Jazyk Java 1

Seminář 7

Daniel Bazala

Katedra informatiky
Univerzita Palackeho v Olomouci

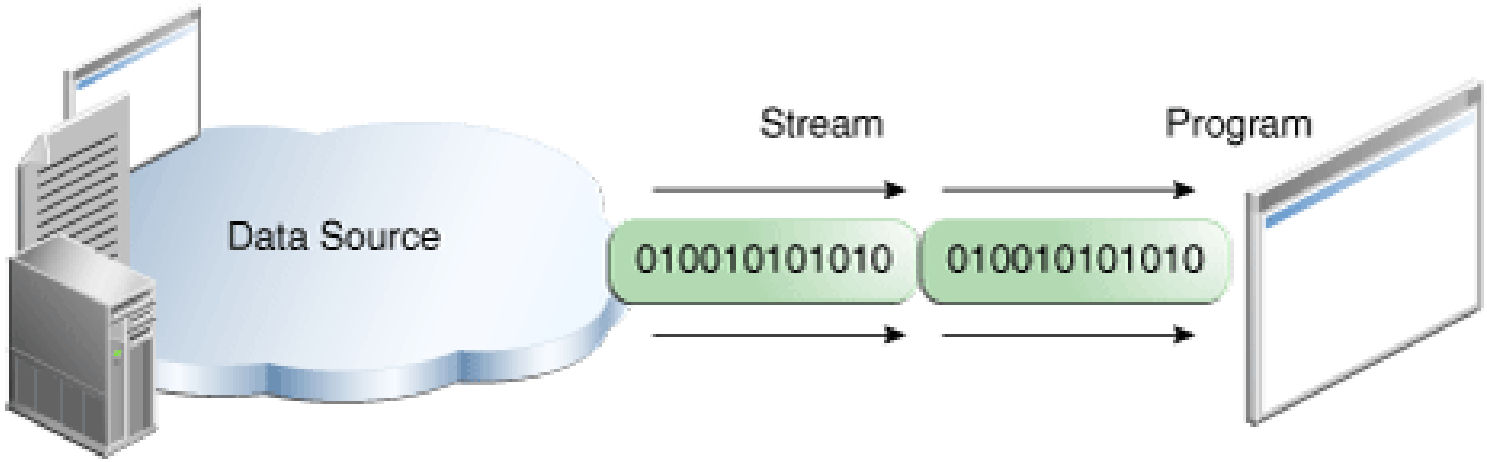
Proudy

Proudy

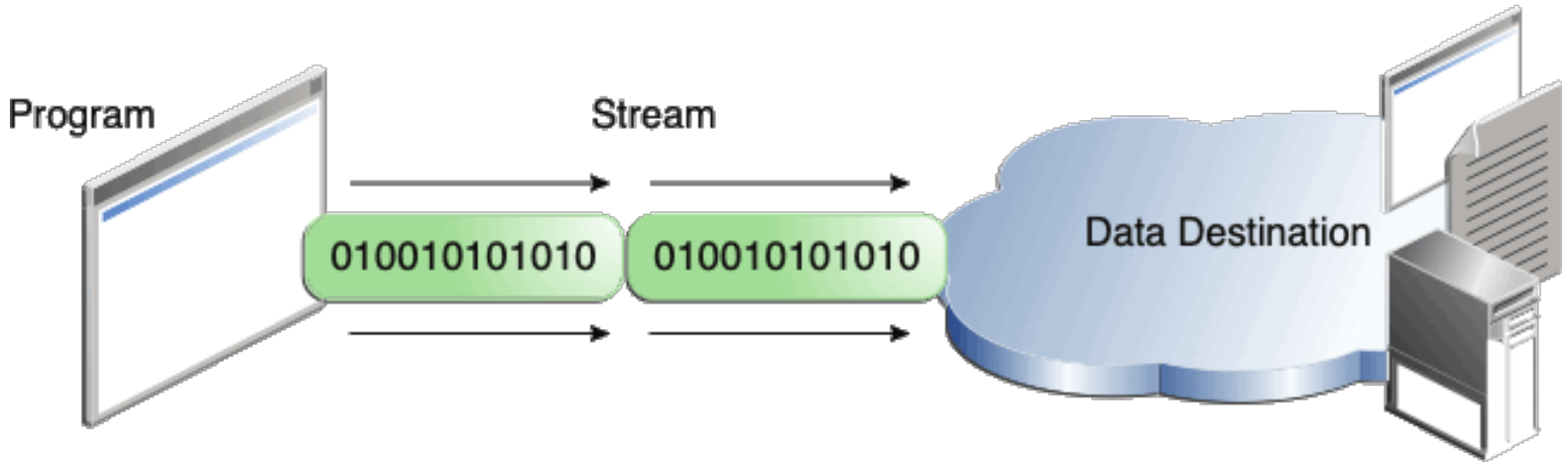
- vstupně/výstupní proudy reprezentují vstupní zdroj dat nebo výstupní destinaci pro data
- pomocí proudů můžeme reprezentovat např. soubory, zařízení, jiné programy..
- proud obvykle jako jednosměrný tok dat ze vstupního zdroje nebo do výstupní destinace
- proudy v Javě podporují různé druhy dat - byty, primitivní datové typy i objekty

