ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Οντοκεντρικός Προγραμματισμός

Ενότητα 3: JAVA: ΕΞΑΙΡΕΣΕΙΣ, ΕΙΣΟΔΟΣ-ΕΞΟΔΟΣ, ΝΗΜΑΤΑ

Είσοδος - Έξοδος

ΔΙΔΑΣΚΟΝΤΕΣ: Ιωάννης Χατζηλυγερούδης, Χρήστος Μακρής

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

ΕΙΣΟΔΟΣ-ΕΞΟΔΟΣ ΔΕΔΟΜΕΝΩΝ

ΕΙΣΟΔΟΣ-ΕΞΟΔΟΣ ΔΕΔΟΜΕΝΩΝ

Στην java οι πληροφορίες αποθηκεύονται και ανακαλούνται/ ανασύρονται με τη χρήση ενός συστήματος επικοινωνίας που χρησιμοποιεί την έννοια του **stream** (κανάλι επικοινωνίας).

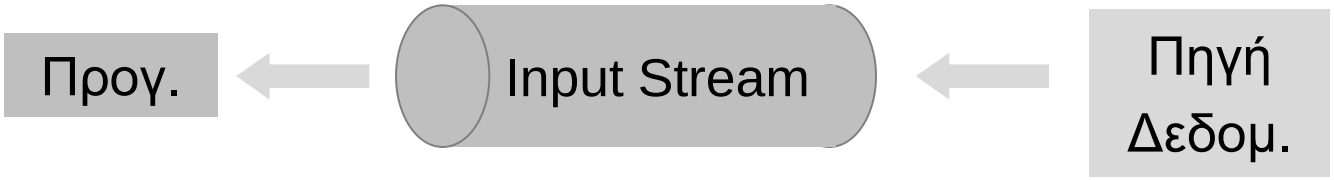
Σαν stream ορίζεται μια «διαδρομή» μέσα από την οποία μεταφέρονται δεδομένα από μια θέση σε μια άλλη. Κάθε stream είναι ένα αντικείμενο (στιγμιότυπο) java.

Η πληροφορία που μεταφέρεται μέσα από ένα stream είναι μια ακολουθία δεδομένων που έχουν μια πηγή ή ένα αποδέκτη.

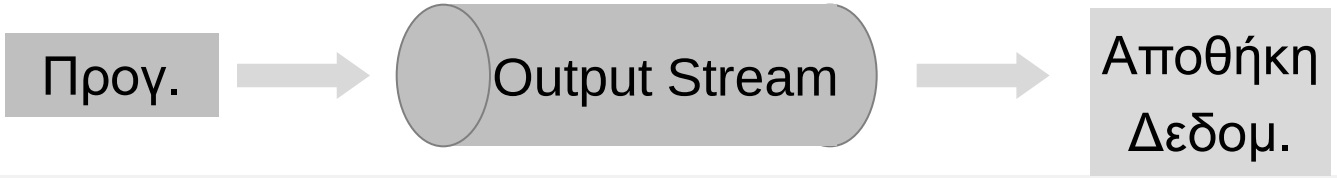
ΤΥΠΟΙ STREAM (1)

Με βάση την κατεύθυνση

- input streams (για ανάγνωση πληροφορίας, δηλ. αποστολή δεδομένων από μια εξωτερική πηγή στο πρόγραμμα)



- output streams (για αποθήκευση πληροφορίας, δηλ. αποστολή δεδομένων από το πρόγραμμα σ' ενα εξωτερικό προορισμό)



ΤΥΠΟΙ STREAM (2)

Με βάση τον τύπο της πληροφορίας

- byte streams(για μεταφορά δεδομένων πρωτογενών τύπων ή αλφαριθμητικών της java)

- character streams (για μεταφορά χαρα κτήρων/κειμένου από αρχεία που τα παριστάνουν μέσω κωδικοποιήσεων ASCII ή Unicode)

ΚΛΑΣΕΙΣ STREAM (I/O)

