Leksioni 8
The I/O Package
Reader and writer classes
InputStream and OutputStream classes

Paketën java.io e përmban pothuajse çdo klasë që mund t'ju duhet për të kryer hyrjet dhe daljen (I / O) në Java. Të gjitha këto lëvizje përfaqësojnë një burim të hyrjes dhe një destinacion të daljes. Transmetimi në paketën java.io mbështet shumë të dhëna si primitivë, objekt, karaktere të lokalizuar, etj.

Të dhënat:

Një e dhënë mund të përcaktohet si një sekuencë e të dhënave. Ekzistojnë dy lloje të dhënash –

1. InPutStream - InputStream përdoret për të lexuar të dhëna nga një burim.
2. OutPutStream - OutputStream përdoret për të shkruar të dhëna në një destinacion.

Java siguron një mbështetje të fortë, por fleksibël për I / O në lidhje me skedarët dhe rrjetet, por ky leksion do të mbulojë më shumë funksione themelore në lidhje me të dhënat dhe I / O.

8.1 Byte Streams

Byte Streams në Java përdoren për të kryer hyrjen dhe daljen e bajteve 8 bitësh. Megjithëse ka shumë klasa që lidhen me të dhënat e bajteve, por klasat më të përdorura janë, FileInputStream dhe FileOutputStream. Më poshtë është një shembull që bën përdorimin e këtyre dy klasave për të kopjuar një skedar hyrje në një skedar dalje

Shembull

```
import java.io.*;
public class CopyFile {
   public static void main(String args[]) throws IOException {
      FileInputStream in = null;
      FileOutputStream out = null;
      try {
         in = new FileInputStream("input.txt");
         out = new FileOutputStream("output.txt");
         int c;
         while ((c = in.read()) != -1) {
            out.write(c);
         }
      }finally {
         if (in != null) {
          in.close();}
         if (out != null) {
            out.close();
         }
      }
   }
}
```

8.2 Character Streams

Byte Streams në Java përdoren për të kryer hyrjen dhe daljen e bajtave 8-bitëshe, ndërsatë dhënat e tipit karakter në Java përdoren për të kryer hyrjen dhe daljen për unicode 16-bitëshe. Megjithëse ka shumë klasa që lidhen me të dhënat të tipit karakter, por klasat që përdoren më shpesh janë, FileReader dhe FileWriter. Megjithëse përbrenda FileReader përdor FileInputStream dhe FileWriter përdor FileOutputStream por këtu ndryshimi kryesor është se FileReader lexon dy bajt në të njëjtën kohë dhe FileWriter shkruan dy bajt njëkohësisht.

Ne mund të rishkruajmë shembullin e mësipërm, i cili bën përdorimin e këtyre dy klasave për të kopjuar një skedar të hyrjes (që ka karaktere unicode) në një skedar dalje.

Shembull

```
import java.io.*;
public class CopyFile {
   public static void main(String args[]) throws IOException {
      FileReader in = null;
      FileWriter out = null;
      try {
         in = new FileReader("input.txt");
         out = new FileWriter("output.txt");
         int c;
         while ((c = in.read()) != -1) {
            out.write(c);
         }
      }finally {
         if (in != null) {
            in.close();
         }
         if (out != null) {
            out.close();
         }
      }
   }
}
```

8.3 Të dhënat Standarte (Standard Streams)

Të gjitha gjuhët e programimit ofrojnë mbështetje për I / O standarte ku programi i përdoruesit mund të marrë të dhëna nga një tastierë dhe më pas të prodhojë një dalje në ekranin e kompjuterit. Nëse jeni të vetëdijshëm për gjuhët programuese C ose C ++, atëherë duhet të keni kujdes për tre pajisje standarde STDIN, STDOUT dhe STDERR. Në mënyrë të ngjashme, Java siguron tre rrjedhat standarde vijuese:

1. Hyrja standarte - Kjo përdoret për të siguruar të dhënat në programin e përdoruesit dhe zakonisht të dhënt hyrese që bëhen nga tastiera sigurohen nëpërmjet komandës *System.in*.

2. Dalja standarde - Kjo përdoret për të nxjerrë të dhëna të prodhuara nga programi i përdoruesit dhe zakonisht për afishimin e rezultatit (outputit) përdoret komanda *System.out*.

3. Gabim Standard - Ky përdoret për të nxjerrë të dhëna gabimi të prodhuara nga programi i përdoruesit dhe kjo krijohet nëpermjet komandës *System.err*.

