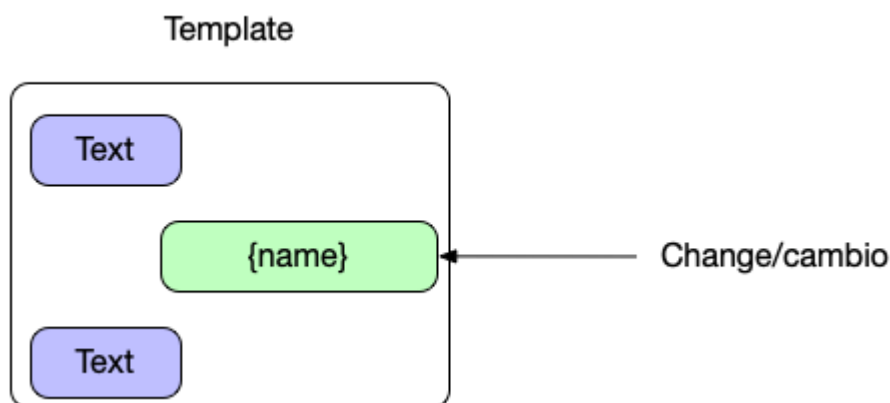


Java template strings o cadenas plantilla ya forman parte oficial del lenguaje a partir de Java 24, sin necesidad de activar funcionalidades en modo *preview*. Esta característica mejora significativamente la forma en que construimos cadenas dinámicas, haciendo el código más expresivo, legible y seguro.



¿Qué son los Java Template String ?

Antes de Java 24, generar cadenas dinámicas implicaba concatenaciones con el operador `+` o el uso de métodos como `String.format()`, lo que a menudo resultaba en código difícil de leer y mantener. Con los template strings, Java permite insertar directamente variables y expresiones dentro de una cadena utilizando `\{ }` como delimitadores.

Ejemplo Básico

```
import static java.util.Formatter.STR;

public class Main {
    public static void main(String[] args) {
        String nombre = "Laura";
        int edad = 28;

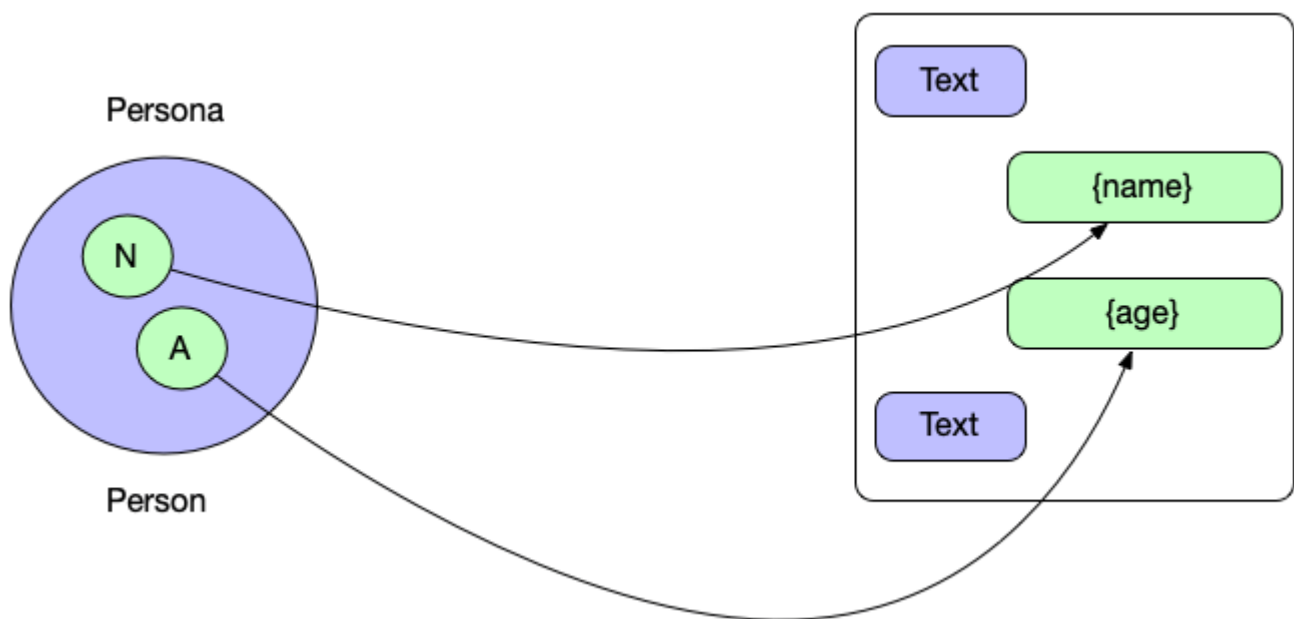
        String mensaje = STR."Hola, soy \{nombre} y tengo \{edad}
años.";
        System.out.println(mensaje);
    }
}
```

```
}  
}
```

Este ejemplo usa el procesador de plantilla STR que convierte automáticamente los valores a texto, escapando caracteres si es necesario.

Ejemplo con Clase Persona

A continuación, un ejemplo más completo usando una clase Persona y accediendo a sus propiedades mediante métodos *getter*:



```
import static java.util.FormatterProcessor.STR;
```

```
public class Persona {  
    private String nombre;  
    private int edad;  
  
    public Persona(String nombre, int edad) {  
        this.nombre = nombre;  
        this.edad = edad;  
    }  
}
```

```

    }

    public String getNombre() {
        return nombre;
    }

    public int getEdad() {
        return edad;
    }

    public static void main(String[] args) {
        Persona persona = new Persona("María", 32);

        String mensaje = STR."La persona se llama
        \{persona.getNombre()} y tiene \{persona.getEdad()} años.";
        System.out.println(mensaje);
    }
}

```

Aquí se demuestra cómo integrar los template strings directamente con objetos del dominio. Aunque no se puede acceder directamente con sintaxis tipo `\{persona.nombre}`, es totalmente válido llamar a los métodos `getNombre()` y `getEdad()` dentro del template.

Ventajas de los Template Strings

- Legibilidad: Código más limpio y directo que concatenaciones o `String.format()`.
- Seguridad: Escapado automático de caracteres especiales.
- Extensibilidad: Posibilidad de definir procesadores personalizados si necesitas lógica adicional.

Conclusión

Los template strings en Java 24 representan una evolución significativa en el lenguaje.

Permiten escribir código más moderno y expresivo, con menos propensión a errores, ideal para cualquier desarrollador que busque claridad y eficiencia en su trabajo diario.

- [Java string literal vs string object](#)
- [Utilizando JavaScript template String](#)
- [Destructuring un concepto de Javascript](#)
- [HTML5 template element](#)
- [JSP JavaScript Template Literal](#)