የጃቫ ገቢና ወጪ ምንጮች (Java IO Streams)

10.1 ምዕላደ-ቃላት

አማርኛ ቃል	እንግሊዘኛ ቃል
ሆሄ/ሆሄያት	character/chararcters
ተጠቃሚ	user
ገቢ	input
ወጪ	output
ምንጭ	stream
ኪቦርድ*	keyboard
ገጽ	screen
ፋይል*	file
የፋይል ጠቋሚ	file pointer
የዴታ ዓይነታት	data types
የመስመር ጫፍ	end of line
አዲስ መስመር	new line
የፋይል መጨረሻ	end of file
ኧክሴፕሽን*	exception

10.2 ምንጮች

በጃቫ እንዴት አድርገን ነው ዴታ ከኪቦርድ ፣ ከፋይሎች ፣ ወይም ከሌሎች ምንጮች የምናነበው? በተጨማሪ ወደ ፋይሎች ፣ ወደ ገጾች ፣ ወይም ወደ ሌሎች ምንጮች ዴታ እንዴት እንጽፋለን? ባለፉት ምዕራፎች የቀረቡት አንዳንድ ምሳሌዎች እነዚህን ጥያቄዎች በጨረፍታ ይነካሉ ፤ ነገር ግን በቂ መልስ አይሰጡም። በዚህ ክፍል መልሶቹን ለመልከት እንሞክራለን።

ምንጭ **(stream)** ዴታዎች ወይም ነገሮች የሚመጡበት ወይም የሚወጡበት አድረገን መገንዘብ እንችላለን። ተጠቃሚዎች ኪቦርድ ተጠቅመው ዴታዎችን ወይም ነገሮችን ካስገቡ ፣ ኪቦርዱን የገቢ ምንጭ **(input stream)** ብለን እንጠራዋለን። እንደ ኪቦርድ ተመሳሳይ ተግባር ያላቸው የኮምፕዩተር መሣሪያዎችን በሙሉ በዚሁ ዘይቤ እንጠቅሳቸዋለን።

ባንጻሩ የወጪ ምንጭ **(output stream)** ብለን የምንጠራቸው የኮምፕዩተር አካሎችና መሣሪያዎች አሉ። ፋይሎች ፣ የሞኒተር ገጾች አብዮቹ ናቸው።

የምንጭ ቃል አጠቃቀማችን ከተለመደው ትርጉሙ ላይ አዲስ ቃና ይጨምራል። ለፕሮግራሞች እነኮቦርድና የመሳሰሉት የገቢ ምንጭ ፤ እነገጽና የመሳሰሉት ደግሞ የወጪ ምንጭ ከተባሉ ፣ «ምንጭ» ሁለት አቅጣጫ ይኖረዋል። ለመጪ ዴታዎችና ነገሮች መነሻ ፤ ለውጪዎቹ ደግሞ መድረሻ ለማለት ነው።

በጃቫ አሠራር ማንኛውም የገቢ ሆነ የወጪ ምንጭ በመደብ ርቢ መወከል አለበት። ከዛ በኃላ ነው ለሥራ ዝግጁ የሚሆነው። በተከታዩ ክፍል ዝርዝሩን እናያለን። በተጨማሪ የጠኑ ችግሮችን ይቀንስ ይረዳ ዘንድ የጃቫ የገቢም ሆነ የወጪ ምንጭ መደቦች

ግዴታዊ ኧክሴፕሽኖች ይጠቀማሉ። በትክክል ተቀብሎ ፥ የተፈለገውን እርምጃ መውሰድ የውዴታ ግዴታ ነው።

10.3 የሆሄያት የገቢ ምንጮች

በምዕራፍ ፪ እንደተጠቀሰው ፥ የጃቫ ሆሄ የዩንኮድን ደንብ ይከተላል። በመሆኑም እያንዳንዱ ሆሄ የሁለት ባይት ቦታ ይወስዳል። በሌላ በኩል የዩንኮድ ሆሄ የማይጠቀም ማንኛውም ኦ/ሲ (Operating System) እንዲሁም ሶፍትዌር ከዚህ ጋር ስለማይጣጣም ከአንድ ስልት ወደሌላኛው በሁለቱም አቅጣጫ ቅየራ መካሄድ አለበት። በዚህ ክፍል ይኸን ሁኔታ ተረድተው ፥ ተገቢውን ቅየራ እየደረጉ የማንበብ ሥራችንን የሚረዱ መደቦች እንጠቀማለን። ወደ ሌሎቹ የጃቫ የዴታ ዓይነቶች ወደ ፊት እንደርስበታለን።