Shembull
Më poshtë është një program i thjeshtë, i cili krijon InputStreamReader për të lexuar rrjedhën standarde të hyrjes derisa përdoruesi të shkruajë një karakter "q".

```
import java.io.*;
public class ReadConsole {
   public static void main(String args[]) throws IOException {
      InputStreamReader cin = null;
      try {
         cin = new InputStreamReader(System.in);
         System.out.println("Enter characters, 'q' to quit.");
         char c;
         do {
            c = (char) cin.read();
            System.out.print(c);
         } while(c != 'q');
      }finally {
         if (cin != null) {
            cin.close();}}}}
```

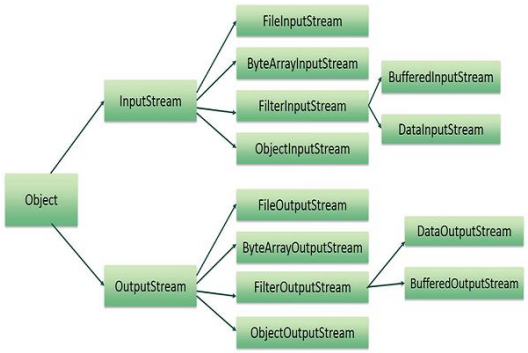
Le të mbajmë kodin e mësipërm në skedarin ReadConsole.java dhe të përpiqemi ta përpilojmë dhe ekzekutojmë atë siç tregohet në programin vijues. Ky program vazhdon të lexojë dhe nxjerrë të njëjtin karakter derisa të shtypim 'q'.

```
$javac ReadConsole.java
$java ReadConsole
```

Enter characters, 'q' to quit.
1
1
e
e
q
q

8.4 Leximi dhe Shkrimi i Skedarëve

Siç është përshkruar më parë, një e dhënë mund të përcaktohet si një sekuencë të dhënash. InputStream përdoret për të lexuar të dhëna nga një burim dhe OutputStream përdoret për të shkruar të dhëna në një destinacion. Këtu është një hierarki e klasave për t'u marrë me rrjedhat e hyrjes dhe daljes.



Dy të dhëna të rëndësishme janë FileInputStream dhe FileOutputStream, të cilat do të diskutohen në këtë material.

a) FileInputStream

Ky stream përdoret për leximin e të dhënave nga skedarët. Objektet mund të krijohen duke përdorur fjalën kyçe new dhe ka disa lloje konstruktorësh të disponueshëm.
Ndërtuesi i mëposhtëm merr një emër skedari si varg për të krijuar një objekt të strimit të hyrjes për të lexuar skedarin:
InputStream f = new FileInputStream("C:/java/hello");

Ndërtuesi i mëposhtëm merr një objekt skedari për të krijuar një objekt të strimit të hyrjes për të lexuar skedarin. Së pari krijojmë një objekt skedari duke përdorur metodën File () si më poshtë:
File f = new File("C:/java/hello");
InputStream f = new FileInputStream(f);

Pasi të keni në dorë objektin InputStream, atëherë ekziston një listë e metodave ndihmëse të cilat mund të përdoren për të lexuar për të transmetuar ose për të bërë operacione të tjera në transmetim.

Nr	Metoda dhe përshkrimi
1	**public void close() throws IOException{}** Kjo metodë mbyll stream e daljes së skedarit. Lëshon çdo burim të sistemit të lidhur me skedarin. Hedh një përjashtim të IO.
2	**public void close() throws IOException{}** Kjo metodë pastron lidhjen me skedarin. Siguron që metoda e mbylljes së këti stream të daljes së skedarit të thirret kur nuk ka më referenca për këtë stream. Hedh një përjashtim të IO.
3	**public int read(int r)throws IOException{}** Kjo metodë lexon bajtin e specifikuar të të dhënave nga InputStream. Kthen një int. Kthen bajtin tjetër të të dhënave dhe -1 do të kthehet nëse është fundi i skedarit.
4	**public int read(byte[] r) throws IOException{}** Kjo metodë lexon bajte r.gjatësi nga rryma e hyrjes në një grup. Kthen numrin total të bajteve të lexuara. Nëse është fundi i skedarit, -1 do të kthehet.
5	**public int available() throws IOException{}** Jep numrin e bajteve që mund të lexohen nga kjo rrjedhë e futjes së skedarit. Kthen një int.

Ekzistojnë tëdhëna të tjera të rëndësishme të hyrjes, për më shumë detaje mund t'i referoheni lidhjeve të mëposhtme - ByteArrayInputStream

8.6 FileOutputStream

FileOutputStream përdoret për të krijuar një skedar dhe për të shkruar të dhëna në të. Transmetimi do të krijonte një skedar, nëse nuk ekziston tashmë, para se ta hapte atë për dalje. Këtu janë dy konstruktorë të cilët mund të përdoren për të krijuar një objekt FileOutputStream. Ndërtuesi vijues merr një emër skedari si varg për të krijuar një objekt të input stream për të shkruar skedarin

OutputStream f = new FileOutputStream("C:/java/hello")
Ndërtuesi i mëposhtëm merr një objekt skedari për të krijuar një objekt të rrjedhës së daljes për të shkruar skedarin. Së pari, ne krijojmë një objekt skedari duke përdorur metodën File () si më poshtë -
File f = new File("C:/java/hello");
OutputStream f = new FileOutputStream(f);
Pasi të keni në dorë objektin OutputStream, atëherë ekziston një listë e metodave ndihmëse, të cilat mund të përdoren për të shkruar për të transmetuar ose për të bërë operacione të tjera në transmetim.

