

# Module 6 - Java

## Working with Files

Advanced Java Certification Training

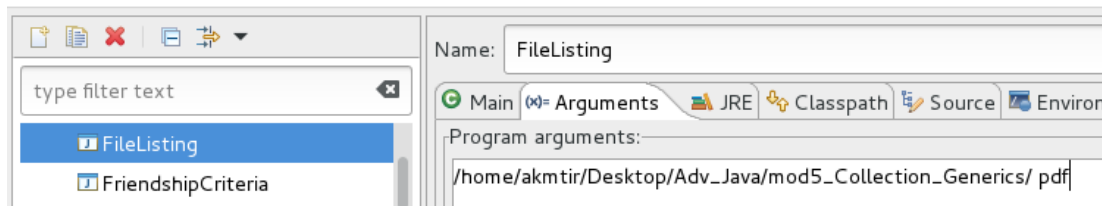
Akram M'Tir

1. Write a programme to search non-recursively for filenames where the user will give the following inputs as command line arguments

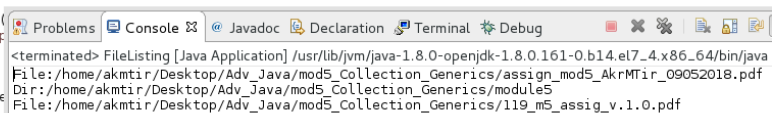
- The search directory
- The file extension

Create, manage, and run configurations

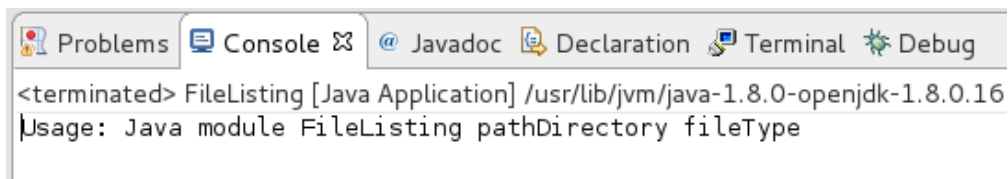
Run a Java application



```
1 package module6;
2
3 import java.io.File;
4 import java.io.FileFilter;
5
6 public class FileListing {
7
8     public void scanFileDirectory(String path, String fileType) {
9
10         File root = new File(path);
11         File[] list = root.listFiles();
12         FileFilter ff = new FileFilter(fileType);
13
14         if (list == null)
15             return;
16
17         for (File f : list) {
18             if (f.isDirectory()) {
19                 System.out.println("Dir:" + f.getAbsolutePath());
20                 scanFileDirectory(f.getAbsolutePath(), fileType);
21             } else {
22                 if (ff.accept(f))
23                     System.out.println("File:" + f.getAbsolutePath());
24             }
25         }
26     }
27
28     public static void main(String[] args) {
29
30         if (args.length != 2) {
31             System.out.println("Usage: Java module FileListing pathDirectory fileType");
32             System.exit(-1);
33         }
34     }
35
36     trv {
```

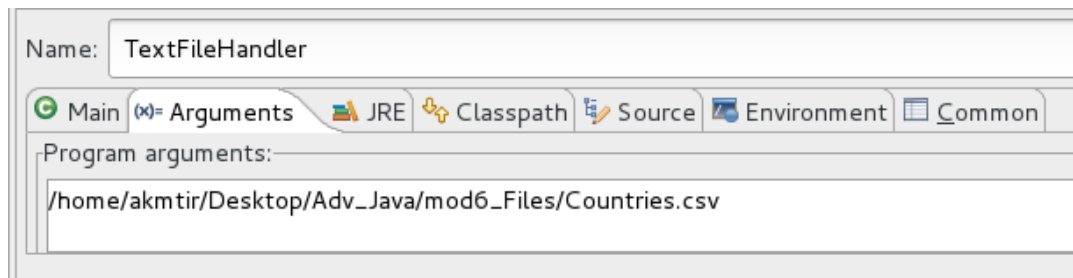


```
<terminated> FileListing [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java
File:/home/akmtir/Desktop/Adv_Java/mod5_Collection_Generics/assign_mod5_AkrMTir_09052018.pdf
Dir:/home/akmtir/Desktop/Adv_Java/mod5_Collection_Generics/module5
File:/home/akmtir/Desktop/Adv_Java/mod5_Collection_Generics/119_m5_assign_v.1.0.pdf
```



```
<terminated> FileListing [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.16
Usage: Java module FileListing pathDirectory fileType
```

