

## String.valueOf

---

Este método pertenece no sólo a la clase String, sino a otras y siempre es un método que convierte valores de una clase a otra. En el caso de los objetos String, permite convertir valores que no son de cadena a forma de cadena. Ejemplos:

```
String numero = String.valueOf(1234);  
String fecha = String.valueOf(new Date());
```

En el ejemplo se observa que este método pertenece a la clase String directamente, no hay que utilizar el nombre del objeto creado (como se verá más adelante, es un método estático).

## métodos de las variables de las cadenas

---

Son métodos que poseen las propias variables de cadena. Para utilizarlos basta con poner el nombre del método y sus parámetros después del nombre de la variable String. Es decir: ***variableString.método(argumentos)***

### length

Permite devolver la longitud de una cadena (el número de caracteres de la cadena):

```
String texto1="Prueba";  
System.out.println(texto1.length()); //Escribe 6
```

### concatenar cadenas

Se puede hacer de dos formas, utilizando el método **concat** o con el operador **+**. Ejemplo:

```
String s1="Buenos ", s2="días", s3, s4;  
s3 = s1 + s2;  
s4 = s1.concat(s2);
```

### charAt

Devuelve un carácter de la cadena. El carácter a devolver se indica por su posición (el primer carácter es la posición 0) Si la posición es negativa o sobrepasa el tamaño de la cadena, ocurre un error de ejecución, una excepción tipo **IndexOutOfBoundsException**. Ejemplo:

```
String s1="Prueba";  
char c1=s1.charAt(2); //c1 valdrá 'u'
```

### substring

Da como resultado una porción del texto de la cadena. La porción se toma desde una posición inicial hasta una posición final (sin incluir esa posición final). Si las posiciones indicadas no son válidas ocurre una excepción de tipo **IndexOutOfBoundsException**. Se empieza a contar desde la posición 0. Ejemplo:

```
String s1="Buenos días";
```

```
String s2=s1.substring(7,10); //s2 = día
```

### indexOf

Devuelve la primera posición en la que aparece un determinado texto en la cadena. En el caso de que la cadena buscada no se encuentre, devuelve -1. El texto a buscar puede ser **char** o **String**. Ejemplo:

```
String s1="Quería decirte que quiero que te vayas";  
System.out.println(s1.indexOf("que")); //Da 15
```

Se puede buscar desde una determinada posición. En el ejemplo anterior:

```
System.out.println(s1.indexOf("que",16)); //Ahora da 26
```

### lastIndexOf

Devuelve la última posición en la que aparece un determinado texto en la cadena. Es casi idéntica a la anterior, sólo que busca desde el final. Ejemplo:

```
String s1="Quería decirte que quiero que te vayas";  
System.out.println(s1.lastIndexOf("que")); //Da 26
```

También permite comenzar a buscar desde una determinada posición.

### endsWith

Devuelve **true** si la cadena termina con un determinado texto. Ejemplo:

```
String s1="Quería decirte que quiero que te vayas";  
System.out.println(s1.endsWith("vayas")); //Da true
```

### startsWith

Devuelve **true** si la cadena empieza con un determinado texto.

### replace

Cambia todas las apariciones de un carácter por otro en el texto que se indique y lo almacena como resultado. El texto original no se cambia, por lo que hay que asignar el resultado de **replace** a un **String** para almacenar el texto cambiado:

```
String s1="Mariposa";  
System.out.println(s1.replace('a','e')); //Da Meripose  
System.out.println(s1); //Sigue valiendo Mariposa
```

### replaceAll

Modifica en un texto cada entrada de una cadena por otra y devuelve el resultado. El primer parámetro es el texto que se busca (que puede ser una expresión regular), el segundo parámetro es el texto con el que se reemplaza el buscado. La cadena original no se modifica.

```
String s1="Cazar armadillos";  
System.out.println(s1.replace("ar","er")); //Da Cazer ermedillos  
System.out.println(s1); //Sigue valiendo Cazar armadillos
```

#### toUpperCase

Devuelve la versión en mayúsculas de la cadena.

#### toLowerCase

Devuelve la versión en minúsculas de la cadena.

Los posibles usos de **this** son:

- **this**. Referencia al objeto actual. Se usa por ejemplo pasarle como parámetro a un método cuando es llamado desde la propia clase.
- **this.atributo**. Para acceder a una propiedad del objeto actual.
- **this.método(parámetros)**. Permite llamar a un método del objeto actual con los parámetros indicados.
- **this(parámetros)**. Permite llamar a un constructor del objeto actual. Esta llamada sólo puede ser empleada en la primera línea de un constructor.

**Actividad.**

Leer el documento y aplicar cada uno de los métodos en Netbens con sus propios ejemplos

3 Ejemplos por cada método.