
 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 7</i>	

Fitxers Serialitzats

Així com a Kanto tenen el PC de Bill per a guardar objectes i Mokepon, al Mallès tenim el Fitxer de Bill per a fer el mateix. Anem a implementar una forma de Desar Objectes i Mokepon al Fitxer de Bill.

Una forma de fer-ho seria guardar totes les estadístiques en un fitxer de text, i després llegir-ho i reconstruir el Mokepon, però això seria lent, tediós, i a més ocuparia. No obstant, hi ha una forma de fer-ho que és mitjançant la Serialització, a on pots desar objectes sencers, en binari.

Escriure en fitxers Serialitzats



L'exemple general per a escriure en fitxers serialitzats és el següent. Com pots veure, és gairebé idèntic al de fitxers de Text

```
try {
    File f = new File("objectes.dat");
    //funciona de forma similar a un fileWriter, amb append inclòs
    FileOutputStream fos = new FileOutputStream(f);
    ObjectOutputStream oos = new ObjectOutputStream(fos);
    Pocio p = new Pocio("SuperPocio", 100);
    oos.writeObject(p);
    oos.flush();
    oos.close();
} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}}}
```

A partir de l'exemple, crea el mètode **afegirObjecte**(Objecte obj) que afegirà un objecte.

Com potser hauràs notat de l'error, només pots afegir classes que implementin l'interfície Serializable. Afegeix Serializable a Objecte. No cal implementar cap mètode. Això sí, només funcionarà amb classes que els seus atributs siguin relativament senzills, res de coses estranyes com Threads, etc.

Igualment, tots els seus atributs han d'implementar també Serializable. Afortunadament la majoria de classes estàndards (com els primitius, Strings, Llistes) ho implementen.

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 7</i>	

Al igual que amb el FileWriter, pots obrir el FileOutputStream en mode append de la següent forma.

```
fos = new FileOutputStream(f, true);
```

PERO COMPTE. HI HA UN BUG GREU AMB EL OBJECTOUTPUTSTREAM EN MODE APPEND.



Bàsicament, escriu una capçalera cada cop que el crees. Això fa que si l'obres en mode append, afegix la capçalera en mig del fitxer i el deixa corrupte. Per evitar-ho pots fer servir aquesta classe derivada en comptes d'ObjectOutputStream

```
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.OutputStream;

public class AppendableObjectOutputStream extends ObjectOutputStream {
    private boolean append;
    private boolean initialized;
    private DataOutputStream dout;

    public AppendableObjectOutputStream(OutputStream out, boolean append)
throws IOException {
        super(out);
        this.append = append;
        this.initialized = true;
        this.dout = new DataOutputStream(out);
        this.writeStreamHeader();
    }

    @Override
    protected void writeStreamHeader() throws IOException {
        if (!this.initialized || this.append) return;
        if (dout != null) {
            dout.writeShort(STREAM_MAGIC);
            dout.writeShort(STREAM_VERSION);
        }
    }
}
```

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 7</i>	

Llegir en fitxers Serialitzats

L'exemple general per a llegir fitxers serialitzats és el següent, assumint que l'objecte és una Poció

```
try {
    File f = new File("objectes.dat");
    FileInputStream fis = new FileInputStream(f);
    ObjectInputStream ois = new ObjectInputStream(fis);

    Pocio p = (Pocio) ois.readObject();
    System.out.println(p);

} catch (FileNotFoundException e) {
    System.out.println("no existeix el fitxer");
    e.printStackTrace();
} catch (IOException e) {
    System.out.println("excepció d'entrada/sortida");
    e.printStackTrace();
} catch (ClassNotFoundException e) {
    System.out.println("no s'ha trobat la classe demanada");
    e.printStackTrace();
}
```



El problema es que no necessàriament rebràs una poció. També pots rebre un Reviure, o un altre objecte. Degut a que `readObject` torna un objecte, per saber a què s'ha de castejar, hauràs de saber a quina classe pertany mitjançant **instanceof**

Fes les següents funcions

recuperaObjecte(). Retorna el primer objecte trobat

Potser t'has adonat que en `ObjectInputStream` no tens un `ready()` que et permeti saber quan sortir del bucle, en cas de llegir un nombre desconegut d'objectes. Hi ha dos formes de tractar-ho

- `ObjectInputStream.available()` torna el nombre de bytes que queden per llegir. Si és més gran que 0, significa que encara no s'ha acabat de llegir. **EN ALGUNS CASOS MOLT ESPECÍFICS POT NO FUNCIONAR**
- llegir quan no queda res dispara una `EOFException`. Es pot fer un bucle infinit [`while(true)`] dintre d'un `try/catch` i captura l'excepció `EOFException`.

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	Moképon - Part 7	

Fes la següent funció

recuperaObjectes(). Retorna una llista amb tots els objectes

recuperaPocioConcreta(int n). Retorna una poció que curi exactament n. Si no hi ha cap al fitxer retorna null

Modificar fitxers serialitzats

Modificar fitxers serialitzats es realitza de la mateixa forma que modificar fitxers de text: Llegir el fitxer, escriure a un altre fitxer temporal, i després esborrar el fitxer original i canviar-li el nom al fitxer temporal.

Recordeu que per fer això teniu accés als mètodes de la classe File. especialment file.delete() i file.renameTo(File f). També assegureu-vos de tancar tots els readers i writers que pengen d'un file abans de fer un f.delete()

Implementa el següent.

afegeixMokepon(MokeponCapturat mok). Afegeix un Mokepon al fitxer si no està ja.

Tingues en compte dues coses

- Recorda que MokeponCapturat ha de ser Serializable, i tots els atributs que contingui també!
- Hauràs de llegir tot el fitxer i comparar cada objecte per saber si és el mateix MokeponCapturat que intentes afegir. Recorda que anteriorment vas implementar el mètode equals a MokeponCapturat i et pot servir

"Fes un equals a Mokepon i a MokeponCapturat. dos Mokepon seran iguals si tenen les mateixes estadístiques (excepte hp_actual, i debilitat, ja que seria el mateix Mokepon, però ferit). És a dir, haurien de tenir iguals nom, sexe, nivell, tipus, atk, def, vel, hp_max. Dos MokeponCapturats seran iguals si, apart dels paràmetres de Mokepon, tenen igual entrenador i nomDonat."

recuperarMokepon(String nom, int nivell, String entrenador, String nomDonat). Recupera un MokeponCapturat amb aquestes dades. Llegirà tot el fitxer, i si troba un Mokepon amb aquestes dades l'esborrarà i el retornarà.

teamMocketAtacaDeNou() Canvia el nom d'entrenador de tots els Mokepon a "Team Mocket". No fa res amb els objectes