Περιέχονται στο πακέτο `java.io`

Για byte streams

`InputStream, OutputStream (abstract)`

`FileInputStream, FileOutputStream`

`BufferedInputStream, BufferedOutputStream`

`DataInputStream, DataOutputStream`

Για character streams

`Reader, Writer (abstract)`

`FileReader, FileWriter`

`BufferedReader, BufferedWriter`

ΚΛΑΣΕΙΣ STREAM (I/O)

```
Object
 InputStream
   FileInputStream
   FilterInputStream
     BufferedInputStream
     DataInputStream
 OutputStream
   FileOutputStream
   FilterOutputStream
     BufferedOutputStream
     DataOutputStream
 RandomAccessFile
 File
```

ΚΥΡΙΑ ΕΙΣΟΔΟΣ-ΕΞΟΔΟΣ(1)

Με την εκτέλεση κάθε προγράμματος (εφαρμογής) δημιουργούνται τρία (3) (στιγμιότυπα) streams:

- **`System.in`** (κύρια είσοδος)
- **`System.out`** (κύρια έξοδος)
- `System.err` (κύριο σφάλμα)

ΚΥΡΙΑ ΕΙΣΟΔΟΣ-ΕΞΟΔΟΣ(2)

- **Βασική έξοδος:**

```
System.out.println(String s);        (έξοδος string)
System.out.write(int x);             (έξοδος ενός byte)
System.out.write(byte b[]);          (έξοδος ενός αριθμού bytes από πίνακα)
```

(Μέθοδοι της OutputStream)

- **Βασική είσοδος:** (Επιστρέφει έναν απρόσημο ακέραιο 0-255 ή -1)

```
System.in.read();                    (είσοδος ενός byte)
System.in.read(byte b[]);            (είσοδος ενός αριθμού bytes σε πίνακα)
```

(Μέθοδοι της InputStream)

Όλες, πλην της println(…):
"throws IOException"

ΠΑΡΑΔΕΙΓΜΑ (1)

```
import java.io.*;
public class Inpdata{
    public static void main(){
      char b='0';
      try {
        System.out.println("Give a character: ");
        if ((b = (char)System.in.read()) != '\n')
            System.out.println(b);
      }
      catch(IOException e){
        System.out.println(e);
      }
    }
}
```

ΠΑΡΑΔΕΙΓΜΑ (2)

```
import java.io.*;
public class MInpdata1{
 public static void main(){
   char b='0';
     try {
            System.out.println("Give a sequence of characters: ");
      for (int i=0; (b=(char)System.in.read()) != '\n'; i++)
        System.out.println(b);
     }
     catch(IOException e){
       System.out.println(e);
     }
 }
}
```

ΠΑΡΑΔΕΙΓΜΑ (3)

```
import java.io.*;
public class MInpdata2{
    public static void main()  {
      byte [] b = new byte[10];
      try {
        System.out.println("Write a string (max length=10): ");
        System.in.read(b);
        String s = new String(b);
        System.out.println(s);
      }
      catch(IOException e) {
        System.out.println(e);
      }
    }
}
```

ΠΑΡΑΔΕΙΓΜΑ (4)

```
import java.io.*;
public class MInpdata3{
    public static void main() {
      byte [] b = new byte[10];
      try {
        System.out.println("Write a string (max length=10): ");
        System.in.read(b);
        System.out.write(b);
      }
      catch(IOException e){
        System.out.println(e);
      }
    }
}
```

ΔΙΑΔΙΚΑΣΙΑ ΕΙΣΟΔΟΥ

1. Άνοιγμα/Δημιουργία ενός stream (π.χ. προκειμένου για αρχείο, δημιουργούμε ένα στιγμιότυπο της FileInputStream και το συσχετίζουμε με το αρχείο εισόδου)
2. Ανάγνωση δεδομένων, ενόσω υπάρχουν, από την πηγή εισόδου (π.χ. με τη βοήθεια μεθόδων της FileInputStream, όπως η read())
3. Κλείσιμο του stream (π.χ. χρήση της μεθόδου close())

ΔΙΑΔΙΚΑΣΙΑ ΕΞΟΔΟΥ

1. Άνοιγμα/Δημιουργία ενός stream (π.χ. προκειμένου για αρχείο, δημιουργούμε ένα στιγμιότυπο της FileOutputStream και το συσχετίζουμε με το αρχείο εξόδου)
2. Αποστολή δεδομένων, ενόσω υπάρχουν, στον προορισμό εξόδου (π.χ. με τη βοήθεια μεθόδων της FileOutputStream, όπως η write())
3. Κλείσιμο του stream (π.χ. χρήση της μεθόδου close())

ΦΙΛΤΡΑΡΙΣΜΑ STREAM

Ορισμός

Φίλτρο είναι ένας τύπος stream που τροποποιεί τον τρόπο χειρισμού ενός άλλου stream.

