Class File

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

Using Class File

- A File type is used for accessing a file
- To be able to use the File class, you need an import statement (as in the case of Scanner), either one of the following:
  - `import java.io.*;`
  - `import java.io.File;`

Class File

- A File type can be created by supplying a String that is treated as the file name, e.g.,

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

- Note that:
  - The expected location of the file is the directory in which the Java program is running
  - The statement only says that myFile should refer to the file by the name of "foo.txt" in the directory in which the program is running
  - It does not guarantee the existence of the file

Direct Path Specification

- It is also possible to specify a “path" from the program-running directory
- Suppose there is a directory by the name of "datasets", which has a directory by the name of "firstSet", and you want to access a file by the name of "test01.txt", then you can use:

```
File myFile =
new File("datasets/firstSet/test01.txt");
```

  - The "/" indicates a sub-folder
  - The "/" should be substituted with the "\" for Windows

List of Methods for File

The methods below should be attached to a file type, for example, if f is a file type (e.g., one generated by `new File("filename")`) and the method is `exists()`, then `f.exists()`

Method	Return type	Function
exists()	boolean	Tests whether the file exists
isDirectory()	boolean	Tests whether the file is a directory
isFile()	boolean	Tests whether the file is a regular file
getName()	String	Returns the file name
getAbsolutePath()	String	Returns the absolute path to the file from the root directory
length()	int	file size (# bytes)
mkdirs()	boolean	Create the directory
renameTo(File g)	boolean	Renames the file to g and returns whether the creation succeeded

FileExplore.java for Exploration

```
1  import java.io.*;
2  import java.util.*;
3  public class FileExplore {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      System.out.print( "Enter the path to a file: " );
7      String path = console.nextLine();
8      File file = new File( path );
9      System.out.print( "The file's name: " );
10     System.out.println( file.getName() );
11     System.out.print( "The file's absoluate path: " );
12     System.out.println( file.getAbsolutePath() );
13     System.out.print( "The file exists: " );
14     System.out.println( file.exists() );
15     System.out.print( "The file can is a regular file: " );
16     System.out.println( file.isFile() );
17     System.out.print( "The file can is a directory: " );
18     System.out.println( file.isDirectory() );
19     System.out.print( "The file length: " );
20     System.out.println( file.length() );
21   }
22 }
```

Create a scanner, prompt the user, and receive the path to the file

FileExplore.java for Exploration

```
1  import java.io.*;
2  import java.util.*;
3  public class FileExplore {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      System.out.print( "Enter the path to a file: " );
7      String path = console.nextLine();
8      File file = new File( path );
9      System.out.print( "The file's name: " );
10     System.out.println( file.getName() );
11     System.out.print( "The file's absoluate path: " );
12     System.out.println( file.getAbsolutePath() );
13     System.out.print( "The file exists: " );
14     System.out.println( file.exists() );
15     System.out.print( "The file can is a regular file: " );
16     System.out.println( file.isFile() );
17     System.out.print( "The file can is a directory: " );
18     System.out.println( file.isDirectory() );
19     System.out.print( "The file length: " );
20     System.out.println( file.length() );
21   }
22 }
```

Create a file by the path

FileExplore.java for Exploration

```
1  import java.io.*;
2  import java.util.*;
3  public class FileExplore {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      System.out.print( "Enter the path to a file: " );
7      String path = console.nextLine();
8      File file = new File( path );
9      System.out.print( "The file's name: " );
10     System.out.println( file.getName() );
11     System.out.print( "The file's absoluate path: " );
12     System.out.println( file.getAbsolutePath() );
13     System.out.print( "The file exists: " );
14     System.out.println( file.exists() );
15     System.out.print( "The file can is a regular file: " );
16     System.out.println( file.isFile() );
17     System.out.print( "The file can is a directory: " );
18     System.out.println( file.isDirectory() );
19     System.out.print( "The file length: " );
20     System.out.println( file.length() );
21   }
22 }
```

