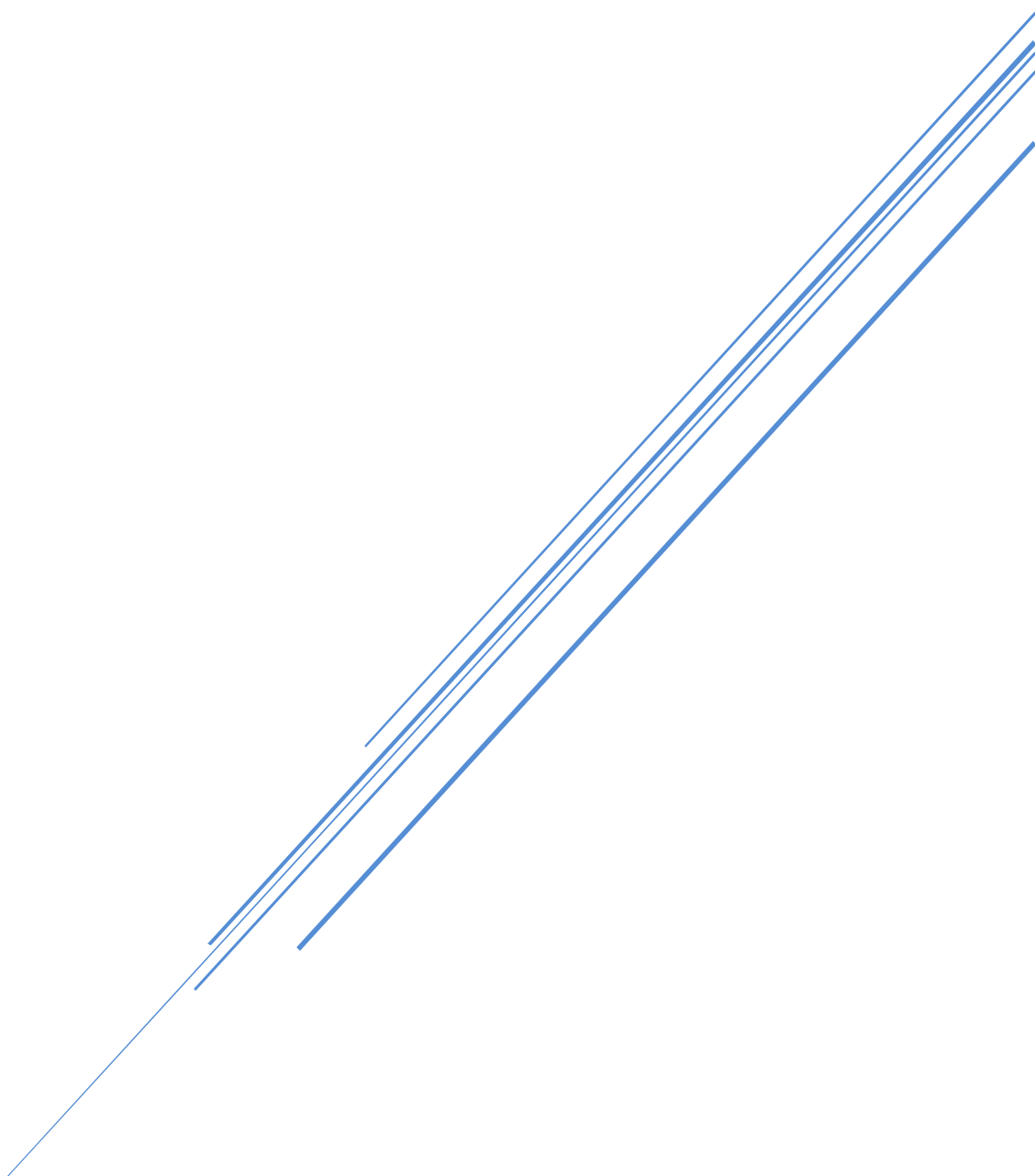


TOKENS

PROGRAMACIÓN



[Escuela]
[Título del curso]

Clase Java StringTokenizer

La clase Java StringTokenizer sirve para separar una cadena de caracteres en una serie de elementos o **tokens**.