በመጀመሪያ ከኪቦርድ ሆሄያትን እንዴት እናነባለን? ከእንደዚህ ዓይነት የገቢ የኮምፕዩተር ክፍሎች ዴታ የምናነብበት ጥቂት ዘዴዎች አሉ። አንዱ ይኸን ይመስላል። በቅድሚያ ድርጊቶቹ በተርታ።

- አንድን ኪቦርድ በ**System.in** ርቢ መወከል እንችላለን።
- ከኪቦርዱ ሆሄያትንና ኧስትሪንግ ለማንበብ የ**InputStreamReader** መደብ እንጠቀማለን
- እያንዳንዱን ፊደል ለማንበብ የዚህን መደብ **read()** ተብሎ የሚጠራ መላ እንጠቀማለን።

እነዚህ ድርጊቶች በተግባር ያካተተው ከዚህ በታች ይታያል። አንባቢው የደመቁትን ሐረጎች ከተመለከተ ፥ ሦስት ነገሮችን ይታዘባል። የመጀመሪያው ፥ ስህተት ተፈጥሮ ኧክሴፕሽን ከተወረወረ ፥ የሜይን መላ እሱን ለጠራው ወገን ያንን ኧክሴፕሽን ያስተላልፋል። ይህ አንድ ዓይነት እርምጃ መሆኑን ለማሳየት እንጂ ተግባሩ የግዴለሽነት ነው። ምንም ችግር ይምጣ ፥ የራሴን ጣጣ ሌላው ወገን አስታቅፋለሁ እንደማለት ነው። አንዳንድ ሁኔታዎች ውስጥ ግን መልካም እርምጃ ሊሆን ይችላል። ቀጥሎ ፥ በ**System.in** የተመሠረተ ከኪቦርድ አንባቢ ርቢ ይፈጥራል። በመጨረሻ ከኪቦርድ መጫዎቹ ሆሄያትን አንድ በአንድ ያነባል።

ከኪቦርድ ሆሄያት አንባቢ መደብ

```
import java.io.* ;
public class KeyboardReader {

   /** Method: An entry point for program execution */
   public static void main(String[] args) throws IOException  {
      InputStreamReader ir = new InputStreamReader(System.in) ;
      String data = "" ;

      // read until no more character
      int c = ir.read() ;
      while (c != 13) {
         data += (char) c ;
         c = ir.read() ;
      }
      System.out.println("Entered data was: " + data) ;
   }
}
```

Download: KeyboardReader.java

ይህ ፕሮግራም ከኪቦርድ ሆሄያትን ማንበብ መቻሉ ባያጠራጥርም ፥ ፍጥነቱን ማሻሻል እንዲሁም ኧስትሪንግ ደጋግሞ ከመፍጠር መቆጠቡ ጥሩ ሥራ ነው። ሆሄያቱን አንዳንድ በተርታ ከማንበብ ይልቅ በኧስትሪንግ መልክ አንድ ጊዜ ማንበቡ ይመረጣል። ይኸንን የምናደርገው የ**BufferedReader** መደብ በመደረብ ነው። ቃል አዳዳፉ እነሆ። አደራረቡን በጥሞና እንመልከት።

```
InputStreamReader ir =
       new InputStreamReader(System.in) ;
BufferedReader br = new BufferedReader(ir) ;
```

አሁን የ**readLine()** መላ ከ**br** ርቢ ፥ በአንድ ተናጠል እርምጃ ከኪቦርዱ የሚመጡትን ሆሄያት በሙሉ ያነባል።

```
String line = br.readLine() ;
```

ቀጥለን እነዚህን ለውጦችና የተሻሻለ የኧክሴፕሽን አያያዝን ያከተተውን ፕሮግራም እንይ።

ከኪቦርድ ሆሄያት አንባቢ መደብ

```
import java.io.* ;
public class KeyboardReaderBetter {

   /** Method: An entry point for program execution */
   public static void main(String[] args)  {
      InputStreamReader ir = new InputStreamReader(System.in) ;
      BufferedReader br = new BufferedReader(ir) ;
      String data = "" ;

      // read until no more character
      try {
         data = br.readLine() ;
      } catch (IOException e) {
         System.out.println(e) ;
      }
      System.out.println("Entered data was: " + data) ;
   }
}
```

