APRENDERAPROGRAMAR.COM

DIVIDIR CADENAS O SEPARAR PALABRAS EN JAVA. CLASE STRINGTOKENIZER. COUNTTOKENS. EJEMPLOS RESUELTOS. (CU00923C)

Sección: Cursos
Categoría: Lenguaje de programación Java nivel avanzado I

Fecha revisión: 2029

Resumen: Entrega nº23 curso "Lenguaje de programación Java Nivel Avanzado I".

Autor: Manuel Sierra

INTRODUCCIÓN

Todos alguna vez nos hemos encontrado con la necesidad de partir una cadena de caracteres en pequeños fragmentos a los que en programación se suele denominar "tokens". La clase StringTokenizer nos permite esta funcionalidad de una manera rápida y eficiente y por eso está en el paquete java.util. A continuación veremos cómo podemos utilizar esta clase.

STRINGTOKENIZER

La clase StringTokenizer es bastante sencilla pero no por ello poco útil. Al crear un objeto podemos especificar o no el carácter que queremos usar para separar la cadena en varias subcadenas, a través del constructor. La clase tiene 3 constructores. Si usamos el constructor donde simplemente pasamos una cadena sin especificar el carácter delimitador, por defecto se separarán las subcadenas usando como delimitadores los caracteres (" \t\n\r\f") que son respectivamente espacio en blanco (" "), tabulador (\t), salto de línea (\n), retorno a principio de línea (\r) o comienzo de página (\f). Este constructor es de la forma StringTokenizer st = new StringTokenizer ("Aquí la frase o cadena de texto");

Ahora bien podemos pasar un segundo argumento en el constructor por ejemplo si queremos o deseamos que nuestro separador sea por ejemplo ";" podemos especificarlo para que la separación se produzca en base a este carácter. Así una cadena de ejemplo como "Un día; Dos Días; Tres Días" si construyéramos un objeto de la clase StringTokenizer con la anterior cadena como primer argumento y le pasáramos como segundo argumento en el constructor el ";", la cadena quedaría subdivida en tres partes ya que existen tres separadores ;. Si pasamos como separador más de un carácter, cada uno de los caracteres indicados se considera un separador. Por ejemplo ";=" hará que tanto ; como = sean considerados separadores.

El tercer argumento que se puede pasar en el constructor es un valor booleano que indica si queremos que se nos devuelva los caracteres delimitadores o no como tokens. En el ejemplo anterior los ; no serían incluidos como tokens, simplemente se entiende que son separadores y no aparecen en los resultados. En cambio con esta opción podrían ser devueltos como tokens que constaran de un solo carácter. Esto que puede parecer no necesario, es útil cuando estamos formateando un string a tokens pero donde contamos con varios delimitadores y tenemos que saber cuál es el delimitador para poder interpretar los resultados obtenidos. Así por ejemplo podemos tener la cadena "Ronaldo=Fútbol; Gasol=Baloncesto; Nadal=Tenis; Alonso=F1;" y querer separar usando los caracteres = y ;.

En este caso puede ser necesario la devolución de los caracteres delimitadores para saber si el siguiente token es un nuevo deporte (si el token extraído es un = lo que sigue es un deporte como Fútbol, Baloncesto, Tenis, F1) o un nuevo deportista (si el token extraído es un ; lo que sigue es un deportista como Ronaldo, Gasol, Nadal, Alonso).

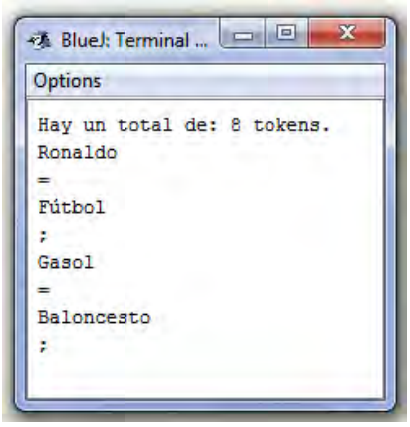
EJEMPLO DE USO DE STRINGTOKENIZER

En el ejemplo que veremos a continuación comprobaremos cómo se puede usar la clase StringTokenizer para dividir una cadena String en tokens. Escribe este código en tu editor y ejecútalo:

```
/* Ejemplo Clase StringTokenizer aprenderaprogramar.com */
import java.util.StringTokenizer;
public class Programa
{
   public static void main (String []args)
   {
      StringTokenizer st = new StringTokenizer("Ronaldo=Fútbol;Gasol=Baloncesto;","=;",true);
      System.out.println("Hay un total de: "+st.countTokens()+" tokens.");
      while (st.hasMoreTokens()) {
         System.out.println(st.nextToken());
      }
   }
}
```

En este caso no hay diagrama de clases, ya que tan solo hemos usado nuestra clase principal Programa.

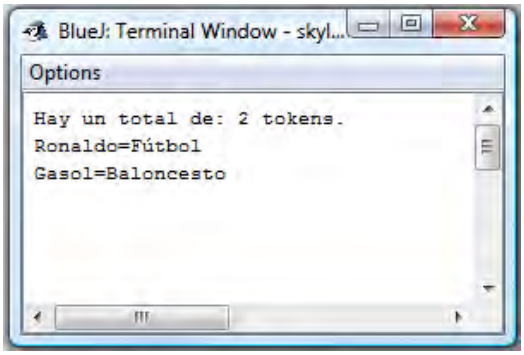
El resultado de ejecución del programa nos devuelve la siguiente salida:



Como podemos observar en la salida, imprimimos el número total de tokens que tenemos. Para ello nos valemos de los métodos hasMoreTokens() y nextToken(), similares a los de otras clases que ya conocemos y que nos devuelven respectivamente un booleano indicador de si existen más tokens y el siguiente token.

En este caso el número de tokens es 8 ya que en el constructor hemos definido se han de usar como delimitadores los caracteres = y ; así como que nos cuente los delimitadores como tokens. Si esto estuviera a false (que es el valor por defecto) entonces tan solo tendríamos 4 tokens que serían Ronaldo, Fútbol, Gasol y Baloncesto.

Prueba ahora a ejecutar el mismo código pero en lugar de indicar dos delimitadores indica solo uno, por ejemplo el ;. El resultado es que tendremos como tokens Ronaldo=Fútbol, ; , Gasol=Baloncesto, ; . Si omitimos el booleano true en el constructor tendríamos solo dos tokens ya que se omitirían los ;.



CONCLUSIONES

Hemos visto lo fácil que es partir una cadena de caracteres con la clase StringTokenizer y como esta trabaja. Comprobamos también que opcionalmente podemos obtener los delimitadores, función muy útil cuando trabajamos con múltiples delimitadores y necesitamos saber cuál es el delimitador que ha dado lugar a la separación. Normalmente cuando hacemos una separación en tokens será para almacenar el resultado de alguna manera (en nuevos Strings, en una colección de Strings, en un array de Strings, en una base de datos...).

Próxima entrega: CU00924C

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente: http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=58&Itemid=180