Se incluye en el paquete java.util

Los tokens se separan mediante delimitadores.

Los delimitadores por defecto son:

- espacio en blanco
- tabulador \t
- salto de línea \n
- retorno \r
- avance de página \f

Un objeto StringTokenizer se construye a partir de un objeto String.

Para obtener los tokens del String podemos utilizar los métodos `hasMoreTokens()` y `nextToken()`.

hasMoreTokens() devuelve true si hay más tokens que obtener en la cadena.

nextToken() devuelve un String con el siguiente token. Lanza una excepción del tipo `NoSuchElementException` si no hay más tokens.

Ejemplo de uso de StringTokenizer:

```
String s = "blanco, rojo, verde y azul";
StringTokenizer st = new StringTokenizer(s);
while (st.hasMoreTokens())
    System.out.println(st.nextToken());
```

La salida que se obtiene por pantalla es:

```
blanco,
rojo,
verde
y
azul
```

Se ha separado el String s en tokens separados por el delimitador por defecto. En este caso el espacio en blanco. Si lo representamos de forma gráfica, el funcionamiento es el siguiente:

La instrucción

```
StringTokenizer st = new StringTokenizer(s);
```

Produce lo siguiente:

Se separa el String en tokens y el objeto st *apunta* al primero.

A continuación

```
st.hasMoreTokens()
```

comprueba si hay tokens que extraer. En este caso st apunta a un token por lo tanto `hasMoreTokens()` devuelve true.

A continuación

st →	Tokens
	blanco,
	rojo,
	verde
	y
	azul

st →	Tokens
	blanco,
	rojo,
	verde
	y
	azul

st.nextToken()

Obtiene el token al que apunta st (blanco,) y avanza al siguiente.

El ciclo while repite el proceso.

Cuando se alcanza el último token y se avanza el siguiente la condición del while será false.

Los **delimitadores** se pueden especificar cuando se crea el objeto StringTokenizer. Por ejemplo, para indicar que los delimitadores son la coma y el espacio en blanco: `StringTokenizer st = new StringTokenizer("colores rojo, verde y azul", ", ");`
La ejecución del while anterior obtendría la salida:

```
colores
rojo
verde
y
azul
```

la coma no aparece ya que se ha especificado como delimitador y los delimitadores no aparecen.

Si queremos que aparezcan se debe escribir true como tercer argumento en el constructor:

```
StringTokenizer st = new StringTokenizer("colores rojo, verde y azul", ", ", true);
```

En este caso la salida es:

```
colores
rojo
,
verde
y
azul
```

Referencias

<http://docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html>

- [java.lang.Object](#)
 - - java.util.StringTokenizer

Constructors

Constructor and Description

[StringTokenizer](#)([String](#) str) Constructs a string tokenizer for the specified string.

[StringTokenizer](#)([String](#) str, [String](#) delim) Constructs a string tokenizer for the specified string.

[StringTokenizer](#)([String](#) str, [String](#) delim, boolean returnDelims)
Constructs a string tokenizer for the specified string.

Methods

Modifier and Type

Method and Description

	<u>countTokens</u> ()
int	Calculates the number of times that this tokenizer's <code>nextToken</code> method can be called before it generates an exception.
	<u>hasMoreElements</u> ()
boolean	Returns the same value as the <code>hasMoreTokens</code> method.
	<u>hasMoreTokens</u> ()
boolean	Tests if there are more tokens available from this tokenizer's string.
	<u>nextElement</u> ()
<u>Object</u>	Returns the same value as the <code>nextToken</code> method, except that its declared return value is <code>Object</code> rather than <code>String</code> .
	<u>nextToken</u> ()
<u>String</u>	Returns the next token from this string tokenizer.
	<u>nextToken</u> (<u>String</u> delim)
<u>String</u>	Returns the next token in this string tokenizer's string.