StringBuilder

Los objetos StringBuilder son como objetos String, excepto que se pueden modificar. Internamente, estos objetos se tratan como matrices de longitud variable que contienen una secuencia de caracteres. En cualquier punto, la longitud y el contenido de la secuencia se pueden cambiar mediante invocaciones de métodos.

Las cadenas siempre se deben usar a menos que los creadores de cadenas ofrezcan una ventaja en términos de código más simple (consulte el programa de ejemplo al final de esta sección) o un mejor rendimiento. Por ejemplo, si necesita concatenar una gran cantidad de cadenas, anexar a un objeto StringBuilder es más eficiente.

Longitud y capacidad

La clase StringBuilder, al igual que la clase String, tiene un método length () que devuelve la longitud de la secuencia de caracteres en el generador.

A diferencia de las cadenas, cada generador de cadenas también tiene una capacidad, la cantidad de espacios de caracteres que se han asignado. La capacidad, que es devuelta por el método capacity (), siempre es mayor o igual a la longitud (generalmente mayor que) y se expandirá automáticamente según sea necesario para acomodar las adiciones al generador de cadenas.

Constructores

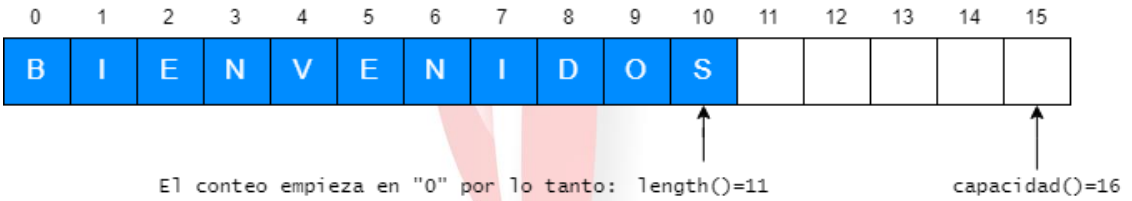
Constructor	Descripción
`StringBuilder()`	Crea un campo String vacío con una capacidad de 16 elementos vacíos.
`StringBuilder(CharSequence cs)`	Construye un String que contiene los mismos caracteres que la clase CharSequence especificada, más un extra de 16 elementos vacíos detrás de la clase CharSequence.
`StringBuilder(int initCapacity)`	Crea un campo String vacío con una capacidad inicial especificada.

StringBuilder(String s)	Crea un campo String vacío cuyo valor inicializa la cadena especificada, más un extra de 16 elementos vacíos detrás de la cadena.

Por ejemplo el siguiente código

```
// crea un constructor vacío, con capacidad de 16 elementos vacios.
StringBuilder sb = new StringBuilder();
// agrega 11 cadena de caracteres en el inicio.
sb.append("Bienvenidos");
```

Construirá una cadena con una longitud de 11 y una capacidad de 16:



Longitud y capacidad de Métodos

Metodos	Descripción
void setLength(int newLength)	Establece la duración de la secuencia de caracteres. Si newLength es menor que length (), los últimos caracteres de la secuencia de caracteres se truncarán. Si newLength es mayor que length (), los caracteres nulos se agregan al final de la secuencia de caracteres.
void ensureCapacity(int minCapacity)	Asegura que la capacidad es al menos igual al mínimo especificado.

Varias operaciones (por ejemplo, append (), insert () o setLength ()) pueden aumentar la longitud de la secuencia de caracteres en el StringBuilder de modo que la longitud resultante () sea mayor que la capacidad actual (). Cuando esto sucede, la capacidad se aumenta automáticamente.

Operaciones de la clase StringBuilder

Las principales operaciones en un StringBuilder que no están disponibles en String son los métodos append () e insert (), que están sobrecargados para aceptar datos de cualquier tipo. Cada uno convierte su argumento en una cadena y luego agrega o inserta los caracteres de esa cadena a la secuencia de caracteres en el generador de cadenas. El método de adición siempre agrega estos caracteres al final de la secuencia de caracteres existente, mientras que el método de inserción agrega los caracteres en un punto específico.

Aquí hay varios métodos de la clase StringBuilder.

Métodos	Descripción
StringBuilder append(boolean b) StringBuilder append(char c) StringBuilder append(char[] str) StringBuilder append(char[] str, int offset, int len) StringBuilder append(double d) StringBuilder append(float f) StringBuilder append(int i) StringBuilder append(long lng) StringBuilder append(Object obj) StringBuilder append(String s)	Añade el argumento a este generador de cadenas. Los datos se convierten en una cadena antes de que tenga lugar la operación de adición.
StringBuilder delete(int start, int end) StringBuilder deleteCharAt(int index)	El primer método elimina la subsecuencia de inicio a fin-1 (inclusive) en la secuencia char de StringBuilder. El segundo método elimina el carácter ubicado en el índice.
StringBuilder insert(int offset, boolean b) StringBuilder insert(int offset, char c) StringBuilder insert(int offset, char[] str) StringBuilder insert(int index, char[] str,	Inserta el segundo argumento en el generador de cadenas. El primer argumento entero indica el

int offset, int len) StringBuilder insert(int offset, double d) StringBuilder insert(int offset, float f) StringBuilder insert(int offset, int i) StringBuilder insert(int offset, long lng) StringBuilder insert(int offset, Object obj) StringBuilder insert(int offset, String s)	índice antes del cual se insertarán los datos. Los datos se convierten en una cadena antes de que se lleve a cabo la operación de inserción.
StringBuilder replace(int start, int end, String s) void setCharAt(int index, char c)	Reemplaza los caracteres especificados en este generador de cadenas.
StringBuilder reverse()	Invierte la secuencia de caracteres en este generador de cuerdas.
String toString()	Devuelve una cadena que contiene la secuencia de caracteres en el generador.

Nota: Puede usar cualquier método String en un objeto StringBuilder convirtiendo primero el generador de cadenas en una cadena con el método toString () de la clase StringBuilder. A continuación, convierta la cadena nuevamente en un generador de cadenas utilizando el constructor StringBuilder (String str).

¿Cuándo Usar?

Los métodos de StringBuilder no son sincronizados, por lo que tiene mejor rendimiento que StringBuffer. En general, la concatenación de String ocurre con variables locales a un método, por lo que es seguro usar StringBuilder en lugar de StringBuffer. En métodos que hacen uso intensivo de la concatenación, la diferente en rendimiento puede ser importante.