**CLASE BUFFEREDREADER & BUFFEREDWRITER:**

* **ENLACES WEB**

<http://ayudaitver.blogspot.com.es/2014/07/uso-basico-del-bufferedreader.html#ubicacion>

<https://www.discoduroderoer.es/clases-bufferedreader-y-bufferedwriter-para-ficheros-de-texto-en-java/>

* **CUTRE TRADUCCIÓN:**

BufferedReader (buffered = amortiguado, reader = lector)

BufferedWriter (buffered = amortiguado, writer = escritor)

* **DEFINICIÓN:**

Estas clases tienen la misma función que FileReader y FileWriter, leer y escribir en ficheros, pero BufferedReader y BufferedWriter optimizan estas funciones.

BufferedReader es una clase cuyas instancias nos permiten hacer lecturas sencillas de texto desde un flujo de caracteres, debido a que esta clase trabaja con su propio buffer provee una lectura eficiente de caracteres, arreglos y líneas de texto, es decir, nos permite leer una línea completa, en lugar ir carácter por carácter como hacia **FileReader,** cuando el fichero termina, devuelve **null** y no un -1 como en **FileReader.**

Con BufferedWriter también podemos añadir una línea, como si pulsáramos un Enter.

Otra de las grandes ventajas de BufferedReader y BufferedWriter es que esta clase está sincronizada lo cual es sinónimo de seguridad al utilizarla en programación concurrente.

* [**UBICACIÓN**](http://ayudaitver.blogspot.com.es/2014/07/uso-basico-del-bufferedreader.html#ubicacion)

Las clases **BufferedReader y BufferedWriter** las podemos encontrar en **java.io.**

También os aconsejamos que importéis **java.io.IOException** puesto que al manejar BufferedReader Y BufferedWriter manejamos FLUJOS por lo que se pueden presentar situaciones extraordinarias o también llamadas Excepciones un ejemplo de esto es que el flujo que estamos utilizando se cierre o sea modificado por un agente externo a nuestro programa.

Adelantamos de que también deberemos importara tanto **java.io.FileReader** y **java.io.InputStreamReader** como **java.io.FileWriter** y **java.io.InputStreamWriter** pero eso lo aclararemos después.

* [**CREACIÓN DE UNA INSTANCIA**](http://ayudaitver.blogspot.com.es/2014/07/uso-basico-del-bufferedreader.html#creacion)

Se crean igual que **FileReader y FileWriter**, pero como parámetro insertaremos un objeto **FileReader** para **BufferedReader** y un objeto **FileWriter** para **BufferedWriter**.

*EJEMPLOS:*

FileWriter fw = new FileWriter("*Dirección del fichero*");

FileReader fr = new FileReader("*Dirección del fichero*");

BufferedReader br = new BufferedReader(fr);

BufferedWriter bw = new BufferedWriter(fw);

*Y DE UNA FORMA MÁS ABREVIADA:*

BufferedReader br = new BufferedReader(new FileReader("*Dirección del fichero*"));

BufferedWriter bw = new BufferedWriter(new FileWriter("*Dirección del fichero*"));

* [**MÉTODOS**](http://ayudaitver.blogspot.com.es/2014/07/uso-basico-del-bufferedreader.html#metodos)

|  |  |  |
| --- | --- | --- |
| **Método** | **Tipo de Retorno** | **Descripción** |
| mark(int limiteCaracteres) | void | Marca la posición actual en la que se encuentre el apuntador en el flujo, el parámetro del método indica el número de caracteres que pueden ser leídos mientras se mantenga esta marca. |
| read() | void | Lee un solo carácter del flujo, lo curioso de este método es que retorna un número entero, ¿Por qué? Más adelante la respuesta. |
| readLine() | String | Lee una línea completa de texto. |
| ready() | boolean | Este método es utilizado para saber si aún hay caracteres en el flujo para ser leídos, detalles más adelante. |
| reset() | void | Reinicia el flujo hasta la marca más reciente que se haya hecho. |
| skip(long n) | long | Mueve el apuntador del flujo las posiciones necesarios para evitar la cantidad de caracteres de n. |

|  |  |  |
| --- | --- | --- |
| **Método** | **Tipo de Retorno** | **Descripción** |
| close() | void | Cierra la corriente. Una vez que la secuencia se ha cerradono puedes ni usar write () ni flush () pues saltará una excepción de IOException. |
| newLine() | Void | Escribe un separador de línea. La cadena del separador de línea está definida por la propiedad del sistema line.separator, y no es necesariamente un carácter de nueva línea ('\ n'). |
| write([String](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html" \o "class in java.lang) s, int off, int len) | void | Escribe una porción de una cadena. Si el valor del parámetro “len” es negativo, entonces no escribe. Utiliza la excepción IndexOutOfBoundsException. |
| write(int c) | void | Escribe un solo carácter. |
| flush() | void | ¿? |

* [**EJEMPLOS**](http://ayudaitver.blogspot.com.es/2014/07/uso-basico-del-bufferedreader.html#ejemplos)
  + **Lista de ejemplos que debemos exponer.**
* **API:**

<https://docs.oracle.com/javase/8/docs/api/index.html?java/io/BufferedReader.html>

<https://docs.oracle.com/javase/7/docs/api/java/io/BufferedWriter.html>

TAREAS:

* Pensar ejemplos e implementarlos
* Buscar métodos de bufferWriter - HECHO
* Hacer Power Point