Download: KeyboardReaderBetter.java

አሁን ወደ ፋይል የገቢ ምንጮች እናመራለን። የማንበብ ሂደቱ ቅደም-ተከተል ይኸን ይመስላል።

- ፋይል ከፍቶ ፥ አንብቦ ፥ በመጨረሻ ለመዝጋት በ**File** ርቢ የግድ መወከል አለበት።
- ሆሄያትን አንድ በአንድ ለማንበብ የ**FileReader** መደብ እንጠቀማለን
- ነገር ግን የንባቡን ሥራ ለማጣደፍ የ**BufferedReader** መደብን ከላይ እንደርብና መስመር በመስመር የፋይሉን ይዘት እናነባለን።

ከፋይል ሆሄ አንባቢ መደብ

```
import java.io.* ;
public class FileCharacterReader {

   /** Method: An entry point for program execution */
   public static void main(String[] args) throws IOException  {
      // instantiate String objects
      String fileName = "text.txt" ;
      String cities = "" ;
```

```
        // instantiate and prepare character stream objects
        File inFile = new File(fileName) ;
        FileReader fr = new FileReader(inFile) ;
        BufferedReader br = new BufferedReader(fr) ;

        // read each line from the file
        String line = br.readLine() ;
        while (line != null) {
           cities += line + '\n' ;
           line = br.readLine() ;
        }
        System.out.println("Contents of " + fileName) ;
        System.out.println(cities) ;
     }
}
```

Download: FileCharacterReader.java
Download: text.txt

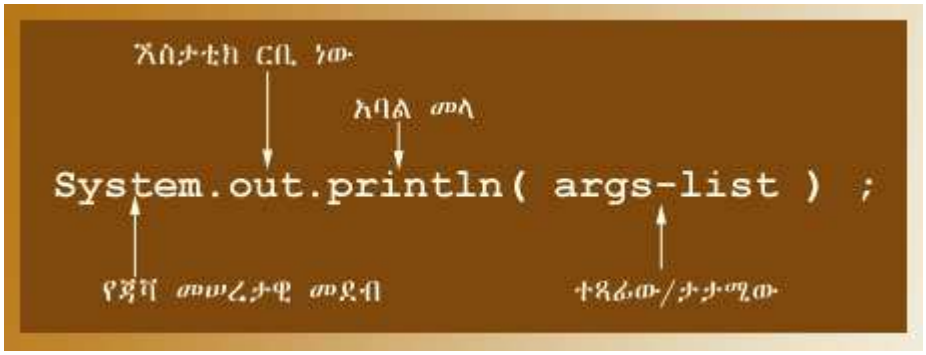
እንደተለመደው ደምቀው በሚታዩት መስመሮች ላይ እናተኩር። የመጀመሪያዎቹ ቃሎች ሦስት ልዩ ልዩ ርቢዎች ከሚከተሉት መደቦች ይፈጥራሉ፦ የFile መደብ ከፋይሎች ጋር አብሮ ለመሥራት 2ኛ) የFileReader መደብ መሠረታዊውን ንባብ ለማካሄድ 3ኛ) የBufferedReader መደብ ንባቡን ለማቀላጠፍ። ሦስቱን ቃሎች አሳጥሮ እንደዚህ መጻፍ ይቻል ነበር ፤ ግን የአንባቢውን ምቾት ይቀንሳል።

አጭር አጻጻፍ

```
BufferedReader br = new BufferedReader(
                        new FileReader(
                            new File(fileName) ) ) ;
```

10.4 የሆሄያት የወጪ ምንጮች

አሁን ትኩረታችንን ወደ የሆሄያት ወጪ ምንጮች እናዞራለን። ነገሮችን ወደ ገጽ መጻፍ ቀላልና ጥናቱ ከተጀመረበት ጊዜ እስካሁን ድረስ ስናካሂደው ቆይተናል። ስለዚህ ወደ ፋይሎች ሆሄያትን እንዴት እንጽፋለን የሚለውን ጥያቄ ለማየት እንሞክራለን። ወደ እሱ ከመግባታችን በፊት ግን የሚከተለውን ቃል ምን እንደነበረ በዝርዝር እንመርምር።