Nr	Metoda dhe përshkrimi
1	**public void close() throws IOException{}** Kjo metodë mbyll stream-in e daljes së skedarit. Lëshon çdo burim të sistemit të lidhur me skedarin. Hedh një përjashtim të IO.
2	**protected void finalize()throws IOException {}** Kjo metodë pastron lidhjen me skedarin. Siguron që metoda e mbylljes së këti stream-i të daljes së skedarit, thërritet kur nuk ka më referenca për këtë stream. Hedh një përjashtim të IO.
3	**public void write(int w)throws IOException{}** Kjo metodë shkruan bajtin e specifikuar në stream-in e daljes.
4	**public void write(byte[] w)** Shkruan bajte w.gjatësi nga grupi i përmendur i bajteve në OutputStream.

Ka rrjedha të tjera të rëndësishme të daljes në dispozicion, për më shumë detaje mund t'i referoheni lidhjeve vijuese:
ByteArrayOutputStream
DataOutputStream

Shembull
Më poshtë është shembulli për të demonstruar InputStream dhe OutputStream -

```
import java.io.*;
public class fileStreamTest {
   public static void main(String args[]) {
      try {
         byte bWrite [] = {11,21,3,40,5};
         OutputStream os = new FileOutputStream("test.txt");
         for(int x = 0; x < bWrite.length ; x++) {
            os.write( bWrite[x] );   // writes the bytes
         }
         os.close();
```

```
            InputStream is = new FileInputStream("test.txt");
            int size = is.available();
            for(int i = 0; i < size; i++) {
               System.out.print((char)is.read() + "  ");
            }
            is.close();
         } catch (IOException e) {
            System.out.print("Exception");
         }
      }
}
```

Kodi i mësipërm do të krijonte skedarin test.txt dhe do të shkruante numrat e dhënë në format binar. E njëjta gjë do të ishte rezultati në ekranin stdout.

8.7 Kërkimi i skedarëve dhe I / O

Ekzistojnë disa klasa të tjera që do të kalonim për të njohur bazat e Navigimit të Skedarëve dhe I/O.

- File Class
- FileReader Class
- FileWriter Class

8.7.1 Direktoritë në Java

Një direktori është një Skedar i cili mund të përmbajë një listë të skedarëve dhe direktorive të tjera. Ne përdorim objektin File për të krijuar direktori, për të renditur skedarët e disponueshëm në një direktori. Për detaje të plota, kontrolloni një listë të të gjitha metodave të cilat mund t'i thërisni në objektin File dhe ato që lidhen me direktoritë.

8.7.2 Krijimi i Direktorive

Ekzistojnë dy metoda të dobishme të përdorimit së skedarit, të cilat mund të përdoren për të krijuar direktori.

1. Metoda mkdir () krijon një direktori, duke kthyer e vërtetë në sukses dhe false në dështim. Dështimi tregon se rruga e specifikuar në objektin File tashmë ekziston, ose që direktoria nuk mund të krijohet sepse e gjithë rruga nuk ekziston ende.

2. Metoda mkdirs () krijon si direktori ashtu edhe të gjithë paraardhësit e direktorisë.

Shembulli vijues krijon drejtorinë "/ tmp / user / java / bin" -

Shembull

```
import java.io.File;
public class CreateDir {

   public static void main(String args[]) {
      String dirname = "/tmp/user/java/bin";
      File d = new File(dirname);
```

```
        // Create directory now.
        d.mkdirs();
    }
}
```

Përpiloni dhe ekzekutoni kodin e mësipërm për të krijuar "/ tmp / user / java / bin".

Shënim - Java automatikisht kujdeset për ndarësit e rrugëve në UNIX dhe Windows sipas kushteve. Nëse përdorni një prerje (/) përpara në një version të Windows të Java, rruga do të zgjidhet siç duhet.

8.7.3 Listimet e Direktorive

Ju mund të përdorni metodën list () të siguruar nga objekti File për të renditur të gjithë skedarët dhe direktoritë e disponueshme në një direktori si më poshtë

```
import java.io.File;
public class ReadDir {
   public static void main(String[] args) {
      File file = null;
      String[] paths;

      try {
         // create new file object
         file = new File("/tmp");

         // array of files and directory
         paths = file.list();

         // for each name in the path array
         for(String path:paths) {
            // prints filename and directory name
            System.out.println(path);
         }
      } catch (Exception e) {
         // if any error occurs
         e.printStackTrace();
      }
   }
}
```

Kjo do të prodhojë rezultatin e mëposhtëm bazuar në direktoritë dhe skedarët e disponueshëm në direktorinë tuaj / tmp –

Output:

test1.txt
test2.txt
ReadDir.java
ReadDir.class