Announce the methods to execute

FileExplore.java for Exploration

```
1  import java.io.*;
2  import java.util.*;
3  public class FileExplore {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      System.out.print( "Enter the path to a file: " );
7      String path = console.nextLine();
8      File file = new File( path );
9      System.out.print( "The file's name: " );
10     System.out.println( file.getName() );
11     System.out.print( "The file's absoluate path: " );
12     System.out.println( file.getAbsolutePath() );
13     System.out.print( "The file exists: " );
14     System.out.println( file.exists() );
15     System.out.print( "The file can is a regular file: " );
16     System.out.println( file.isFile() );
17     System.out.print( "The file can is a directory: " );
18     System.out.println( file.isDirectory() );
19     System.out.print( "The file length: " );
20     System.out.println( file.length() );
21   }
22 }
```

Execute the methods

Table of Contents

Creating a Scanner from a File

- To create a Scanner object from a file, create a new Scanner type

```
Scanner myScanner = new Scanner(theFile);
```

where theFile is a File type and myScanner is the name of the Scanner type

Creating a Scanner from a File

- To create a Scanner object from a file, create a new Scanner type

```
Scanner myScanner = new Scanner(theFile);
```

  where theFile is a File type and myScanner is the name of the Scanner type

- This is when the existence of the File type is tested and if the file does not exist, a runtime error by the name of:

```
FileNotFoundException
```

  occurs

Dealing with FileNotFoundException

To deal with the error the method has to have an additional declaration

```
throws FileNotFoundException
```

This has to appear between the end of the method declaration and the left curly bracket of the method body

Simple Program for `Scanner` Creation

- Use a do-while loop to receive input from user; in the loop
  - Receive a file name using `next`
  - Create a `File` object by the name
  - Create a `Scanner` object from the file
  - Ask user whether to continue
  - Terminate loop if the answer starts with a `"y"` (use `startsWith` method)

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

The main method has a `throws` declaration

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

Create a Scanner

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

Declare String variables

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

The do-while loop that lasts until response does not start with "y"

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

Receive the path to the file and create a file with the path

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

Create a scanner out of the file

Scanner Creation Without Error Handling

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreate {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      String response, path;
8      do {
9        System.out.print( "Type a path to file: " );
10       response = console.nextLine();
11       File myFile = new File( response );
12       System.out.println( "... Creating a scanner... " );
13       Scanner fScan = new Scanner( myFile );
14       System.out.print( "------ Continue (y/n)? " );
15       response = console.nextLine();
16     } while ( response.startsWith( "y" ) );
17   }
18 }
```

Receive response from the user

Handling FileNotFoundException Using Try-Catch

One can use a `try-catch` construct to handle execptions

The structure is:

```
1  try {
2    // actions to perform
3  } (catch EXCEPTION-TYPE1 e) {
4    // actions to perform
5  } (catch EXCEPTION-TYPE2 e) {
6    // actions to perform
7  } ...
8    (catch EXCEPTION-TYPEk e) {
9    // actions to perform
10 } ...
```

The exceptions must be of incompatible types

Handling FileNotFoundException Using Try-Catch

One can use a `try-catch` construct to handle execptions

The structure is:

```
1  try {
2    // actions to perform
3  } (catch EXCEPTION-TYPE1 e) {
4    // actions to perform
5  } (catch EXCEPTION-TYPE2 e) {
6    // actions to perform
7  } ...
8    (catch EXCEPTION-TYPEk e) {
9    // actions to perform
10 } ...
```

The exceptions must be of incompatible types

- The code inside the `try` part is executed
- If no error, nothing else happens
- If an exception occurs and if there is a catch with type matching this exception type, the code after the catch clause is executed
- If an exception occurs and if there is no match, the method halts with the exception

Scanner Creation with Try-Catch

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreateTryCatch {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      String response, path;
7      do {
8        System.out.print( "Type a path to file: " );
9        response = console.nextLine();
10       File myFile = new File( response );
11       System.out.println( "... Creating a scanner... " );
12       try {
13         Scanner fScan = new Scanner( myFile );
14       } catch( FileNotFoundException e ) {
15         System.out.printf( "Scanner for %s could not be created.%n",
16             myFile.getName() );
17       }
18       System.out.print( "------ Continue (y/n)? " );
19       response = console.nextLine();
20     } while ( response.startsWith( "y" ) );
21   }
22 }
```

