Ficheros de Texto (Escritura)

- La forma más cómoda de generar y leer ficheros de texto es mediante las clases PrintWriter y Scanner.
- La clase java.io.PrintWriter:
  - Escribe representaciones formateadas de tipos primitivos a un fichero de tipo texto.

Método / Constructor	Descripción
PrintWriter(File file)	Crea un nuevo PrintWriter a partir de un File
PrintWriter(OutputStream out)	Crea un nuevo PrintWriter a partir de un stream
void **print**(float f)	Escribe un float (y no termina la línea)
void **println**(float f)	Escribe un float (y añade un salto de línea)
void **print**(boolean b)	Escribe un boolean (y no termina la línea)
…	…
boolean **checkError**()	Comprueba si ha habido error en la escritura anterior

La Clase PrintWriter

Ejemplos alternativos de creación de un PrintWriter:

- PrintWriter pw = new **PrintWriter**(new File("/tmp/f.txt"));

  Si el fichero no existe se crea uno nuevo, si existe se trunca su tamaño a cero.

- PrintWriter pw2 = new **PrintWriter**(
  new **FileOutputStream**("/tmp/f.txt", **true**));

  Si el fichero no existe se crea uno nuevo, si existe se mantiene, ya que se escribirá añadiendo (**append**) al final del mismo.

- Si el fichero especificado no es accesible en el sistema de archivos, el constructor de PrintWriter o de FileOutputStream puede lanzar la excepción *FileNotFoundException*, que deberá ser tratada.

Uso de la Clase PrintWriter

- Ejemplo de programa que usa PrintWriter:

```
import java.io.*;
import java.util.Scanner;
public class TestPrintWriter {
  public static void main(String[] args){
    String fichero = "file2.txt";
    try {
        PrintWriter pw = new PrintWriter(new File(fichero));
        pw.print("El veloz murciélago hindú");
        pw.println(" comía feliz cardillo y kiwi");
        pw.println(4.815162342);
        pw.close();
     } catch (FileNotFoundException e) { System.err.println("Problemas al abrir el fichero " + fichero); }
  }
}
```

- El programa escribe en el fichero:

  El veloz murciélago hindú comía feliz cardillo y kiwi
  4.815162342

Ficheros de Texto (Lectura)

- La clase java.io.Scanner:
  - Escáner de texto que permite procesar tipos primitivos y cadenas.
  - Divide los datos de entrada en tokens (elementos individuales) que serán procesados de forma individual.

Método / Constructor	Descripción
Scanner(File source)	Crea un nuevo Scanner a partir de un File
Scanner(String src)	Crea un nuevo Scanner para leer datos a partir del String
boolean **hasNext**()	Devuelve true si hay al menos un token en la entrada
boolean **hasNextInt**()	Devuelve true si el siguiente token de la entrada es un entero
int **nextInt**()	Devuelve el siguiente token de la entrada como un entero. De no serlo, lanza InputMismatchException
…	…
String **next**()	Devuelve el siguiente token de la entrada
String **nextLine**()	Devuelve el resto de la línea, desde el punto de lectura actual
void **close**()	Cierra el Scanner

Ficheros de Texto (Lectura). Excepciones

- Si el fichero especificado no está accesible en el sistema de archivos, el constructor de Scanner puede lanzar la excepción (checked) *FileNotFoundException*.
  - Esta excepción deberá ser tratada.

- Algunos de los métodos de lectura como nextInt(), nextDouble(), etc, pueden lanzar una excepción (unchecked) de formato incorrecto: *InputMismatchException* o, si se intenta acceder más allá del final del fichero se provocará la *NoSuchElementException*.
  - El tratamiento de estas excepciones permitirá recuperar el proceso si se intenta leer un dato mal formateado o inexistente.

Ejemplo de Ficheros de Texto

- Programa que escribe 10 enteros en un fichero de texto y luego los lee antes de mostrarlos por pantalla.

```
import java.io.*;
import java.util.Scanner;
class TestPrintWriter1{
  public static void main(String[] args){
    String fichero = "file1.txt";
    try {
            PrintWriter pw = new PrintWriter(new File(fichero));
            for (int i = 0; i < 10 ; i++)  pw.println(i);
            pw.close();
            Scanner scanner = new Scanner(new File(fichero));
            while (scanner.hasNext()) { System.out.println("Valor leido: "+scanner.nextInt()); }
            scanner.close();
    } catch (FileNotFoundException e) {
            System.err.println("Problemas al abrir el fichero " + fichero);
    }
 } }
```

Ejemplo de uso de Scanner (I)

```
import java.io.*;  import java.util.Scanner;
public class TestScanner {
  public static void main(String[] args){
    System.out.println("Leeremos 3 números y una línea de texto");
    Scanner scanner = null;
    try{
        scanner = new Scanner(new File("cosas.txt"));
    }catch(FileNotFoundException ex){
        System.err.println("El fichero no existe." + ex.getMessage());   System.exit(0);
    }
    int n1 = scanner.nextInt();  int n2 = scanner.nextInt();  int n3 = scanner.nextInt();
    scanner.nextLine();
    String linea = scanner.nextLine();
    System.out.println("Los tres números son: " + n1 + ", " + n2 + ", " + n3);
    System.out.println("La línea es: " + linea);
    scanner.close();
  }
}
```

cosas.txt

```
1 2
3 4
Multiplícate por cero!
```

Ejemplo de uso de Scanner (II)

```
import java.io.*;
import java.util.Scanner;
public class TestScannerWhile {
   public static void main (String[] args) {
    try {
       Scanner scanner = new Scanner(new File("carreras.txt"));
       while (scanner.hasNextLine()){
          String linea = scanner.nextLine();
          String[] tokens = linea.split("\t");
          System.out.println(tokens[0] + " : " + tokens[1]);
       }
    } catch (FileNotFoundException ex) {
       System.err.println("El fichero no existe." + ex);
    }
   }}
```

carreras.txt

Juan Lopez	09:34:25
Paco Martínez	09:38:25
Jane Doe	10:38:25

- Lectura de todas las líneas de un fichero de texto donde cada valor está separado por un tabulador (\t).

Sobre el Uso de Scanner

- Scanner no leerá el fichero de texto si contiene algún acento o carácter fuera del ASCII ya que estará guardado con una codificación diferente a ASCII (probablemente UTF-8).
  - No obtendrás ningún error pero no se leerán los datos.
- La solución pasa por indicar a Scanner la codificación empleada para guardar el fichero de texto.
  - Tipicamente dependiente del sistema operativo, la codificación suele ser UTF-8 (OS X, Linux) o ISO-8859-1 (Windows), aunque es el editor de ficheros el que puede elegir una codificación diferente

```
Scanner scanner = new Scanner(new FileInputStream("carreras.txt"), "UTF-8");
```