2. Write a method in the module6.TextFileHandler to load the file into a map, where key=country and value = capital.



```
72         if (key == null) {
73             key = tokens.nextToken().trim();
74             continue;
75         }
76         val = tokens.nextToken().trim();
77         table.put(key, val);
78     }
79 }
80 }
81 }
82 }
83
84 public static void dumpTable(Map<String, String> map) {
85     Set<String> keys = map.keySet();
86     Iterator<String> it = keys.iterator();
87     while (it.hasNext()) {
88         String key = it.next();
89         System.out.println("key: " + key + " value: " + map.get(key));
90     }
91 }
92
93
94
95 public static void main(String[] args) {
96     if (args.length != 1) {
97         System.out.println("Usage: Java module TextFileHandler <file>");
98         System.exit(-1);
99     }
100
101     try {
102         // TextFileHandler.printFirst10Lines();
103         TextFileHandler.loadFile(args[0]);
104         TextFileHandler.dumpTable(TextFileHandler.getTable());
105     } catch (IOException e) {
106         e.printStackTrace();
107     }
108 }
109
110
111
112
```

Problems Console Javadoc Declaration

<terminated> TextFileHandler [Java Application] /usr/

key: France value: Paris  
key: Spain value: Madrid  
key: UK value: London  
key: Germany value: Berlin  
key: Italie value: Rome  
key: Tunisia value: Tunis  
key: Algerie value: Alger  
key: Maroco value: Casablanca  
key: Belgium value: Brussels  
key: Australia value: Canberra  
key: Canada value: Ottawa  
key: China value: Beijing  
key: Cote d'Ivoire value: Yamoussoukro  
key: Cuba value: Havana  
key: Egypt value: Cairo

3. Write a method in the module6.TextFileHandler to write, to a file, all the country names that starts with a letter passed as an argument, along with their capitals.
- public static void writeToFile(String newFilename, char countryNameBeginningWith) where countryNameBeginningWith = 'S'
  - Name the new file countries\_s.csv

```

100         while (it.hasNext()) {
101             String key = it.next();
102
103             if (key.startsWith(String.valueOf(countryNameBeginningWith))) {
104                 i++;
105                 System.out.println(i + " key: " + key + " value: " + table.get(key));
106                 line = key + "," + table.get(key);
107                 bw.write(line);
108                 bw.write("\n");
109             }
110         }
111     }
112 }
113
114
115
116
117 public static void main(String[] args) {
118     if (args.length != 3) {
119         System.out.println("Usage: Java module TextFileHandler3 cs
120         System.exit(-1);
121     }
122
123     try {
124         TextFileHandler3.loadFile(args[0]);
125         TextFileHandler3.writeToFile(args[1], args[2].charAt(0));
126     } catch (IOException e) {
127         e.printStackTrace();
128     }
129 }
130
131
132
133

```

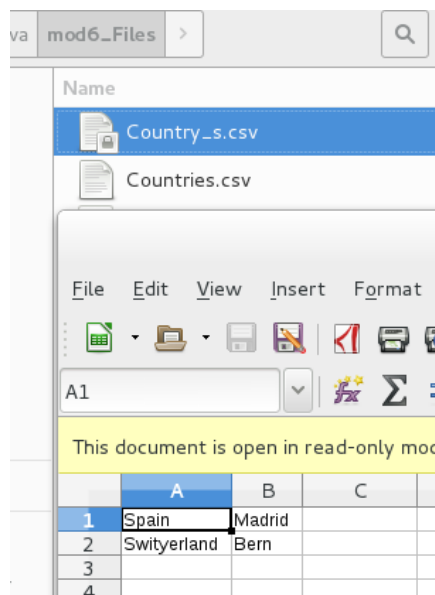
Problem Console Javadoc Declarati

<terminated> TextFileHandler3 [Java Application] /usr/lit

key: Spain value: Madrid  
key: UK value: London  
key: Germany value: Berlin  
key: Italie value: Rome  
key: Tunisia value: Tunis  
key: Algerie value: Alger  
key: Maroco value: Casablanca  
key: Belgium value: Brussels  
key: Australia value: Canberra  
key: Canada value: Ottawa  
key: China value: Beijing  
key: Cote d'Ivoire value: Yamoussoukro  
key: Cuba value: Havana  
key: Egypt value: Cairo  
key: Swityerland value: Bern