No need to declare throws

Scanner Creation with Try-Catch

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreateTryCatch {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      String response, path;
7      do {
8        System.out.print( "Type a path to file: " );
9        response = console.nextLine();
10       File myFile = new File( response );
11       System.out.println( "... Creating a scanner... " );
12       try {
13         Scanner fScan = new Scanner( myFile );
14       } catch( FileNotFoundException e ) {
15         System.out.printf( "Scanner for %s could not be created.%n",
16             myFile.getName() );
17       }
18       System.out.print( "------ Continue (y/n)? " );
19       response = console.nextLine();
20     } while ( response.startsWith( "y" ) );
21   }
22 }
```

Enclose the creation of a Scanner in try

Scanner Creation with Try-Catch

```
1  import java.util.*;
2  import java.io. * ;
3  public class FileScannerCreateTryCatch {
4    public static void main( String[] args ) {
5      Scanner console = new Scanner( System.in );
6      String response, path;
7      do {
8        System.out.print( "Type a path to file: " );
9        response = console.nextLine();
10       File myFile = new File( response );
11       System.out.println( "... Creating a scanner... " );
12       try {
13         Scanner fScan = new Scanner( myFile );
14       } catch( FileNotFoundException e ) {
15         System.out.printf( "Scanner for %s could not be created.%n",
16             myFile.getName() );
17       }
18       System.out.print( "------ Continue (y/n)? " );
19       response = console.nextLine();
20     } while ( response.startsWith( "y" ) );
21   }
22 }
```

Action to perform upon FileNotFoundException

Table of Contents

Reading data or line from a file

Once a Scanner type has been created from a file, we can use the methods `next()`, `nextLine()`, `nextInt()`, and `nextDouble()`

Scanner and tokens

Scanner moves its position on the character sequence and does the following:

- `hasNext()`, `hasNextInt()`, `hasNextDouble()`, and `hasNextBoolean()` answer whether there are any token, any int, any double, and boolean coming up starting from the current position, respectively
- `hasNextLine()` asks whether there is a "\n" ahead
- `next()`, `nextInt()`, `nextDouble()`, and `nextBoolean()` respectively retrieve the next token as a String, an int, a double, and a boolean and move the position the letter immediately after the token
- `nextLine()` retrieves the series of letters until the next newline and moves the position the letter immediately after the newline

Table of Contents

Printing a File with Numbers

- Print the contents of the text file on screen with line numbers
- Assume that the number of lines is less than a million and use six character spaces for printing the number
- For each line, print the line number in six character spaces, one colon, one space, and then the line

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Method declaration with **throws** declaration

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Create a scanner for the user input

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Receive a path, create the file, and open a file scanner

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Set the line counter to 0

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

while there is a line remaining do the body of the loop

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Increase the counter by 1

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Read the next line

PrintWithLineNumber.java

```
1  import java.io. * ;
2  import java.util. * ;
3  // print a file with line numbers
4  public class PrintWithLineNumber {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter a path: " );
9      String fileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( fileName ) );
11     int count = 0;
12     while ( fileScanner.hasNextLine() ) {
13       count ++ ;
14       String line = fileScanner.nextLine();
15       System.out.printf( "%06d: %s%n", count, line );
16     }
17   }
18 }
```

Generate output

Reading Formatted Data

- An input file has a header line and after that has a repetition of:
  - name (String) height (double) weight (double) age (int)
- Read from the file and print the data

Reading Formatted Data

- An input file has a header line and after that has a repetition of:
  - name (String) height (double) weight (double) age (int)
- Read from the file and print the data

```
1  Name  Height  Weight  Age
2  Johnson,Jack  67.0  140.5 40
3  Garland,Judy  72.0  125.3 38
4  Dickinson,Emily 68.5  158.4 27
5  Cole,Kenneth  40.5  110.7 12
```

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Method declaration with `throws`

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Create a scanner for the user input

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Prompt the user to receive file path and then create a Scanner

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Read the head line and print it

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Then the loop is executed as long as there remains a token

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Read four tokens

Scanner and Reading Formatted Data

