StringBuilder en Java es una alternativa a String para manejar y modificar cadenas de texto de manera más eficiente y cuando se necesitan hacer múltiples cambios a la cadena original. A diferencia de los objetos de tipo String, que son inmutables, StringBuilder permite la mutabilidad, es decir, los strings pueden ser cambiados sin necesidad de crear un nuevo objeto cada vez.

1)Constructores:

StringBuilder(): Crea un constructor sin caracteres y una capacidad inicial de 16 caracteres.

StringBuilder(int capacity): Crea un constructor con una capacidad inicial especificada.

StringBuilder(String str): Crea un objeto que contiene la misma secuencia de caracteres que el String dado y una capacidad extra para 16 caracteres adicionales.

Codigo de ejemplo:

```
StringBuilder sb1 = new StringBuilder();

StringBuilder sb2 = new StringBuilder(50);

StringBuilder sb3 = new StringBuilder("Hello");

System.out.println("sb1 initial capacity: " + sb1.capacity());

System.out.println("sb2 initial capacity: " + sb2.capacity());

System.out.println("sb3 contents: " + sb3.toString());
```

2) SetLength()

setLength(int newLength): Establece la longitud del contenido actual del StringBuilder. Si newLength es menor que la longitud actual, se trunca el contenido. Si es mayor, se rellena con caracteres nulos.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello World");

sb.setLength(5);

System.out.println(sb.toString()); // "Hello"

sb.setLength(10);

System.out.println(sb.toString()); // "Hello\0\0\0\0\0"
```

3)ensureCapacity

ensureCapacity(int minimumCapacity): Asegura que la capacidad sea al menos igual a la especificada. Si la capacidad actual es menor, se aumenta.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder();

sb.ensureCapacity(100);

System.out.println("Capacity: " + sb.capacity());
```

4)append

append: Añade la representación de texto de la data proporcionada al final del contenido actual de StringBuilder.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello");

sb.append(" World");

System.out.println(sb.toString()); // "Hello World"
```

5)insert

insert(): Inserta la representación de texto del dato proporcionado en la posición especificada.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello World");

sb.insert(6, "Beautiful ");

System.out.println(sb.toString()); // "Hello Beautiful World"
```

6)delete

delete(int start, int end): Elimina la subsecuencia de caracteres en el StringBuilder desde el índice start hasta end-1.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello Beautiful World");

sb.delete(6, 16);

System.out.println(sb.toString()); // "Hello World"
```

7)deleteAtChar

deleteCharAt(int index): Elimina el carácter en la posición especificada.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello World");

sb.deleteCharAt(5); // Elimina el espacio

System.out.println(sb.toString()); // "HelloWorld"
```

8)reverse

reverse(): Invierte el orden de los caracteres en el StringBuilder.

Codigo de ejemplo:

```
StringBuilder sb = new StringBuilder("Hello World");

sb.reverse();

System.out.println(sb.toString()); // "dlroW olleH"
```