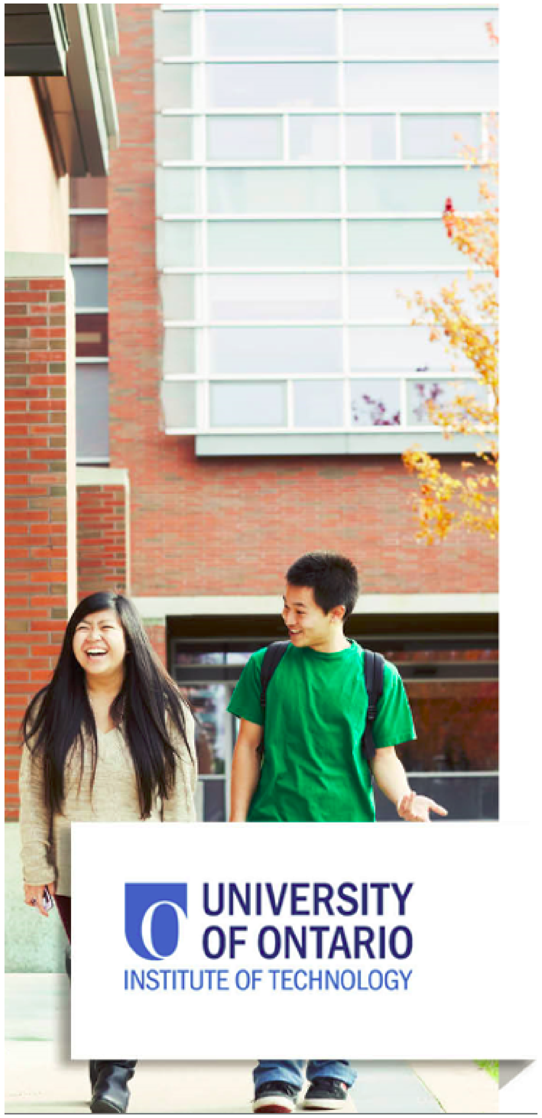


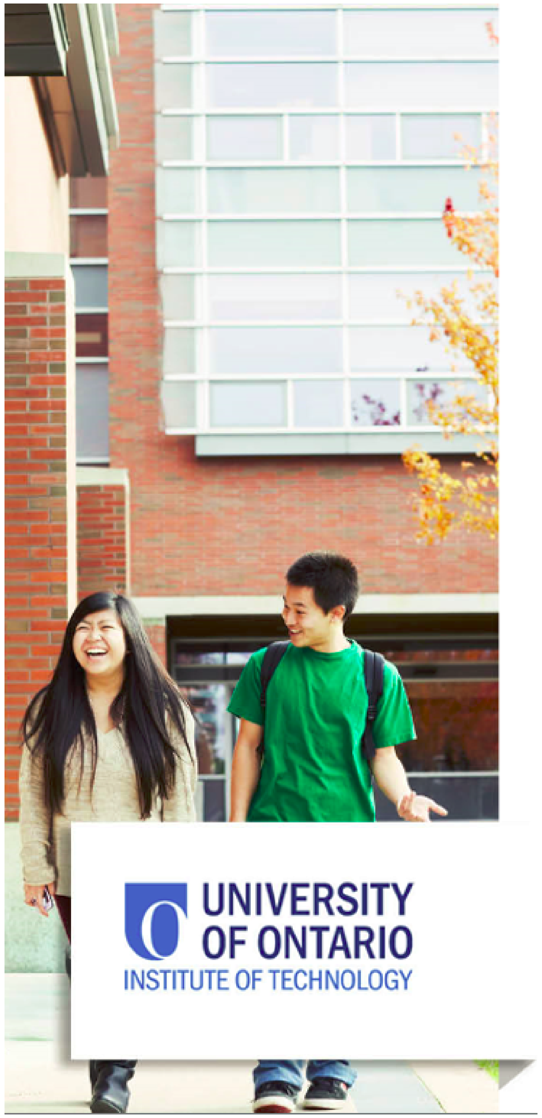
Files and Streams

Loading and Saving Data

Randy J. Fortier
randy.fortier@uoit.ca
@randy_fortier

Outline

- Streams
  - InputStream
  - OutputStream
- Files
  - File
  - FileInputStream
  - FileReader
  - FileOutputStream
  - FileWriter
- Scanner



Files and Streams

Streams

Blocks

- Alternative: just load data on demand
  - Too many disk accesses
  - Delays
- Blocks
  - Buffering
  - Block size
- Problem with blocks:
  - What if we don't want an entire block?