```
1  import java.util.*;
2  import java.io. * ;
3  public class ReadTokensFromFile {
4    public static void main( String[] args )
5        throws FileNotFoundException {
6      Scanner console = new Scanner( System.in );
7      System.out.print( "Enter a path: " );
8      String fileName = console.nextLine();
9      Scanner fileScanner = new Scanner( new File( fileName ) );
10     System.out.println( fileScanner.nextLine() );
11     while ( fileScanner.hasNext() ) {
12       String name = fileScanner.next();
13       double height = fileScanner.nextDouble();
14       double weight = fileScanner.nextDouble();
15       int age = fileScanner.nextInt();
16       System.out.printf( "%-20s\t%-6.2f\t%-6.2f\t%-2d%n",
17           name, height, weight, age );
18     }
19   }
20 }
```

Generate a formatted output of the four tokens

Table of Contents

Class PrintStream

- PrintStream is a class for writing to files
- PrintStream objects can be created using a declaration similar to the one for Scanner, e.g.,
  - PrintStream myStream = new PrintStream(f)
    where f is a File
- PrintStream has methods
  printf(. . .), print(. . .), and println(. . .),
  which work the same way as their System.out versions

Import and FileNotFoundException

- An appropriate import for PrintStream is either java.io.PrintStream or java.io.*
- If the said file does not exist, Java creates a new File specified by the Scanner, but
  - Java does not create a folder, if the file belongs to a non-existing folder
  - if that happens, some exception may occur (FileNotFoundException, in particular), but javac does not enforce treatment

Converting the Input File to All Upper-case

Convert the contents of a file to all upper case

- Read from a file that the user provides
- Create a new file with the converted lines
- Both Scanner and PrintStream may produce a run-time error of FileNotFoundException at the time of creation

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Main method with `throws`

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Create console

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Prompt the user for an input file path; create a Scanner

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Prompt the user for an output file path; create a PrintStream

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

The while loop is executed as long as there are lines remaining

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Read one line

Converting the Input File to All Upper-case

```
1  import java.util.*;
2  import java.io. * ;
3  // Convert a file contents to all upper case
4  public class ToUpper {
5    public static void main( String[] args )
6        throws FileNotFoundException {
7      Scanner console = new Scanner( System.in );
8      System.out.print( "Enter an input file path: " );
9      String inFileName = console.nextLine();
10     Scanner fileScanner = new Scanner( new File( inFileName ) );
11     System.out.print( "Enter an output file path: " );
12     String outFileName = console.nextLine();
13     PrintStream fileStream =
14         new PrintStream( new File( outFileName ) );
15     while ( fileScanner.hasNextLine() ) {
16       String line = fileScanner.nextLine();
17       fileStream.println( line.toUpperCase() );
18     }
19   }
20 }
```

Print the uppercase version

Result

```
"We Are Young"
(feat. Janelle Monae)

Give me a second I,
I need to get my story straight
My friends are in the bathroom getting higher than the Empire State
My lover she's waiting for me just across the bar
My seat's been taken by some sunglasses asking 'bout a scar, and
I know I gave it to you months ago
I know you're trying to forget
But between the drinks and subtle things
The holes in my apologies, you know
I'm trying hard to take it back
So if by the time the bar closes
And you feel like falling down
I'll carry you home

Tonight
We are young
So let's set the world on fire
We can burn brighter than the sun

Tonight
We are young
So let's set the world on fire
```

Result

```
"WE ARE YOUNG"
(FEAT. JANELLE MONAE)

GIVE ME A SECOND I,
I NEED TO GET MY STORY STRAIGHT
MY FRIENDS ARE IN THE BATHROOM GETTING HIGHER THAN THE EMPIRE STATE
MY LOVER SHE'S WAITING FOR ME JUST ACROSS THE BAR
MY SEAT'S BEEN TAKEN BY SOME SUNGLASSES ASKING 'BOUT A SCAR, AND
I KNOW I GAVE IT TO YOU MONTHS AGO
I KNOW YOU'RE TRYING TO FORGET
BUT BETWEEN THE DRINKS AND SUBTLE THINGS
THE HOLES IN MY APOLOGIES, YOU KNOW
I'M TRYING HARD TO TAKE IT BACK
SO IF BY THE TIME THE BAR CLOSES
AND YOU FEEL LIKE FALLING DOWN
I'LL CARRY YOU HOME

TONIGHT
WE ARE YOUNG
SO LET'S SET THE WORLD ON FIRE
WE CAN BURN BRIGHTER THAN THE SUN

TONIGHT
WE ARE YOUNG
SO LET'S SET THE WORLD ON FIRE
```

The End