# Text File Input and Output

Course: CPSC 1150
Term: Fall 2016
Instructor: Adina Goldberg
E-mail: agoldberg@langara.bc.ca

Langara College
100 West 49th Avenue
Vancouver, BC
www.langara.bc.ca

Lecture 18

# Files in Java

- It is often necessary to have a program load/store data
  - The program must be able to read from a file/write to a file
- Files are treated as objects in java
- There is a `File` class with built-in methods for managing files
- Can read from files in a similar way to scanning for user input, using a `Scanner`
- Can write to files in a similar way to printing to the console, using a `PrintWriter`

# Creating a `File` object for a .txt file

- Whether you want to read from or write to a text file, you should create a `File` object in your program
- To use `File` objects, you need the following import statement at the top of your code:

```
import java.io.File;
```

## Creating a `File` object

```
File myFile = new File("fileName.txt");
```

- This statement does not create a file in your file system, but just creates a **Java object** to represent the file in your program
- Could also use a `File` object to represent a directory

# Some methods in the `File` class

- These are all **instance** methods

| **Method** | **What it does...** |
|---|---|
| `exists()` | Returns `true` if the file (or directory) exists. |
| `isDirectory()` | Returns `true` if the `File` object represents a directory. |
| `isFile()` | Returns `true` if the `File` object represents a file. |
| `getName()` | Returns a `String` containing the filename. |
| `getPath()` | Returns a `String` containing the path. |
| `lastModified()` | Returns a `long` containing a time (in ms). |
| `length()` | Returns a `long` containing the size. |
| `delete()` | Deletes the file and returns `true` if successful. |
| `mkdir()` | Creates a directory and returns `true` if successful. |

There are more in Section 12.10 in the textbook.

# Throwing exceptions

- In Java, you as a programmer are forced to explicitly deal with errors that might occur, instead of letting them crash your program
  - File I/O is prone to many possible errors
- In this course, we don't learn to handle **exceptions**, so for now, we need to put the code

```
throws IOException
```

  at the end of any method header (including `main`) which performs file I/O
- Now code that uses our methods must either **catch** (handle) or continue to **throw** the exception
  - Think of an exception as a hot potato or a grenade – you need to neutralize it or throw it away

# Reading from a file

- General steps:
  1. Create a `File` object using the filename you want to read from.
  2. Check if that file exists (and is readable, if you like).
     - If not, do not attempt to read it.
  3. Create a `Scanner` of the file.
  4. Use the usual `Scanner` methods to extract data.
  5. Process/use the data however you like, once you have extracted it.
  6. Make sure to close this `Scanner` when you are done!

## Example

Let's look at the program `ReadFromFile` for an example of how to do this.

# Writing to a file

- General steps:
  1. Create a `File` object using the filename you want to write to.
  2. Check if that file exists (and is writeable, if you like).
     - If it does, and you write to it, beware that you will overwrite the current contents of the file.
     - Usually it's good to warn the user that the file already exists, and they need to pick another filename.
  3. Create a `PrintWriter` object to allow you to write to the file.
  4. Use the usual `print`, `println`, `printf` methods with the `PrintWriter`.
  5. Close the `PrintWriter` when finished.

## Example

Let's look at the program `WriteToFile` for an example of how to do this.

# Creating a PrintWriter object

- You can use a `PrintWriter` in order to write to a file.
- To use `PrintWriter` objects, you need the following import statement at the top of your code:

```
import java.io.PrintWriter;
```

## Creating a PrintWriter object

```
PrintWriter pw = new PrintWriter(myFile);
//myFile is a File object
```

- From the API: "If the file exists then it will be truncated to zero size; otherwise, a new file will be created."
- `pw` replaces `System.out` when invoking print methods to write to files