Streams

- Streams are an operating system construct
  - Input stream
    - To the programmer: endless incoming data source
    - Reality: as the disk data is loaded, it is placed into the input buffer
  - Output stream
    - To the programmer: endless outgoing data sink
    - Reality: the output is placed into an output buffer
  - The result is much simpler file (and network) code

Input Streams

Dear employees,

```
Dear employees,\n
```

Input stream buffer

Input Streams

Dear employees,

Wonderful news!

```
Dear employees,\nWonderful news!
```

Input stream buffer

Input Streams

Dear employees,

Wonderful news!

We've been chosen as company of the year!

```
Dear employees,\nWonderful news!\nWe've been chosen as company of the year!
```

Input stream buffer

Input Streams

Read a line: Returns the following immediately:

```
Dear employees,\n
```

```
Wonderful news!\nWe've been chosen as company of the year!
```

Input stream buffer

Input Streams

Read a line: Returns the following immediately:

```
Wonderful news!\n
```

```
We've been chosen as company of the year!
```

Input stream buffer

Input Streams

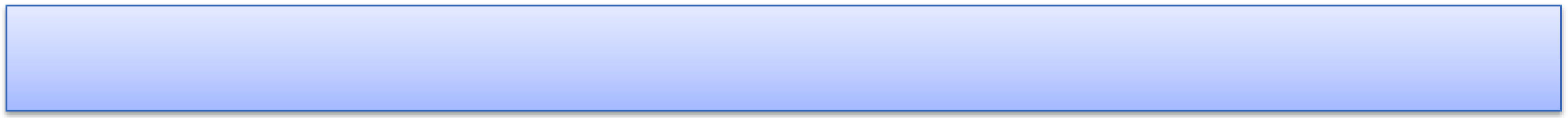
Read a line: Returns the following immediately:

```
We've been chosen as company of the year!\n
```

Input stream buffer

Input Streams

Read a line: Nothing available, so our program blocks



Input stream buffer

Input Streams

Dear employees,

Wonderful news!

We've been chosen as
company of the year!

CEO Sandra Kelley

```
CEO Sandra Telly\n
```

Input stream buffer

Input Streams

Blocking ends, and the following text is returned:

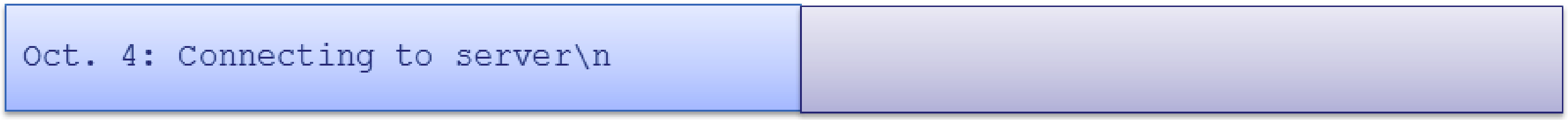
```
CEO Sandra Kelley\n
```

Input stream buffer

Output Streams

Output string:

```
Oct. 4: Connecting to server\n
```



```
Oct. 4: Connecting to server\n
```

Output stream buffer

Output Streams

Output string:

```
Oct. 5: Unauthorized login attempt\n
```

`Oct. 4: Connecting to server\nOct. 5:`	`Unauthorized login attempt\n`

Output stream buffer

Output Streams

Conditions met: sufficient data

```
Oct. 4: Connecting to
server\nOct. 5:
```

```
Unauthorized login attempt\n
```

Output stream buffer

Input Streams in Java

- InputStream and FileInputStream:

```
final int BLOCK_SIZE = 1024;
InputStream input = new FileInputStream("myfile.txt");
byte[] buffer = new byte[BLOCK_SIZE];
int numBytesRead = 0;
while ((numBytesRead = input.read(buffer)) != -1) {
    // do something with buffer[0..numBytesRead-1]
}
```

Output Streams in Java

- OutputStream and FileOutputStream:

```
final int BLOCK_SIZE = 1024;
OutputStream output = new FileOutputStream("myotherfile.txt");
byte[] buffer = new byte[BLOCK_SIZE];
boolean keepGoing = true;
while (keepGoing) {
    // fill up buffer with data

    output.write(buffer);

    // update keepGoing if we are done writing data
}
```

