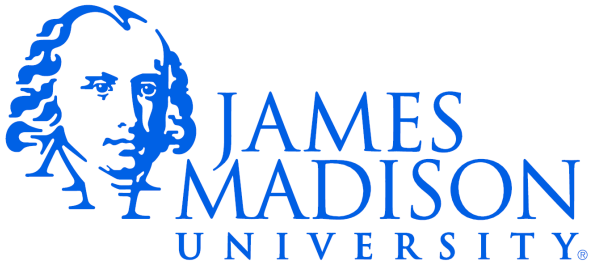
CS159



What is a File?

- A named unit of persistent storage
- What is stored in a file?
  - Bits: 0's and 1's
  - Organized into bytes (8-bit units).
- The binary data has no inherent meaning. We select an interpretation/encoding

Text Files

- Binary data that encodes a sequence of characters
  - Many possibilities:
    - ASCII – 7 bits, 128 possible characters
    - UTF-8 - “8-bit Unicode”
      - Superset of ASCII, uses 8 bits for ASCII characters, more bits when necessary (international alphabets etc.)
    - Etc.
- Examples

Binary Files

- Everything else.
- Any encoding that *isn't* human-readable in a text editor.
- Example of a non-text binary format:
  - IEEE 754

PrintWriter

- java.io.PrintWriter API
- Example:

```
public static void fileDemo() throws FileNotFoundException
{
    PrintWriter pw = new PrintWriter("tmp.txt");

    pw.println("Hey there!");
    pw.close();
}
```

I/O Streams

- Java I/O Stream Tutorial
- You have worked with streams before...
  - System.in is an InputStream
  - System.out is a PrintStream (Which is a type of OutputStream)

Appending to Files: FileWriter

- java.io.FileWriter API
- Example:

```
public static void appendDemo() throws IOException
{
    FileWriter fw = new FileWriter("tmp.txt", true);
    PrintWriter pw = new PrintWriter(fw);

    pw.println("Everybody.");
    pw.close();
}
```

Reading Files: Scanner

- java.util.Scanner API
- java.io.File API

```
public static void readDemo() throws FileNotFoundException
{
    File file = new File("tmp.txt");
    Scanner sc = new Scanner(file);

    while (sc.hasNext())
    {
        System.out.println("THE NEXT THING:");
        System.out.println(sc.next());
    }
}
```

Quiz

- What will be printed when this code executes?

```
public static void readQuiz()
{
    Scanner sc = new Scanner("tmp.txt");

    while (sc.hasNext())
    {
        System.out.println("THE NEXT THING:");
        System.out.println(sc.next());
    }
}
```

"Tokenizing" Strings

- Often need to individual data elements from a string:
  - "Bob salary:100,000 age:22"
- Three common approaches:
  - Scanner nextX methods – Flexible, but requires a method call for each token
  - String split method – Convenient for sequences separated with a standard delimiter
  - StringTokenizer – Existed before the other approaches were introduced. Clunky.

> "StringTokenizer is a legacy class that is retained for compatibility reasons although its use is discouraged in new code. It is recommended that anyone seeking this functionality use the split method of String or the java.util.regex package instead."

Regular Expressions

- Both split and Scanner objects use “regular expressions” to specify delimiters
- java.util.regex.Pattern API

```
public static void scannerDelimiterDemo()
{
    String input = "Bob salary:100,000 age:22";

    Scanner sc = new Scanner(input);

    System.out.println("Name:" +  sc.next()); //Should be "Bob"

    sc.useDelimiter("[:\\s+]"); // Colon OR whitespace.

    System.out.println("Field:" +  sc.next()); //Should be "salary"
    System.out.println("Value:" +  sc.next()); //Should be "100,000"
    System.out.println("Field:" +  sc.next()); //Should be "age"
    System.out.println("Value:" +  sc.next()); //Should be "22"
}
```