10.A. Introducción al almacenamiento de datos y flujos.

4. Clases relativas a flujos.

4.1. Ejemplo comentado de una clase con flujos.

Vamos a ver un ejemplo con una de las clases comentadas, en concreto, con `StreamTokenizer`.

La clase `StreamTokenizer` obtiene un flujo de entrada y lo divide en "tokens". El flujo tokenizer puede reconocer identificadores, números y otras cadenas.

El ejemplo que puedes ver a continuación, muestra cómo utilizar la clase `StreamTokenizer` para contar números y palabras de un fichero de texto. Se abre el flujo con ayuda de la clase `FileReader`, y puedes ver cómo se "monta" el flujo `StreamTokenizer` sobre el `FileReader`, es decir, que se construye el objeto StreamTokenizer con el flujo FileReader como argumento, y entonces se empieza a iterar sobre él.

```

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package tokenizer;

import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.StreamTokenizer;


public class token {

    public void contarPalabrasyNumeros(String nombre_fichero) {

        StreamTokenizer sTokenizer = null;
        int contapal = 0, numNumeros = 0;

        try {

            sTokenizer = new StreamTokenizer(new FileReader(nombre_fichero));

            while (sTokenizer.nextToken() != StreamTokenizer.TT_EOF) {

                if (sTokenizer.ttype == StreamTokenizer.TT_WORD)
                    contapal++;
                else if (sTokenizer.ttype == StreamTokenizer.TT_NUMBER)
```

```
                    numNumeros++;
            }

            System.out.println("Número de palabras en el fichero: " + contapal);
            System.out.println("Número de números en el fichero: " + numNumeros);

        } catch (FileNotFoundException ex) {
           System.out.println(ex.getMessage()) ;
        } catch (IOException ex) {
           System.out.println(ex.getMessage()) ;
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        new Main().countWordsAndNumbers("c:\\datos.txt");
    }
}
```

El método `nextToken` devuelve un int que indica el tipo de token leído. Hay una serie de constantes definidas para determinar el tipo de token:

- `TT_WORD` indica que el token es una palabra.
- `TT_NUMBER` indica que el token es un número.
- `TT_EOL` indica que se ha leído el fin de línea.
- `TT_EOF` indica que se ha llegado al fin del flujo de entrada.

En el código de la clase, apreciamos que se iterará hasta llegar al fin del fichero. Para cada token, se mira su tipo, y según el tipo se incrementa el contador de palabras o de números.

Autoevaluación

Indica si la siguiente afirmación es verdadera o falsa.

Según el sistema operativo que utilicemos, habrá que utilizar un flujo u otro. ¿Verdadero o Falso?

- ○ Verdadero.
- ○ Falso.