File handling Java

Learning Objectives

- Learn file handling in Java:
  - Create
  - Write
  - Read

Java File Handling - File Class

File Class: It is part of the java.io package and allow us to work with files.

How to use it?

Create a File object, and specify the filename or directory name:

import java.io.File; => Import the File class from package java.io

File myFile = new File("filename.txt"); => Create File object and indicate the filename

File methods

Some File methods:

createNewFile(): Creates an empty file

delete(): Deletes a file

exists(): Checks whether the file exists

getName(): Returns the name of the file

Create a file

```
import java.io.File;  // Import the File class
import java.io.IOException;  // Import the IOException class to handle errors

public class MyFiles {

   public static void createFile() {
     try {
          File myFile = new File("filename.txt");
          if (myFile.createNewFile()) {
               System.out.println("File created: " + myFile.getName());
          } else {
               System.out.println("File already exists.");
          }
     } catch (IOException e) {
          System.out.println("An error occurred.");
          e.printStackTrace();
     }
   }
}
```

Write File

```
import java.io.FileWriter;
import java.io.IOException;

  public static void writeFile() {
    try {
        FileWriter myWriter = new FileWriter("writeme.txt");
        myWriter.write("This is a test.\nAnother test.");
        myWriter.close();
        System.out.println("Successfully!!!");
    } catch (IOException e) {
        System.out.println("Error!!!");
        e.printStackTrace();
    }
  }
```

Write File - Append mode

```
import java.io.FileWriter;
import java.io.IOException;

  public static void writeFile() {
    try {
        // The argument true enables append mode
        FileWriter myWriter = new FileWriter("writeme.txt",  true);
        myWriter.write("This is a test.\nAnother test.");
        myWriter.close();
        System.out.println("Successfully!!!");
    } catch (IOException e) {
        System.out.println("Error!!!");
        e.printStackTrace();
    }
  }
```

Read a file - Scanner

You can use a Scanner on files, strings or user input.

Import the java.util.Scanner package before we can use the Scanner class:

import java.util.Scanner;

Create a Scanner Object in Java:

Scanner sc = new Scanner(" File Here");

Scanner methods

hasNext() - Returns true if this Scanner has another token in its input. (Does not consume that text)

hasNextLine() - Returns true if there is another line in the input of this Scanner. (Does not consume that text)

hasNextInt() - Returns true if the next token in this Scanner's input is an int when using the nextInt() method

There is a **hasNextTYPE()** - for every **nextTYPE()** method.

next() - reads a word from the Scanner

nextLine() - reads a line of text from the Scanner

nextInt() - reads an int value from the Scanner

nextDouble() - reads a double value from the Scanner

nextBoolean() - reads a boolean value from the Scanner

Important: the **tokens** that next(), nextInt(), nextDouble(), and nextBoolean() reads are **separated by whitespace** spaces (" "), **tabs** ("\t") and **newlines** ("\n").

Read a file

```
public class MyFiles {
     public static void main(String[] args) {
               readFile();
      }

     public static void readFile() {
          try {
               File file = new File("read_filename.txt");
               Scanner input = new Scanner(file);
               while (input.hasNextLine()) {
                    String line = input.nextLine();
                    System.out.println(line);
               }
               input.close();//releases the file from your program

          } catch (FileNotFoundException e) {
               System.out.println("File not found.");
               e.printStackTrace();
          }
     }
}
```