Cuando pensamos en cadenas de texto en Java, lo normal es usar la clase String. Sin embargo, existe una interfaz llamada CharSequence, que nos da más flexibilidad para trabajar con secuencias de caracteres.

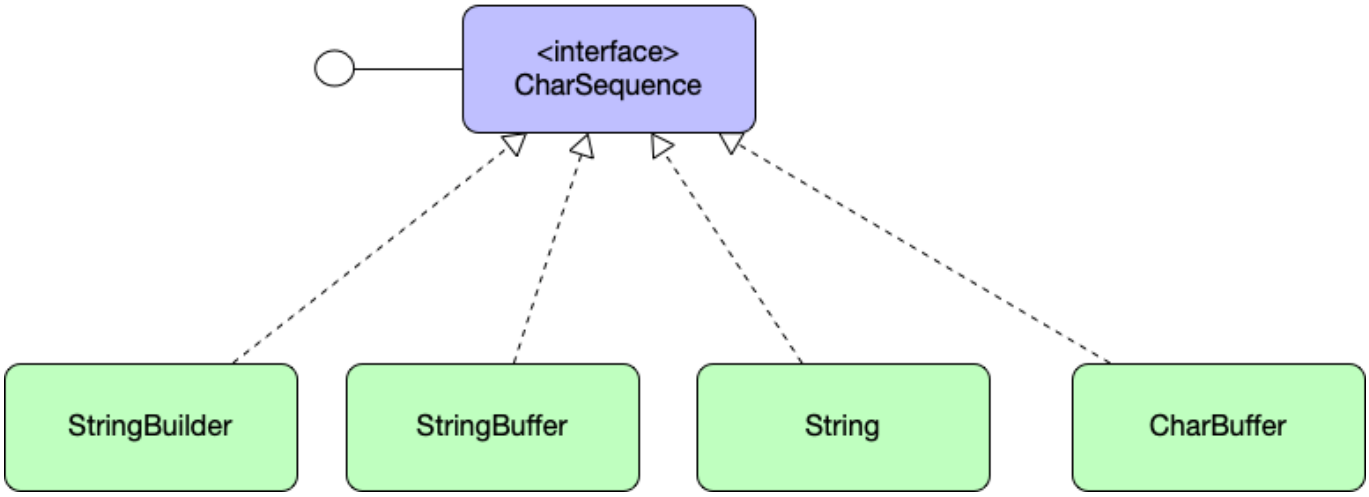
En este artículo veremos qué es, qué implementaciones tiene y cómo usarla en nuestros programas.

¿Qué es CharSequence?

CharSequence es una interfaz (desde Java 1.4) que representa una secuencia de caracteres. Se encuentra en:

`java.lang.CharSequence`

Sirve como una forma genérica de trabajar con texto, sin importar si la implementación es un String, un StringBuilder o cualquier otra clase.



Métodos principales

Estos son los métodos que define la interfaz:

- length() → devuelve el número de caracteres.
- charAt(int index) → devuelve el carácter en una posición concreta.

- subSequence(int start, int end) → obtiene una subsecuencia de caracteres.
- toString() → convierte la secuencia en un objeto String.
  Ejemplo:

```
CharSequence texto = "Hola Mundo";

System.out.println("Longitud: " + texto.length());
System.out.println("Primer carácter: " + texto.charAt(0));
System.out.println("Subsecuencia: " + texto.subSequence(0, 4));
```

  Salida:

```
Longitud: 10
Primer carácter: H
Subsecuencia: Hola
```

Implementaciones de CharSequence

Las implementaciones más comunes son:

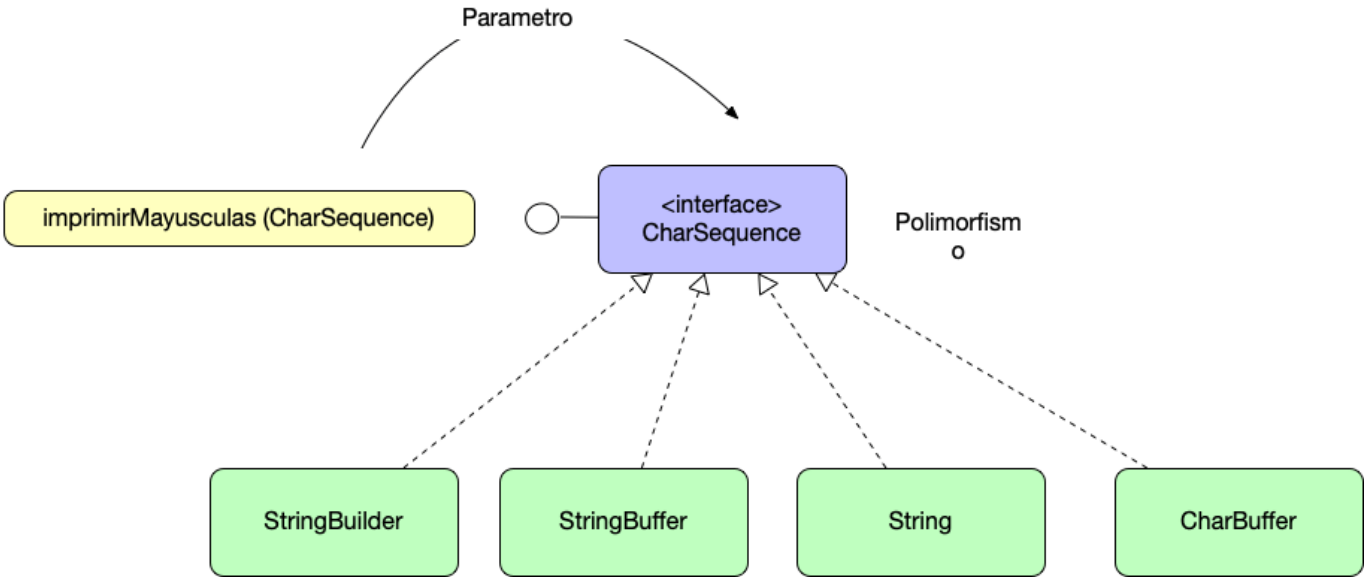
1. String → inmutable, la más usada.
2. StringBuilder → mutable, ideal para concatenar o modificar texto en un hilo.
3. StringBuffer → mutable y sincronizado, pensado para entornos multi-hilo.
4. CharBuffer → de java.nio, útil para trabajar con buffers en operaciones de E/S.

¿Por qué usar CharSequence?

Usar CharSequence como tipo de referencia tiene ventajas:

- Permite escribir código genérico que acepta varias implementaciones.
- Favorece la abstracción y el polimorfismo.
- Hace que nuestro código sea más reutilizable.

Para ello nos tendremos que apoyar el el polimorfismo :



Ejemplo:

```
public static void imprimirEnMayusculas(CharSequence secuencia) {
    System.out.println(secuencia.toString().toUpperCase());
}

public static void main(String[] args) {
    imprimirEnMayusculas("hola");
    imprimirEnMayusculas(new StringBuilder("mundo"));
}
```

Salida:

```
HOLA
MUNDO
```

Conclusión

Aunque normalmente trabajamos con String, la interfaz CharSequence nos ayuda a escribir código más flexible y reutilizable.
Si necesitas un método que funcione tanto con String, como con StringBuilder o StringBuffer, usar CharSequence es la mejor opción.

- Java Generics (II) uso de WildCard
- Java String y metodos fundamentales
- Gaphor UML y software OpenSource
- Java Stream File y manejo de ficheros
- TypeScript interface utilizando Angular DTOs