Proudy

Input stream



Output stream



Bytové proudy

- dědí z InputStream nebo OutputStream
- typicky pro zápis/čtení souborů

```
FileOutputStream out = null;
String data = "ahoj";

try {
  out = new FileOutputStream("output.txt");
  out.write(data.getBytes());
} finally {
  if (out != null) {
    out.close();
  }
}
```

Bytové proudy

- analogicky FileInputStream pro čtení ze souboru

```
FileInputStream in = null;

try {
  in = new FileInputStream("file.txt");
  int c;

  while ((c = in.read()) != -1) {
    System.out.print((char)c);
  }
} finally {
  if (in != null) {
    in.close();
  }
}

Výstup: obsah textového souboru "file.txt"
```

- problém s kódováním
- řeší znakové proudy

Znakové proudy

- v Javě jsou znaky uloženy pomocí Unicode konvencí
- znakové proudy automaticky převádí z/na místní znakovou sadu

```
FileReader inputStream = null ;

try {
  inputStream = new FileReader("file.txt");
  int c;

  while ((c = inputStream.read()) != -1) {
    System.out.print((char)c);
  }
} finally {
  if (inputStream != null) {
    inputStream.close();
  }
}
```

Proudy s vyrovnávací pamětí

- BufferedInputStream nebo BufferedOutputStream pro bytové proudy
- BufferedReader nebo BufferedWriter pro znakové proudy
- běžný proud můžeme zabalit do proudu s bufferem

```
inputStream = new BufferedReader(new FileReader("input.txt"));
outputStream = new BufferedWriter(new FileWriter("output.txt"));
```

- zápis můžeme vynutit metodou flush()
- BufferedReader má navíc metodu readLine(), která načte řádek

Datové proudy

- datové proudy umožňují čtení/zápis primitivních datových typů a řetězců
- zápis:

```
final double[] prices = { 119.99, 9.99, 2.99};
final int[] units = { 2, 8, 10};
final String[] descs = {"phone", "book", "paper"};

FileOutputStream fos = new FileOutputStream("file");
BufferedOutputStream bos = new BufferedOutputStream(fos);
DataOutputStream out = new DataOutputStream(bos);
for (int i = 0; i < prices.length; i ++) {
  out.writeDouble(prices[i]);
  out.writeInt(units[i]);
  out.writeUTF(descs[i]);
}
out.close();
```

Datové proudy

- čtení:

```
FileInputStream fis = new FileInputStream("file");
BufferedInputStream bis = new BufferedInputStream(fis);
DataInputStream in = new DataInputStream(bis);
try {
  while (true) {
    double price = in.readDouble();
    int unit = in.readInt();
    String desc = in.readUTF();
  }
} catch (EOFException e) {
}
```

Třída Scanner

- používá se pro rozdělení vstupu na části podle datového typu
- výchozím separátorem je mezera/tabulátor/nový řádek

```
Scanner s = null;

try {
  s = new Scanner(new BufferedReader(new FileReader("in.txt")));

  while (s.hasNext()) {
    System.out.println(s.next());
  }
} finally {
  if (s != null) {
    s.close();
  }
}
```

- nastavení jiného separátoru pomocí metody useDelimiter(String separator)

Třída Scanner

- metody pro čtení různých datových typů
- next(), nextInt(), nextFloat, nextBoolean..
- stejně tak hasNext(), hasNextInt(), hasNextFloat()..
- lokalizace přes metodu useLocale(Locale locale);

```
while (s.hasNextInt()) {
  int i = s.nextInt();
  System.out.println(i);
}
```

Systémový vstup

- načítání vstupů od uživatele
- vstupní proud System.in
- můžeme zabalit do třídy Scanner:

```
Scanner scanner = new Scanner(System.in);
System.out.print("Enter␣a␣number:␣");
int i = scanner.nextInt();
System.out.println(i);
scanner.close();
```

Formátování

- https://docs.oracle.com/javase/tutorial/essential/io/formatting.html

Práce se soubory

Práce se soubory

- Cesty: https://docs.oracle.com/javase/tutorial/essential/io/pathOps.html
- kontrola, zda soubor existuje:

```
File file = new File("file.txt");
if(file.exists() && !file.isDirectory()) {
  //Práce se souborem
}
```

- Odstranění souboru:

```
File file = new File("file.txt");
if (file.delete()) {
  System.out.println("Deleted!");
} else {
  System.out.println("Failed!");
}
```

- Více: https://docs.oracle.com/javase/tutorial/essential/io/fileio.html

Standardní knihovna

Standardní knihovna

- https://docs.oracle.com/en/java/javase/16/docs/api/java.base/module-summary.html

Úkol

Úkol seminář 7

- http://marcus.webly3d.net/ukol7