10.E. Almacenamiento de objetos en ficheros. Serialización.

Tabla de contenidos

1 Almacenamiento de objetos en ficheros. Serialización.

¿Qué es la **serialización**? Es un proceso por el que **un objeto se convierte en una secuencia de bytes** con la que más tarde se podrá reconstruir el valor de sus variables. Esto permite guardar un objeto en un archivo.

Para serializar un objeto:

- éste debe **implementar el interface** `java.io.Serializable`. Este interface no tiene métodos, sólo se usa para informar a la JVM (Java Virtual Machine) que un objeto va a ser serializado.
- Todos los objetos incluidos en él tienen que implementar el interfaz Serializable.

Todos los **tipos primitivos en Java son serializables** por defecto. (Al igual que los arrays y otros muchos tipos estándar).

Para leer y escribir objetos serializables a un stream se utilizan las clases java: `ObjectInputStream` y `ObjectOutputStream`.

En el siguiente ejemplo se puede ver cómo leer un objeto serializado que se guardó antes. En este caso, se trata de un String serializado:

```
FileInputStream fich = new FileInputStream(“str.out”);
```

```
ObjectInputStream os = new ObjectInputStream(fich);
```

```
Object o = os.readObject();
```

Así vemos que `readObject` lee un objeto desde el flujo de entrada fich. Cuando se leen objetos desde un flujo, se debe tener en cuenta qué tipo de objetos se esperan en el flujo, y se han de leer en el mismo orden en que se guardaron.

Autoevaluación

Indica si es verdadera o falsa la siguiente afirmación:

Para serializar un objeto basta con que la clase, a la que pertenece dicho objeto, implemente el interface Serializable. ¿Verdadero o falso?

- ○ Verdadero.
- ○ Falso.

1.1 Serialización: utilidad.

La serialización en Java se desarrolló para utilizarse con RMI. RMI necesitaba un modo de convertir los parámetros necesarios a enviar a un objeto en una máquina remota, y también para devolver valores desde ella, en forma de flujos de bytes. Para datos primitivos es fácil, pero para objetos más complejos no tanto, y ese mecanismo es precisamente lo que proporciona la serialización.

El método `writeObject` se utiliza para guardar un objeto a través de un flujo de salida. El objeto pasado a `writeObject` debe implementar el interfaz `Serializable`.

```
FileOutputStream fisal = new FileOutputStream(“cadenas.out”);
```

```
ObjectOutputStream oos = new ObjectOutputStream(fisal);
```

```
Oos.writeObject();
```

La serialización de objetos se emplea también en la arquitectura de componentes software JavaBean. Las clases bean se cargan en herramientas de construcción de software visual, como NetBeans. Con la paleta de diseño se puede personalizar el bean asignando fuentes, tamaños, texto y otras propiedades.

Una vez que se ha personalizado el bean, para guardarlo, se emplea la serialización: se almacena el objeto con el valor de sus campos en un fichero con extensión .ser, que suele emplazarse dentro de un fichero .jar.