Διαδικασία

1. Δημιουργία ενός stream (εισόδου ή εξόδου).
2. Συσχέτιση φίλτρου με το stream.
3. Ανάγνωση/αποστολή δεδομένων από/προς το φίλτρο (αντί του stream).

BYTE STREAMS ΑΡΧΕΙΩΝ (1)

Είναι στιγμιότυπα των FileInputStream και FileOutputStream, που είναι υποκλάσεις των InputStream και OutputStream αντίστοιχα.

(Για να μπορούμε να διαβάσουμε δεδομένα από ή να γράψουμε δεδομένα σε ένα αρχείο πρέπει να το συσχετίσουμε μ' ένα stream εισόδου ή εξόδου αντίστοιχα.)

Εισόδου

Δημιουργία: **`FileInputStream(String fname)`**

Ανάγνωση: **`read()`** (επόμενο byte: ακέραιος ή –1)

`read(byte[], int, int)` (αριθ. Bytes ή –1)

θέση 1ου byte

αριθμός bytes

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class ReadFile{
   public static void main(){
   try {FileInputStream fis1 = new FileInputStream("test.dat");
         boolean eof = false;
         int count = 0;
         while (!eof) {
           int inp = fis1.read();
           System.out.print(inp + " ");
           if (inp = = -1)
              eof = true;
           else count++;}
         fis1.close();
         System.out.println("\nBytes read: " + count);}
      catch(IOException e)
       {System.out.println(e);}
   }
}
```

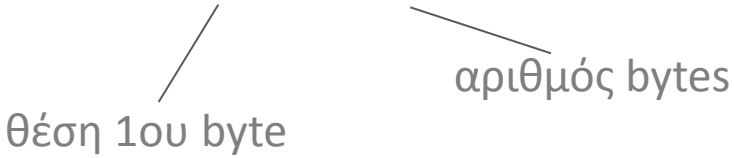
BYTE STREAMS ΑΡΧΕΙΩΝ (2)

Εξόδου

Δημιουργία: **`FileOutputStream(String fname)`**

Εγγραφή: **`write(int)`** (εγγραφή byte)

`write(byte[], int, int)` (εγγραφή πολλών bytes)



ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class WriteFile{
   public static void main(){
     int [] data = {71, 73, 65, 0, 56, 33, 18, 22, 0, 0, 250, 178, 4, 2, 0, 0, 65, 0}
     try {FileOutputStream fos1 = new FileOutputStream("testo.dat");
         for (int i=0; i < data.length; i++)
             fos1.write(data[i]);
         fos1.close();
     }
     catch(IOException e) {
        System.out.println(e);
     }
   }
}
```

BUFFERED (ΕΝΤΑΜΙΕΥΜΕΝΑ) STREAMS

Για αποδοτικότερη χρήση, χρησιμοποιούν ένα buffer (ενταμιευτή) για ενδιάμεση αποθήκευση δεδομένων.

- **Εξόδου**

  Δημιουργία: `BufferedOutputStream(OutputStream)`
  `BufferedOutputStream(OutputStream, int)`

  Εγγραφή: `write(int),write(byte[], int, int)`

  Άδειασμα buffer: `flush()`

- **Εισόδου**

  Δημιουργία: `BufferedInputStream(InputStream)`
  `BufferedInputStream(InputStream, int)`