በተሰጡት ምሳሌዎች በሙሉ ይህ ሳይጠቀስ አላለፈም። የ**System** መሠረታዊ ከሚባሉት የጃቫ መደቦች አንዱ ነው። ከኪቦርድ ሆሄያትን ለማንበብ ፥ ወደ ገጽ ነገሮችን ለመጻፍ ፥ ስዓት ለማወቅ ፥ እንዲሁም ለሌሎች ተግባሮች እንጠቀመዋለን። ይህ መደብ በይፋ ያወጣቸው የኧስታቲክ ተውላጠ - ቃላት አሉት፦ **err** ፥ **in** እና **out** ናቸው። የ**out** ርቢ የ**PrintStream** መደብ ዘርና የተለያዩ የዴታ ዓይነቶችን ወደ ገጽ ለመጻፍ የሚያስችሉ መላዎች ያሉት ነው። አሁን ወደ ፋይሎች ጉዳይ እንሂድ።

ከፋይሎች ማንበብ ወይም ወደ እነሱ መጻፍ ፥ የFile ርቢ ውክልና ማስፈለጉን አስቀድመን አውስተናል። ስለዚህ የመጀመሪያው ተግባራችን ይኸው ይሆናል።

```
File outFile = new File("planets.txt") ;
```

ይህ ፋይል ከሌለ አዲስ ይፈጠራል ፤ ካለ ግን ይዘቱ ይሰረዝና በአዲሱ ይተካል። በነገራችን ላይ ፥ የFile መደብ በራሱ የማንበብ ሆነ የመጻፍ ችሎታ ስለሌለው ጸሐፊ መደብ መደረብ ይኖርብናል ፤ እናም እነሆ።

```
FileWriter fr = new FileWriter(outFile) ;
```

የFileWriter መደብ ሆሄያትን የመጻፍ ችሎታ አለው ፤ ግን ሥራው ይጣደፍ ዘንድ የBufferedWriter መደብ መደረቡ ይረዳል።

```
BufferedWriter bw = new BufferedWriter(fr) ;
```

እነዚህን ሁሉ አካቶ ወደ ፋይል የመጻፉን ሥራ የሚያካሂደው መደብ ይህ ነው።

ሆሄያት ጸሐፊ መደብ

```
import java.io.* ;
public class FileCharacterWriter {

   /** Method: An entry point for program execution */
   public static void main(String[] args) throws IOException  {
      // instantiate array of String objects
      String[] planets = {"Mercury", "Venus", "Earth",
                          "Mars", "Jupiter", "Saturn",
                          "Uranus", "Neptun", "Pluto" } ;

      // instantiate and prepare character stream objects
      File outFile = new File("planets.txt") ;
      FileWriter fw = new FileWriter(outFile) ;
      BufferedWriter bw = new BufferedWriter(fw) ;

      // write the array contents into an output file
      try {
         for (int i=0; i < planets.length; i++) {
            bw.write(planets[i]) ;  // write string
            bw.newLine() ;          // write newline
         }
         bw.close() ;               // close file
      } catch (IOException e) {
         System.out.println("File write failur") ;
      }
   }
}
```

Download: FileCharacterWriter.java

10.5 የጃቫ ዴታ ዓይነታትና ምንጮች

በሌሎች የኮምፕዩተር ቋንቋዎች ልዩ ልዩ የዴታ ዓይነታትን ፥ ማለት char ፥ int ፥ float ፥ double እና የመሳሰሉትን በቀላሉ መጻፍ ወይም ማንበብ ይቻላል። ነገር ግን ወደ ጃቫ ስንመጣ ሁኔታው ለየት ይላል። አንዳንድ የጃቫ ዴታ ዓይነታት ከሌሎች ቋንቋዎች ጋር ሲነጻጸሩ ልዩነት እናገኛለን። ለምሳሌ ያህል ፥ የድሮ ቋንቋዎችና ኦ/ሲ የዩንኮድን ሆሄያት አይደግፉም ፤ ማለትም አሁንም ለአንድ ሆሄ የሚሰይሙት ቦታ አንድ ባይት (Byte) ነው። ለምሳሌ ዊንዶስ 98 ብንወስድ ይህ ገሀድ ይሆናል።

C++ መሠረታዊ የዴታ ዓይነታት አውንታዊ ዕሴት ብቻ ወይም አውንታዊና አሉታዊ ዕሴቶችን እንዲይዙ ይፈቅዳል። በጃቫ አውንታዊ ብቻ የሚለው አማራጭ የለም። ከchar በስተቀር ሁሉም አውንታዊና አሉታዊ ዕሴቶችን የመጠበቅ ኃላፊነት አለባቸው።

