Un fitxer binari o de dades està format per seqüències de bytes. Aquests arxius poden contenirdades de tipus bàsic (int, float, char, etc) i objectes. Per poder llegir el contingut d'un fitxer binari, cal conèixer l'estructura interna d'aquest, és a dir, hem de saber com ha estat escrit, en quin ordre. Si no es coneix la seva estructura es pot llegir byte a byte.

Escriure dades en fitxers binaris

Per escriure dades en un fitxer binari cal utilitzar les classes `FileOutputStream` i `DataOutputStream` derivades d'`OutputStream`.

1. FileOutputStream

La classe `FileOutputStream` permet tenir accés al fitxer per escriure bytes. Per crear objectes `FileOutputStream` podem utilitzar els següents constructors:

```
    FileOutputStream (String ruta)
    FileOutputStream (File objetoFile);
    FileOutputStream (String ruta, boolean append)
    FileOutputStream (File objetoFile, boolean append)
```

Si el paràmetre `append` és cert significa que les dades s'afegiran a les ja existents. Contràriament, si és fals, les dades existents es perden, així com si s'utilitza qualsevol dels dos primers constructors. Els constructors poden llençar una excepció `FileNotFoundException` en cas que no existeixi el fitxer, o bé no s'hagi pogut crear.

La classe `FileOutputStream` proporciona el mètode `write()` per poder escriure bytes al fitxer. Aquest mètode pot llençar una excepció `IOException`.

2. DataOutputStream

A partir d'un objecte `FileOutputStream` es pot crear un objecte `DataOutputStream`, que proporciona mètodes per escriure dades de tipus primitiu al fitxer. Per crear un objecte `DataOutputStream` s'utilitza el constructor:

```
    DataOutputStream(OutputStream nombre);
```

Aquesta classe proporciona el mètode `writeXxx()`, on Xxx és el nom de tipus rimitiu. Pot generar una `IOException`.

3. Exemple

Programa que llegeix sencers per teclat i els escriu al fitxer dades.dat. La lectura de les dades finalitza quan s'introdueix -1.

```
    import java.io.DataOutputStream;
```

```
    import java.io.FileNotFoundException;
    import java.io.FileOutputStream;
    import java.io.IOException;
    import java.util.Scanner;

    public class Binarios1 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        FileOutputStream fos = null;
        DataOutputStream salida = null;
        int num;
        try {
            fos = new FileOutputStream("/ficheros/datos.dat");
            salida = new DataOutputStream(fos);
            System.out.print("Introduce número entero. -1 para acabar:
");
            num = sc.nextInt();
            while (num != -1) {
                salida.writeInt(num); //se escribe el número entero en
el fichero
                System.out.print("Introduce número entero. -1 para
acabar: ");
            num = sc.nextInt();
            }
        }catch(FileNotFoundException e){
            System.out.println(e.getMessage());
        }catch(IOException e){
            System.out.println(e.getMessage());
        }finally{
            try{
                if(fos!=null)
                    fos.close();
                if(salida!=null)
                    salida.close();
            }catch(IOException e){
                System.out.println(e.getMessage());
            }
        }
    }
}
```

Lectura de fitxers binaris

Per llegir d'un fitxer binari utilitzarem les classes `FileInputStream` i `DataInputStream`, ambdues derivades d'`InputStream`.

1. FileInputStream

La classe `FileInputStream` permet llegir bytes d'un fitxer. Per crear objectes FileOutputStream podem utilitzar els següents constructors:

```
FileInputStream (String ruta)
FileInputStream (File objetoFile);
```

Ambdós poden llençar una excepció `FileNotFoundException` si el fitxer no existeix, així com el mètode `read()`, que serveix per llegir bytes del fitxer, llença una excepció `IOExcepion`.

2. DataInputStream

A partir d'un objecte FileInputStream podem crear un objecte La classe `DataInputStream` per poder llegir dades de tipus primitiu. Per crear un objecte DataInputStream s'utilitza el constructor:

```
DataInputStream (inputStream nombre);
```

La classe proporciona mètodes `readXxx()` on Xxx és el nom de tipus primitiu. Aquests mètodes llencen excepcions `IOException`, i en el cas d'arribar al final del fitxer llencen una `EOFException`.

3. Exemple

Programa de Java que llegeix el contingut del fitxer creat a l'exemple anterior. S'utilitzarà una iteració de la que es sortirà al llençar l'excepció EOFException.

```
    import java.io.DataInputStream;
    import java.io.EOFException;
    import java.io.FileInputStream;
    import java.io.FileNotFoundException;
    import java.io.IOException;

    public class Binarios3 {

    public static void main(String[] args) {
        FileInputStream fis = null;
        DataInputStream entrada = null;
        int num;
        try {
            fis = new FileInputStream("/ficheros/datos.dat");
            entrada = new DataInputStream(fis);
            while (true) {
                num=entrada.readInt();  //se lee  un int del fichero
                System.out.println(num);  //se muestra en pantalla
            }
        } catch (FileNotFoundException e) {
            System.out.println(e.getMessage());
        } catch (EOFException e) {
            System.out.println("Fin de fichero");
        } catch (IOException e) {
            System.out.println(e.getMessage());
        } finally {
            try {
                if (fis != null) {
                    fis.close();
                }
                if (entrada != null) {
                    entrada.close();
                }
            } catch (IOException e) {
                System.out.println(e.getMessage());
```

```
            }
        }
    }
}
```