Podstawy i języki programowania

Laboratorium 10 - pliki binarne, serializacja i obiektowość

mgr inż. Krzysztof Szwarc

krzysztof@szwarc.net.pl

Sosnowiec, 8 stycznia 2018

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 plik = "D:\\binarnie.txt";
        FileOutputStream zapis= new
            FileOutputStream(plik);
        zapis.write("test".getBytes());
        zapis.flush();
        zapis.close();
}
```

Przykład 2

```
public static void main(String[] args)
    throws IOException
{
        String plik = "D:\\binarnie.txt";
        OutputStreamWriter zapis = new
            OutputStreamWriter(new
            FileOutputStream(plik));
        zapis.write("test");
        zapis.flush();
        zapis.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 plik = "D:\\binarnie.txt";
      FileInputStream odczyt = new
         FileInputStream(plik);
      int odczytane = 0;
      while ((odczytane=
         odczyt.read())!=-1)
      {
          System.out.print((char)odczytane);
      }
      odczyt.close();
}
```

Przykład 2

```
public static void main(String[] args)
    throws IOException
{
      String plik = "D:\\binarnie.txt";
      InputStreamReader odczyt = new
         InputStreamReader(new
         FileInputStream(plik));
      int odczytane = 0;
      while ((odczytane=
         odczyt.read())!=-1)
      {
           System.out.print((char)odczytane);
      }
      odczyt.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 plik = "D:\\binarnie.txt";
        DataOutputStream  zapis = new
            DataOutputStream (new
            FileOutputStream(plik));
        zapis.writeUTF("test");
        zapis.writeInt(7);
        zapis.flush();
        zapis.close();
}
```

Przykład 2

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

Uwaga! Musimy znać kolejność zapisanych danych.

Przykład 3

```
public static void main(String[] args)
    throws IOException
{
      String plik = "D:\\binarnie.txt";
      DataInputStream  odczyt = new
         DataInputStream (new
         FileInputStream(plik));
      int liczba = odczyt.readInt(); //
         BŁĄD
      String test = odczyt.readUTF();
      odczyt.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 plik = "D:\\plik.txt";
      RandomAccessFile strumien = new
         RandomAccessFile(plik, "rw");
      strumien.writeUTF("s");
      strumien.seek(0);
      System.out.println(strumien.readUTF());
      strumien.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 plik = "D:\\plik.txt";
    RandomAccessFile raf = new
        RandomAccessFile(plik, "rw");
    while (raf.getFilePointer()<raf.length())
    {
        String imie = raf.readUTF();
        long pozycja = raf.getFilePointer();
        int liczba = raf.readInt();
        if (liczba<0)
        {
            raf.seek(pozycja);
            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 Pracownik implements
   java.io.Serializable {}
class Zajecia
{
 public static void main(String[] args)
    throws IOException
 {
    String plik = "D:\\plik.txt";
    ObjectOutputStream strumien = new
       ObjectOutputStream(new
       FileOutputStream(plik));
    Pracownik obiekt = new Pracownik();
    strumien.writeObject(obiekt);
    strumien.flush();
    strumien.close();
 }}
```

Przykład

```
class Zajecia
{
 public static void main(String[] args)
    throws IOException,
    ClassNotFoundException
 {
    String plik = "D:\\plik.txt";
    ObjectInputStream strumien = new
       ObjectInputStream(new
       FileInputStream(plik));
    Pracownik obiekt =
       (Pracownik)strumien.readObject();
    strumien.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