Object-Oriented Programming

Session: Week 5 Session 1
Instructor: Eric Pogue

Agenda:

1. Review this week's programming Assignment
2. Introduce the week's Learning Objectives
3. Topics & Examples

FastPrime… plus Questions

Write a performance optimized command line Java application that will programmatically find prime numbers [link] and store those numbers sorted in an output file.

In FastPrime we will create a command line Java application that will:

1. Use multiple threads to find the prime numbers between two numbers
2. Sort those results and store them to a file
3. Perform some timings
4. … And do this all very fast

See the details in this week's assignment

Application performance is ALWAYS a challenge. Learn how to optimize, test, and enhance the performance regularly. It can't be built in at the end of a project!

Learning Objectives – Week 5

1. Understand how Java uses **files** for input and output (IO)
2. Design and implement a controller class to **serialize** (reads & writes) data to a text file
3. Understand **threads** and how to develop and optimize multi-treaded Java applications
4. Understand Java **packages** and compile a class so that it belongs to a particular package
5. Import a class you write that belongs to a particular package.
6. Identify reasons for using JAR (Java ARchive) files to group together related java classes
7. Create a JAR file that stores the contents of a particular package
8. Explain software **testing** terms including unit, integration, user acceptance, performance testing, manual, automated, verification, validation, etc.
9. Understand the importance of testing and the criticality of finding/fixing defects early
10. Explain the purpose, syntax, and annotations of the various assert statements **JUnit** supports
11. Install JUnit onto your machine and execute a JUnit test on your application

Serialization and Writing/Reading Text Files (IO)

Serialization is an object-oriented programming term that means converting an object to a byte steam usually to be written to or read from a text or binary file.

To write to a text file:

- Create a File object, feeding the file's path to the File class constructor
- Create a FileWriter to access the File
- Create a BufferedWriter to write data to the FileWriter efficiently
- Use BufferedWriter's write and newLine functions to commit the data to the file.

To read from a text file:

- Create a File object, feeding the file's path to the File class constructor
- Attach a Scanner object to it
- Use Scanner's readLine and hasNextLine functions to read the file

A buffered writer or buffered stream efficiently organization reads and writes for optimal disk performance. The danger is that if there is a power failure (or someone accidentally kicks your power strip) you could lose data. It is good to "flush the buffer" at critical times when using a buffered writer on mission critical projects. There is little or no danger in using a buffered reader.

Binary files versus text files.

XML – Text file format for structured data… HTML for data.
JSON – "Simplified" text file format for structured data.
Binary

XML Example

In computing, XML (Extensible Markup Language) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is an open standard that:

- Supports nearly all development languages and platforms
- Allows us to cross between many applications
- Can result in large files
- Supports schema to validate data

```xml
<?xml version="1.0" encoding="UTF-8"?>

<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

JSON Example

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is an open standard that:

- Supports nearly all development languages and platforms
- Allows us to cross between many applications
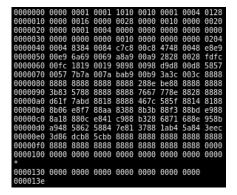- Can result in large files

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    },
    {
      "type": "mobile",
      "number": "123 456-7890"
    }
  ],
  "children": [],
  "spouse": null
}
```

JSON is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is a very common data format used for asynchronous browser/server communication, including as a replacement for XML in some web service style systems.

Binary Files

A binary file is a computer file that is not a text file. The term "binary file" is often used as a term meaning "non-text file". The can be open or closed formats that are generally:

- Fast, small, and efficient*

- Often not very portable across applications and platforms

- Difficult to maintain backward compatibility

```
0000000 0000 0001 0001 1010 0010 0001 0004 0128
0000010 0000 0016 0000 0028 0000 0010 0000 0020
0000020 0000 0001 0004 0000 0000 0000 0000 0000
0000030 0000 0000 0000 0010 0000 0000 0000 0204
0000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
0000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfc
0000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
0000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
0000080 8888 8888 8888 8888 288e be88 8888 8888
0000090 3b83 5788 8888 8888 7667 778e 8828 8888
00000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
00000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
00000c0 8a18 880c e841 c988 b328 6871 688e 958b
00000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
00000e0 3d86 dcb8 5cbb 8888 8888 8888 8888 8888
00000f0 8888 8888 8888 8888 8888 8888 8888 0000
0000100 0000 0000 0000 0000 0000 0000 0000 0000
*
0000130 0000 0000 0000 0000 0000 0000 0000
000013e
```

Some people would say that binary files include all files, and that text files are just binary files that are being interpreted in a specific way.

7

Serialization Example

We are going to demonstrate Writing and Reading OvalDraw data to a proprietary text file by:

- Starting with the OvalDraw application
- Hooking up the "Open" and "Save" menu events
- Teaching our OvalDraw class to Write and Read itself
- Saving our session
- Loading our session

A buffered writer or buffered stream efficiently organization reads and writes for optimal disk performance. The danger is that if there is a power failure (or someone accidentally kicks your power strip) you could lose data. It is good to "flush the buffer" at critical times when using a buffered writer on mission critical projects. There is little or no danger in using a buffered reader.

Binary files versus text files.

XML – Text file format for structured data… HTML for data.
JSON – "Simplified" text file format for structured data.
Binary

End of Session

Course Number: CPSC-24500

Week: 5

Session: 1

Instructor: Eric Pogue

Threads & Multithreaded Applications

Multithreading: A technique by which a single set of code can be used by several processors or cores at different stages of execution.

Mul

I am using the multithreading and multiprocessing terms synonymously for our purposes.

Testing

We test to find defects and/or to validate that we have not introduced new defects.
Defects are exponentially more expensive to fix the longer the exist.
Performance issues are often the most difficult and expensive defects to fix. They are often not found until the application if running under production load… which is often only when it is in production.

Unit - $200
Integration - $600
User Acceptance - $6,000
Production - $100,000+

The permutations of modern software features, data, tools, environments, etc. quickly becomes unmanageable. Testability needs to be goal of nearly all non-trivial applications.

Integration (or Functional) Test: Another kind of test. These test whether the functionality of the entire application works as it should, or that the entire application behaves as expected.

Performance Test: Still another kind of test. These benchmark the performance of an application, especially under high load.

Behavior vs. State Testing: Behavior testing focuses on verifying that the correct functions were called with the correct parameters. In other words, are the various pieces interacting correctly? State testing focuses on the results of those calls. We will focus on state testing.

Manual Testing	Automated Testing
Executing a test cases manually without any tool support is known as manual testing.	Taking tool support and executing the test cases by using an automation tool is known as automation testing.
Time-consuming and tedious – Since test cases are executed by human resources, it is very slow and tedious.	**Fast** – Automation runs test cases significantly faster than human resources.
Huge investment in human resources – As test cases need to be executed manually, more testers are required in manual testing.	**Less investment in human resources** – Test cases are executed using automation tools, so less number of testers are required in automation testing.
Less reliable – Manual testing is less reliable, as it has to account for human errors.	**More reliable** – Automation tests are precise and reliable.
Non-programmable – No programming can be done to write sophisticated tests to fetch hidden information.	**Programmable** – Testers can program sophisticated tests to bring out hidden information.

False positives versus valid defects found.

Manual Testing and Automated Testing can be supportive of each other.