Podstawy programowania

Laboratorium 11 - pliki binarne, serializacja i obiektowość

mgr inż. Krzysztof Szwarc

krzysztof@szwarc.net.pl

Sosnowiec, 1 czerwca 2020

Klasy do obsługi plików binarnych

Zapis pliku

1. FileOutputStream (operuje na bajtach).
2. OutputStreamWriter (dostosowany do różnych systemów kodowania znaków).

Przykład

```
public static void main(String[] args)
    throws IOException {
        String file = "D:\\binarnie.txt";
        FileOutputStream writer= new
            FileOutputStream(file);
        writer.write("test".getBytes());
        writer.flush();
        writer.close();
}
```

Przykład 2

```
public static void main(String[] args)
    throws IOException {
        String file = "D:\\binarnie.txt";
        OutputStreamWriter writer = new
            OutputStreamWriter(new
            FileOutputStream(file));
        writer.write("test");
        writer.flush();
        writer.close();
}
```

Klasy do obsługi plików binarnych

Odczyt pliku

1. FileInputStream (operuje na bajtach).
2. InputStreamReader (dostosowany do różnych systemów kodowania znaków).

Przykład

```
public static void main(String[] args)
    throws IOException {
        String file = "D:\\binarnie.txt";
        FileInputStream reader = new
            FileInputStream(file);
        int readChar = 0;
        while
            ((readChar=reader.read())!=-1) {
             System.out.print((char)readChar);
        }
        reader.close();
}
```

Przykład 2

```
public static void main(String[] args)
    throws IOException {
      String file = "D:\\binarnie.txt";
      InputStreamReader reader = new
         InputStreamReader(new
         FileInputStream(plik));
      int readChar = 0;
      while ((readChar =
         reader.read())!=-1) {
          System.out.print((char)readChar);
      }
      reader.close();
}
```

Klasy pomagające w przetwarzaniu plików binarnych

Pliki binarne

1. DataInputStream.
2. DataOutputStream.

Ciekawostka: metoda writeUTF służąca do zapisu łańcucha znaków dopisuje dwa bajty na początku tekstu.

Przykład

```
public static void main(String[] args)
    throws IOException {
        String file = "D:\\binarnie.txt";
        DataOutputStream writer = new
            DataOutputStream(new
            FileOutputStream(file));
        writer.writeUTF("test");
        writer.writeInt(7);
        writer.flush();
        writer.close();
}
```

Przykład 2

```
public static void main(String[] args)
    throws IOException {
      String file = "D:\\binarnie.txt";
      DataInputStream  reader = new
         DataInputStream (new
         FileInputStream(file));
      String test = reader.readUTF();
      int number = reader.readInt();
      reader.close();
}
```

Uwaga! Musimy znać kolejność zapisanych danych.

Przykład 3

```
public static void main(String[] args)
    throws IOException {
      String file = "D:\\binarnie.txt";
      DataInputStream  reader = new
         DataInputStream (new
         FileInputStream(file));
      int number = reader.readInt(); //
         ERROR
      String test = reader.readUTF();
      reader.close();
}
```

Zadania

1. Zapisz plik z wykorzystaniem DataOutputStream zapisujący dowolną linijkę tekstu oraz liczbę zmiennoprzecinkową. Następnie sprawdź zawartość stworzonego pliku.
2. Odczytaj utworzony w zadaniu 1 plik z wykorzystaniem klasy DataInputStream.

Jednoczesny zapis i odczyt z pliku

Opis

Klasa RandomAccessFile wykorzystywana jest do jednoczesnego zapisywania i odczytywania z tego samego pliku. W jego konstruktorze podajemy ścieżkę do pliku oraz prawa dostępu do pliku (rw lub r). Wskaźnik w pliku nie jest oddzielny, więc chcąc zapisać dane i je odczytać musimy przesunąć go na ich początek korzystając z metody seek.

Przykład

```
public static void main(String[] args)
    throws IOException {
      String file = "D:\\plik.txt";
      RandomAccessFile stream = new
         RandomAccessFile(file, "rw");
      stream.writeUTF("s");
      stream.seek(0);
      System.out.println(stream.readUTF());
      stream.close();
}
```

Przegląd najważniejszych metod

Metoda	Opis
seek(pozycja)	Przesuwa wskaźnik na określoną pozycję
getFilePointer()	Zwraca aktualną pozycję wskaźnika
length()	Zwraca wielkość pliku

Przykład 2

```
public static void main(String[] args) throws
    IOException {
     String file = "D:\\plik.txt";
     RandomAccessFile raf = new
         RandomAccessFile(file, "rw");
     while (raf.getFilePointer()<raf.length()) {
         String name = raf.readUTF();
         long position = raf.getFilePointer();
         int number = raf.readInt();
         if (number<0) {
             raf.seek(position);
             raf.writeInt(0);
         }
     }
     raf.close();
}
```

Zadanie

1. Napisz metodę zapisującą do pliku tekst „Ala ma dużego kota” oraz liczbę typu całkowitego 6. Następnie z tego samego pliku niech będzie odczytywana wyłącznie wpisana liczba, bez zamykania pliku i z wykorzystaniem jednego strumienia. Użyj klasy RandomAccessFile i uzasadnij zaproponowane rozwiązanie.

Serializacja

Opis

Serializacja jest procesem konwersji obiektów na strumień bajtów (z zachowaniem aktualnego stanu). Aby skorzystać z serializacji w Javie musimy zaimplementować interfejs Serializable.

Klasy wykorzystywane w serializacji

Serializacja

1. ObjectInputStream.
2. ObjectOutputStream.

Przykład

```
class Employee implements
    java.io.Serializable {}
class Lesson {
 public static void main(String[] args)
     throws IOException {
     String file = "D:\\plik.txt";
     ObjectOutputStream stream = new
        ObjectOutputStream(new
        FileOutputStream(file));
     Employee object = new Employee();
     stream.writeObject(object);
     stream.flush();
     stream.close();
 }
}
```

Przykład

```
class Lesson {
 public static void main(String[] args)
    throws IOException,
    ClassNotFoundException {
    String file = "D:\\plik.txt";
    ObjectInputStream stream = new
       ObjectInputStream(new
       FileInputStream(file));
    Employee object =
       (Employee)stream.readObject();
    stream.close();
 }
}
```

Zadania

1. Utwórz klasę „Osoba” z polem imię, która implementuje interfejs „Serializable”. Następnie stwórz jej pięć obiektów z dowolnymi imionami, po czym zapisz je do pliku korzystając z klasy ObjectOutputStream.
2. Odczytaj zapisane obiekty i przypisz je do tablicy z wykorzystaniem klasy ObjectInputStream.

Dziękuję za uwagę