Readers in Java

- FileReader: Reads characters (not bytes)
- BufferedReader:
  - Handles buffering
  - Read line-by-line
- Example:

```java
FileReader fileReader = new FileReader("myotherfile.txt");
BufferedReader input = new BufferedReader(fileReader);
String line = null;
while ((line = input.readLine()) != null) {
    // do something with line
}
```

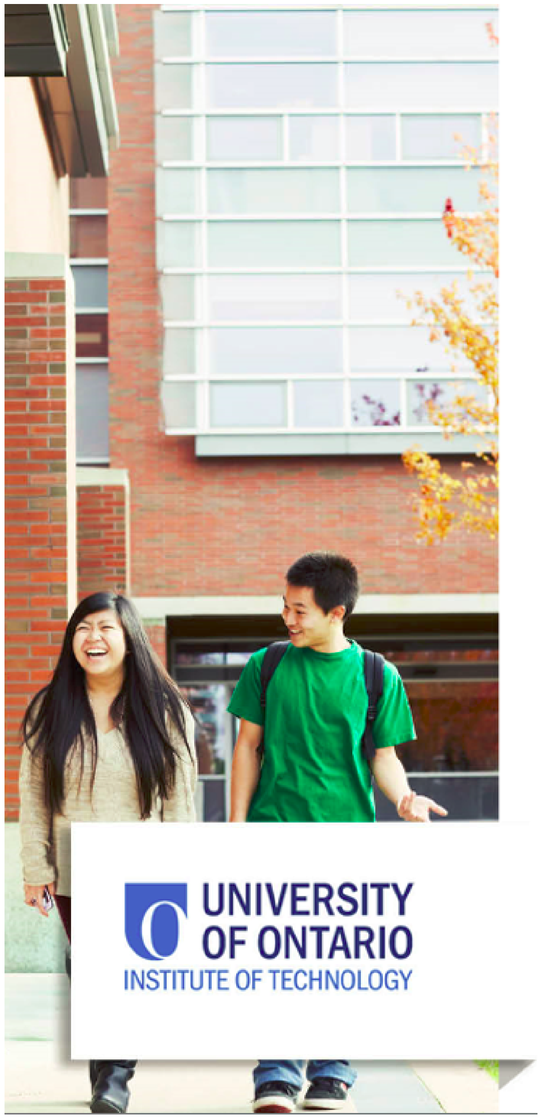
Writers in Java

- FileWriter: Writes characters (not bytes)
- PrintWriter:
  - Write line-by-line
  - e.g. System.out
- Example:

```java
PrintWriter output = new PrintWriter("myotherfile.txt");
boolean keepGoing = true;
String line = null;
while (keepGoing) {
    // update line with new data

    output.println(line);

    // update keepGoing, if no more data to save
}
output.close();
```



Files and Streams

Files

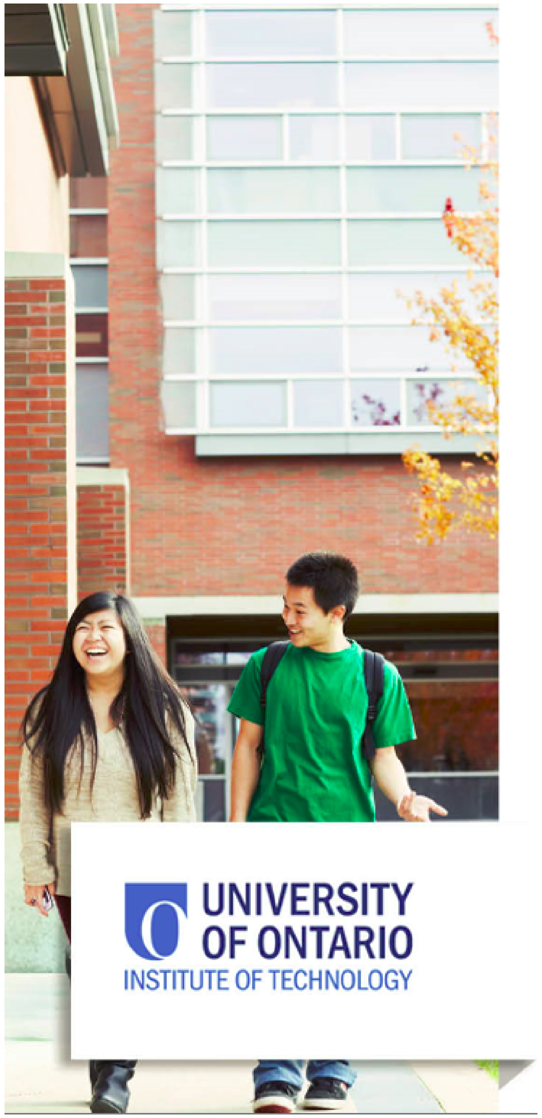
Files

- File:
    - File::exists()
    - File::isDirectory()
    - File::mkdir(), File::mkdirs()
    - File::renameTo(File)
    - File::setLastModified(long)
    - File::setReadOnly()
    - File::File::toURL()
    - File::File::canRead()
    - File::File::canWrite()
    - File::getAbsolutePath()

