CPSC 501 Assignment 3

Stephen Armstrong

00306924

**Final Code**

**Unit Tests**

**Refactors**

Long Function: Main function was basically all of the code in the Sender.java file so I broke it into several smaller parts. I extracted a method for each of the create Object functions as well as the serializer

**private** **static** **void** serialize(String server, **int** port) **throws** IOException, Exception {

System.*out*.println("Serialize and transfer to receiver?");

System.*out*.println("1. Yes");

System.*out*.println("2. No");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

**if** (input.equals("1")) {

**for** (Object obj : *objList*) {

System.*out*.println("Deserializing object...");

XMLOutputter out = **new** XMLOutputter();

Document doc = Serializer.*serialize*(obj);

File aFile = **new** File("sentdata.xml");

BufferedWriter writer = **new** BufferedWriter(

**new** FileWriter(aFile));

out.output(doc, writer);

writer.close();

System.*out*.println("Transferring file...");

**try** {

Socket s = **new** Socket(server, port);

OutputStream output = s.getOutputStream();

FileInputStream fileInputStream = **new** FileInputStream(

aFile);

**byte**[] buffer = **new** **byte**[1024 \* 1024];

**int** bytesRead = 0;

**while** ((bytesRead = fileInputStream

.read(buffer)) > 0) {

output.write(buffer, 0, bytesRead);

}

fileInputStream.close();

s.close();

System.*out*.println("Transfer Complete");

} **catch** (IOException e) {

System.*out*.println("connection refused");

}

}

} **else** **if** (input.equals("2")) {

} **else** {

System.*out*.println("Invalid input");

}

}

**private** **static** **void** createCollection() **throws** IOException {

System.*out*.println("Enter a list of strings. Eg: 'this,is,a,list'");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

String[] fields = input.split(",");

ArrayList<String> list = **new** ArrayList<String>();

**for** (String s : fields){

list.add(s);

}

Object obj = **new** collectionObject(list);

*objList*.add(obj);

}

**private** **static** **void** createSimpleArray() **throws** IOException {

System.*out*.println("Enter field values. Eg: '1,2,3'");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

String[] fields = input.split(",");

**int**[] ints = **new** **int**[fields.length];

**for** (**int** i = 0; i < ints.length; i++) {

**try** {

ints[i] = Integer.*parseInt*(fields[i]);

} **catch** (Exception e) {

System.*out*.println("You can only enter numbers");

}

}

Object obj = **new** simpleArray(ints);

*objList*.add(obj);

}

**private** **static** **void** createRefArray() **throws** IOException {

System.*out*.println("Enter number of points");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

**int** number = Integer.*parseInt*(input);

List<**int**[]> points = **new** ArrayList<**int**[]>();

System.*out*

.println("Enter x and y values for a point: Eg '1,2'");

**for** (**int** i = 0; i < number; i++) {

bufferRead = **new** BufferedReader(

**new** InputStreamReader(System.*in*));

input = bufferRead.readLine();

String[] xy = input.split(",");

**if** (xy.length != 2) {

System.*out*.println("must enter 2 values");

**break**;

} **else** {

**int**[] \_xy = **new** **int**[2];

\_xy[0] = Integer.*parseInt*(xy[0]);

\_xy[1] = Integer.*parseInt*(xy[1]);

points.add(\_xy);

}

}

Point[] \_points = **new** Point[points.size()];

**for** (**int** i = 0; i < points.size(); i++) {

Point point = **new** Point(points.get(i)[0],

points.get(i)[1]);

\_points[i] = point;

}

Object obj = **new** refArray(\_points);

*objList*.add(obj);

}

**private** **static** **void** createRefObject() **throws** IOException {

System.*out*.println("Enter String");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

Object obj = **new** refObject(input);

*objList*.add(obj);

}

**private** **static** **void** createSimpleObject() **throws** IOException {

System.*out*.println("Enter field values. Eg: '1,true'");

BufferedReader bufferRead = **new** BufferedReader(**new** InputStreamReader(

System.*in*));

String input = bufferRead.readLine();

String[] fields = input.split(",");

**if** (fields.length != 2) {

System.*out*.println("You must enter exactly 2 parameters");

} **else** {

**boolean** bool = **false**;

**if** (fields[1].toLowerCase().equals("true")) {

bool = **true**;

} **else** **if** (fields[1].toLowerCase().equals("false")) {

bool = **false**;

} **else** {

System.*out*

.println("You must enter true or false for a boolean");

}

**try** {

**int** i = Integer.*parseInt*(fields[0]);

Object obj = **new** simpleObject(i, bool);

*objList*.add(obj);

} **catch** (Exception e) {

System.*out*.println("You must enter a number");

}

}

}

I then further broke down the serialize function into even smaller parts creating a function for file creation and file transfer.

**private** **static** File createFile(XMLOutputter out, Document doc)

**throws** IOException {

File aFile = **new** File("sentdata.xml");

BufferedWriter writer = **new** BufferedWriter(

**new** FileWriter(aFile));

out.output(doc, writer);

writer.close();

**return** aFile;

}

**private** **static** **void** transferFile(String server, **int** port, File aFile) {

System.*out*.println("Transferring file...");

**try** {

Socket s = **new** Socket(server, port);

OutputStream output = s.getOutputStream();

FileInputStream fileInputStream = **new** FileInputStream(

aFile);

**byte**[] buffer = **new** **byte**[1024 \* 1024];

**int** bytesRead = 0;

**while** ((bytesRead = fileInputStream

.read(buffer)) > 0) {

output.write(buffer, 0, bytesRead);

}

fileInputStream.close();

s.close();

System.*out*.println("Transfer Complete");

} **catch** (IOException e) {

System.*out*.println("connection refused");

}

}

In Receiver.java I broke up the main function into several functions. I created functions for receiving the files and building the jdom document.

**private** **static** **void** receiveFile(File aFile, Socket s) **throws** IOException,

FileNotFoundException {

InputStream input = s.getInputStream();

FileOutputStream out = **new** FileOutputStream(aFile);

**byte**[] buffer = **new** **byte**[1024 \* 1024];

**int** bytesReceived = 0;

System.*out*.println("receiving file");

**while** ((bytesReceived = input.read(buffer)) > 0) {

out.write(buffer, 0, bytesReceived);

System.*out*.println(bytesReceived + " Bytes received");

**break**;

}

}

**private** **static** Object buildDocument(File aFile) {

SAXBuilder builder = **new** SAXBuilder();

Object obj = **null**;

**try** {

Document doc = (Document) builder.build(aFile);

obj = Deserializer.*deserialize*(doc);

} **catch** (JDOMException e) {

e.printStackTrace();

} **catch** (Exception e) {

e.printStackTrace();

}

**return** obj;

}

**Version Control Logs**