  Ανάγνωση: `read(), read (byte[], int, int)`

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class CopyFile{
  public static void main(String args []){
   try {FileInputStream fis = new FileInputStream(args[0]);
        BufferedInputStream bis = new BufferedInputStream(fis);
        FileOutputStream fos = new FileOutputStream(args[1]);
        BufferedOutputStream bos = new BufferedOutputStream(fos);
        int nbytes;
        while (bis.available() > 0) {
           nbytes = bis.read();
           bos.write(nbytes); }
        bis.close(); bos.flush(); bos.close();
      }
      catch(IOException e)
       {System.out.println(e);}
   }
}
```

BYTE STREAMS ΔΕΔΟΜΕΝΩΝ

Για να εργαστούμε με δεδομένα που δεν παριστάνονται σαν ακολουθίες bytes, αλλά σαν λογικά τμήματα.

- **Εξόδου**

  Δημιουργία: `DataOutputStream(OutputStream)`

  Εγγραφή: `writeDouble(double),writeFloat(float) writeInt(int),writeLong(long),writeShort(int), writeByte (int), writeBoolean(boolean)`

- **Εισόδου**

  Δημιουργία: `DataInputStream(InputStream)`

  Ανάγνωση: `readDouble(),readFloat(),readInt(),readLong(), readShort(), readByte (), readBoolean()`

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class WriteFile{
   public static void main(){
   double [] data = {71.2, 73.1, 65.0, 0.0, 56.4, 33.4, 18.7, 22.0, 0, 0,
                                                  250.0, 178.3, 4.2, 2.1, 0, 0, 6.35, 0};
   try {FileOutputStream fos1 = new FileOutputStream("testr.dat");
         DataOutputStream fods1 = new DataOutputStream(fos1);
         for (int i=0; i < data.length; i++)
              fods1.writeDouble(data[i]);
         fods1.close();
    }
    catch(IOException e)
      {System.out.println(e);}
   }
}
```

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class ReadFile{
  public static void main(){
    try { FileInputStream fis1 = new FileInputStream("testr.dat");
          DataInputStream fids1 = new DataInputStream(fis1);
          int count = 0;
           try {   while (true) {
                      double inp = fids1.readDouble();
                      System.out.print(inp + " \n");
                    count++;}  }
            catch(EOFException eof)
                 {fids1.close();}
          System.out.println("\nNums read: " + count);}
     catch(IOException e)
        {System.out.println("Error-- " + e.toString());}
  }
}
```

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class WritePrimeNums{
  public static void main(String args []){
      int num = 2;
      try { FileOutputStream fis = new FileOutputStream("primes.dat");
          BufferdOutputStream bos = new BufferedInputStream(fis);
          DataOutputStream dos = new DataOutputStream(bos);
          while (num < 400) {
            if (isPrime(num))
              dos.writInt(num);
            num++;
          }
          dos.close();}
      catch(IOException e)
          {System.out.println(e);}
   }
}
```

ΠΑΡΑΔΕΙΓΜΑ

```
import java.io.*;
public class ReadPrimeNums{
  public static void main(String args []){
    try {DataInputStream dos = new DataInputStream(
                                new BufferedInputStream(
                                new FileOutputStream("primes.dat")));
      try {while (true) {
            int num = dos.readInt();
            System.out.println(num + " ");}
      }
      catch (EOFException eof) {}
      finally {dos.close()};
    }
    catch(IOException e)
      {System.out.println("Error: " + e.toString() );}
  }
}
```

CHARACTER STREAMS ΑΡΧΕΙΩΝ (1)

Είναι στιγμιότυπα των FileReader και FileWriter, που είναι υποκλάσεις των InputStreamReader και OutputStreamWriter αντίστοιχα.

- **Εισόδου**

  Δημιουργία: `FileReader(String fname)`

  Ανάγνωση: `read()`

  `read(char[], int, int)`

  αριθμός χαρακτήρων

  θέση 1ου χαρακτήρα

CHARACTER STREAMS ΑΡΧΕΙΩΝ (2)

Για ανάγνωση ολόκληρης γραμμής, αντί χαρακτήρα.

Δημιουργία: `BufferedReader(Reader)`

`BufferedReader(Reader, int)`

Ανάγνωση: `read(), read (byte[], int, int)`

`readLine()` (επιστρέφει ένα String, που είναι μια γραμμή κειμένου)

ΔΙΑΧΕΙΡΙΣΗ ΑΡΧΕΙΩΝ ΑΜΕΣΗΣ ΠΡΟΣΠΕΛΑΣΗΣ (1)

Μέχρι τώρα αναφερόμαστε σε αρχεία σειριακής προσπέλασης (για να βρούμε μια πληροφορία πρέπει να περάσουμε από όλες τις προηγούμενες)

Μπορούμε όμως να δημιουργήσουμε και αρχεία τυχαίας προσπέλασης (random access), όπου μπορούμε να μεταβαίνουμε σε ένα τμήμα πληροφορίας χωρίς να είναι απαραίτητο να περάσουμε απ' όλα τα προηγούμενα.