ይኸን ችግር እዚህ ማንሳቱ ዋጋው ምንድን ነው? በC++ የተጻፈ ፕሮግራም ያወጣውን ፋይል የተለያዩ የጃቫ ዴታ ዓይነታትን ተጠቅመን እናንብብ ብንል አዘቅት ውስጥ እንገባለን። ዴታዎች ይዘበራረቃሉ። ማንነታቸው ጠፍቶ ትርጉም ያጣሉ። በእነዚህና በመሳሰሉት ምክንያት ልዩ ልዩ የጃቫ ዴታ ዓይነታትን ማንበብ ወይም መጻፍ ከፍተኛ ጥንቃቄ ይጠይቃል። ሊነሱ የሚችሉ ችግሮችን ሙሉ በሙሉ ለመቀነስ ፥ መፍትሔው ከሆሄያት ውጭ ንባብ ጠያቂ ፋይሎችን በጃቫ ካልተጻፉ በስተቀር አለመሞከር።

ይኸንኑን ምክር በመከተል ፥ መጀመሪያ በምዕራፍ ፫ የተሰጠውን የጨረቃ ዴታ ወደ ፋይል የሚጽፍ መደብ እናቀርብና አስከትሎ ያንን ፋይል መልሶ አንባቢ መደብ እናያለን።

እንደ ተለመደው ወደ ፋይል ለመጻፍ የFile ርቢ እንፈጥራለን። ቀጥሎ ከፋይል ጋር መሥራት ያለበትን መደብ ርቢ እናበጃለን። በመጨረሻ ፥ የጃቫ ዴታ ዓይነታት መጻፍ የሚችለውን መደብ በመደረብ ዝግጅቱን እናጠናቅቃለን።

```
File in = new File("moon.txt") ;
FileOutputStream fos = new FileOutputStream(in) ;
DataOutputStream dos = new DataOutputStream(fos) ;
```

የDataOutputStream መደብ ማንኛውንም መሠረታዊ የጃቫ የዴታ ዓይነት መጻፍ ይችላል። ለእያንዳንዳቸው የራሱን ጸሐፊ መላ ይለግሳል። ለምሳሌ የdouble ዴታ መጻፍ ብንሻ ውጤቱ ይኸን ይመስላል።

```
final double pi = 3.14159 ;
dos.writeDouble(pi) ;
```

ተግባራዊ ምሳሌ፦

የጃቫ ዴታ ዓይነታት ጸሐፊ

```
import java.io.* ;
public class DataWriter {

   /** Method: An entry point for program execution */
   public static void main(String[] args) throws IOException  {
      double[] data = {7.36E22, 1738, 3340, 1.67, 3.80E5} ;

      // instantiate and prepare character stream objects
      File outFile = new File("moon.txt") ;
```

```
      FileOutputStream fos = new FileOutputStream(outFile) ;
      DataOutputStream dos = new DataOutputStream(fos) ;

      // write the array contents into an output file
      try {
         for (int i=0; i < data.length; i++) {
            dos.writeDouble(data[i]) ;
         }
         dos.close() ;                // close file
      } catch (IOException e) {
         System.out.println("File write failur") ;
      }
   }
}
```

Download: DataWriter.java

መደቡ በኧሬይ ውስጥ የተጠበቁትን የጨረቃ ዴታዎች ወደ «moon.txt» ይጽፋል። የተጠቀምናቸውን መደቦችና ስማቸውን አንባቢው በጥንቃቄ ሊመረምራቸው ይገባል። የተጻፈውን ፋይል ቀጥሎ የቀረበው መደብ አንብሎ ውጤቱን ገጽ ላይ ያሳያል።

የጃቫ ዴታ አንባቢ መደብ

```
import java.io.* ;
public class DataReader {

   /** Method: An entry point for program execution */
   public static void main(String[] args) throws IOException  {

      // instantiate and prepare byte stream objects
      File inFile = new File("moon.txt") ;
      FileInputStream fis = new FileInputStream(inFile) ;
      DataInputStream dis = new DataInputStream(fis) ;

      // Read from the file and print the content
      boolean next = true ;
      try {
         while (next) {
            System.out.println(dis.readDouble()) ;
         }
         dis.close() ;                // close file
      } catch (IOException e) {
         // do nothing
      }
   }
}
```

Download: DataReader.java

To contact: info@senamirmir.com
Copyright © 2002–2005 Senamirmir Project