File

- Example:

```
File outFile = new File("relativeFile.txt");
File inFile = new File("/path/to/file/absoluteFile.txt");
if (inFile.exists()) {
    BufferedReader input = new BufferedReader(new FileReader(inFile)
    PrintWriter output = new PrintWriter(outFile);
    String line = null;
    while ((line = input.readLine()) != null) {
        output.println(line);
    }
    input.close();
    output.close();
}
```



Files and Streams

Scanner

Scanner

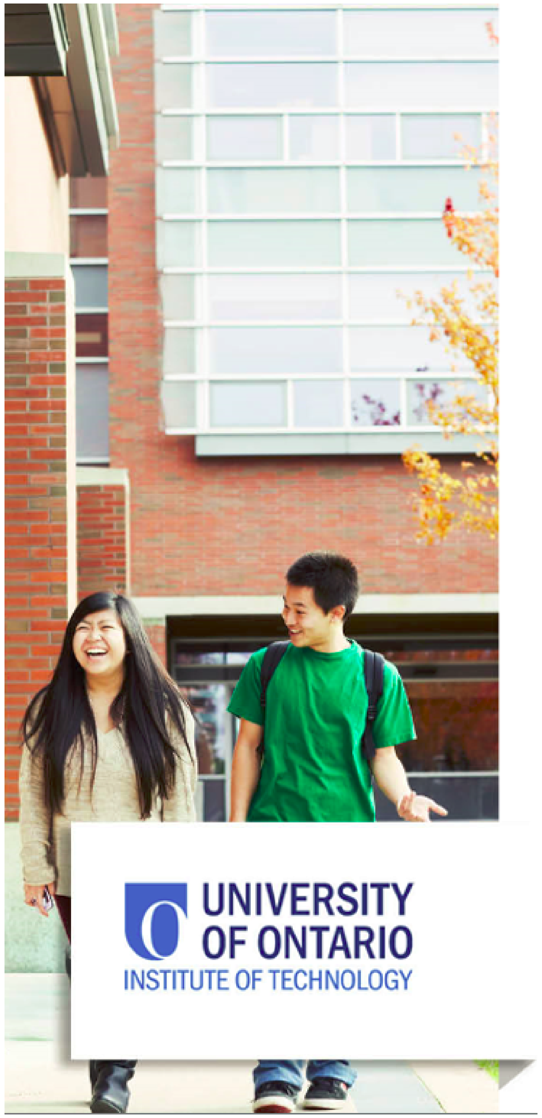
- Scanner:
  - Parses data values from any input stream or reader

```
File inFile = new File("/path/to/file/absoluteFile.txt");
Scanner scanner = new Scanner(inFile);
while (scanner.hasNext()) {
    String nextWord = scanner.next();
}
```

Scanner

- Scanner:
  - Values are separated by delimiters
    - By default, delimiters are whitespace characters
    - You can change them to anything you like

```
File inFile = new File("/path/to/file/absoluteFile.txt");
Scanner scanner = new Scanner(inFile);
scanner.useDelimiter("[^0-9]"); // any non-digit characters
while (scanner.hasNextInt()) {
    int nextInt = scanner.nextInt();
}
```



Files and Streams

Scanner

CSV

- Comma-separated values:
  - Values are separated by comma delimiters
  - Spreadsheet programs (e.g. Calc, Excel) can export it
  - Some open/API data is shared in this format
    - Toronto Parking Tickets

```
Name,Asmt1,Asmt2,Labs,Midterm,Final
Bart Simpson,6.0,4.5,6.5,20.25,29.0
Lisa Simpson,10.0,10.0,10.0,29.5,58.25
Ralph Wiggum,0.5,0.25,0.75,8.0,12.5
Homer Simpson,6.5,5.5,5.5,18.5,26.5
```

Wrap-Up

- In this section we learned about:
  - Input and output streams
  - Files
  - Readers and writers
  - Scanner