ΔΙΑΧΕΙΡΙΣΗ ΑΡΧΕΙΩΝ ΑΜΕΣΗΣ ΠΡΟΣΠΕΛΑΣΗΣ (2)

Η δημιουργία και διαχείριση αρχείων τυχαίας προσπέλασης γίνεται μέσω της κλάσης RandomAccessFile.

Δημιουργοί: (και οι δύο throws FileNotFoundException)

```
RandomAccessFile (String name, String mode)
RandomAccessFile (File name, String mode)
```

αντικείμενο αρχείου (βλ. αργότερα)

Κανονίζει τον τρόπο ανοίγματος, π.χ. “r” (για ανάγνωση μόνο), “rw” (για ανάγνωση και εγγραφή)

ΔΙΑΧΕΙΡΙΣΗ ΑΡΧΕΙΩΝ ΑΜΕΣΗΣ ΠΡΟΣΠΕΛΑΣΗΣ (3)

Οι μέθοδοι ανάγνωσης και εγγραφής είναι οι ίδιες με αυτές των byte streams αρχείων και δεδομένων.

Ένα χαρακτηριστικό των αρχείων τυχαίας προσπέλασης είναι ότι τα δεδομένα τους διαβάζονται και γράφονται αρχίζοντας όχι απαραίτητα από την αρχή, αλλά από κάποια θέση του δείκτη που καθορίζουμε. Αυτό γίνεται με τη μέθοδο (public void):

`seek (long index)` **`(η οποία throws IOException)`**

Μετακινεί τον δείκτη ακριβώς πριν από το (index+1) byte του αρχείου. Το πρώτο byte ενός αρχείου είναι στη θέση 0.

ΔΙΑΧΕΙΡΙΣΗ ΑΡΧΕΙΩΝ ΑΜΕΣΗΣ ΠΡΟΣΠΕΛΑΣΗΣ (4)

Οι εντολές/μέθοδοι ανάγνωσης και εγγραφής ενεργούν πάντα στην τρέχουσα θέση του δείκτη στο αρχείο. Η εκτέλεσή τους έχει σαν αποτέλεσμα τη μετακίνηση του δείκτη στην επόμενη θέση.

Η RandomAccessFile παρέχει και άλλες χρήσιμες μεθόδους, όπως:

```
public long getFilePointer()
```
Επιστρέφει την τιμή του δείκτη.

```
public long length()
```
Επιστρέφει το μήκος του αρχείου σε bytes.

```
public void setlength(long newlen)
```
Κάνει το μήκος του αρχείου ίσο με newlen.

(Όλες `throws IOException`)

ΑΝΤΙΚΕΙΜΕΝΑ ΑΡΧΕΙΩΝ(1)

Τα αντικείμενα αρχείων αντιπροσωπεύουν συγκεκριμένα αρχεία ή καταλόγους και χρησιμοποιούνται όχι για ανάγνωση ή εγγραφή, αλλά για τη διαχείριση πληροφοριών σχετικών με τα «φυσικά» χαρακτηριστικά αρχείων ή καταλόγων.

Δημιουργούνται ως στιγμιότυπα της κλάσης File (3 δημιουργοί):

```
File (String name)

File (String path, String name)

File (File directory, String name)
```

ΑΝΤΙΚΕΙΜΕΝΑ ΑΡΧΕΙΩΝ(2)

Παράδειγμα

File myDir = new File("c:\\ihatz\\java"); (1ος δημ.)
File myFile = new File(myDir, "test.dat"); (3ος δημ.)
Οι δύο αυτές προτάσεις μπορούν να συνοψιστούν:
File myFile=new File("c:\\ihatz\\java", "test.dat") (2ος δημ.)

Η File παρέχει πολλές μεθόδους:

```
exists(),    isFile(),    isDirectory(),    isAbsolute(),
canRead(),   canWrite(),  delete()  (όλες τύπου boolean)

getName(), getParent(), getPath()   (όλες τύπου String)
```

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.1

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Σημείωμα Χρήσης Έργων Τρίτων