

### Clase MiObjectOutputStream.java

/\*problema en el **ObjectOutputStream** es que al instanciarlo, escribe unos bytes de cabecera en el fichero, antes incluso de que escribamos nada. Como el **ObjectInputStream** lee correctamente estos bytes de cabecera, aparentemente no pasa nada y ni siquiera nos enteramos que existen. El problema se presenta si escribimos unos datos en el fichero y lo cerramos. Luego volvemos a abrirlo para añadir datos, creando un nuevo **ObjectOutputStream** así. El true del final indica que se abre el fichero para añadir datos al final del fichero.

```
ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(fichero,true));
```

Esto escribe una nueva cabecera justo al final del fichero. Luego se irán añadiendo los objetos que vayamos escribiendo. El fichero como mínimo dos cabeceras (según las veces que lo instanciamos).

¿Qué pasa cuando leamos el fichero?. Al crear el **ObjectInputStream**, este lee la cabecera del principio y luego se pone a leer objetos. Cuando llegamos a la segunda cabecera que se añadió al abrir por segunda vez el fichero para añadirle datos, obtendremos un error **StreamCorruptedException** y no podremos leer más objetos.

La única solución que he encontrado (que seguramente no es la única) es hacernos nuestro propio **ObjectOutputStream**,\* heredando del original y redefiniendo el método **writeStreamHeader()** como en la figura, vacío, para que no haga nada. \*/

```
package ejMenu;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.OutputStream;

/**
 * Redefinición de la clase ObjectOutputStream para que no escriba una
cabecera
 * al inicio del Stream.
 * */
public class MiObjectOutputStream extends ObjectOutputStream{
    /** Constructor que recibe OutputStream */
    public MiObjectOutputStream(OutputStream out) throws IOException
    {
        super(out);
    }

    /** Constructor sin parámetros */
    protected MiObjectOutputStream() throws IOException, SecurityException
    {
        super();
    }

    /** Redefinición del método de escribir la cabecera para que no haga nada.
    */
    @Override
    protected void writeStreamHeader() throws IOException
    {
    }
}
```