-----


1 key: Spain value: Madrid  
2 key: Swityerland value: Bern



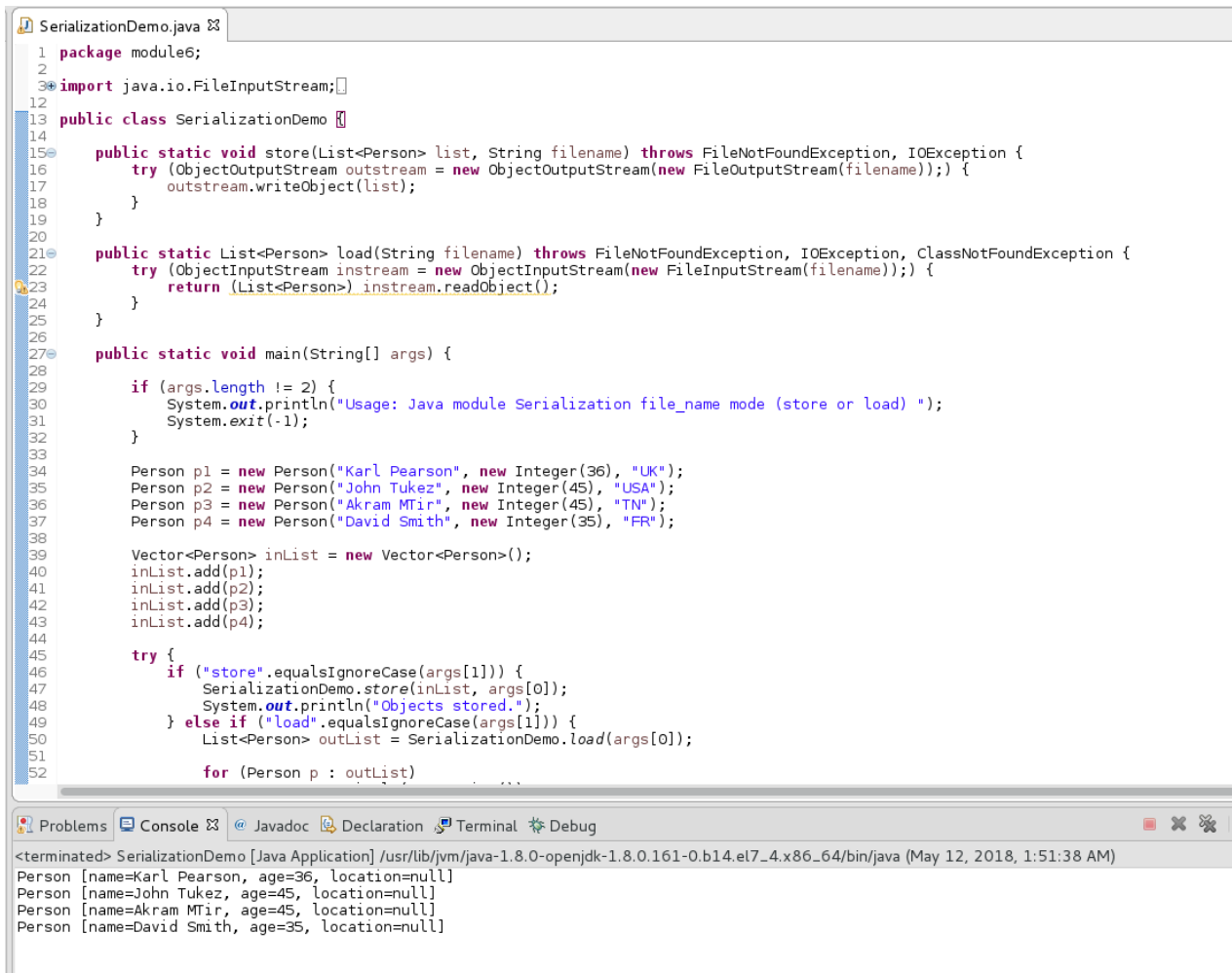
#### 4. For the class module6.SerializationDemo

- Comment out the load(..) method call and just store the list by running the programme.
- Now change the serialVersionUID to something else.
- Comment the store(...) method call, uncomment the load(...) method call and run the programme.

What do you observe?



```
<terminated> SerializationDemo [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 12, 2018, 1:59:52 AM)
java.io.InvalidClassException: module6.Person; local class incompatible: stream classdesc serialVersionUID = -4756619175145079158, local class serialVersionUID = 12345
    at java.io.ObjectStreamClass.initNonProxy(ObjectStreamClass.java:687)
    at java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:1875)
    at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1744)
    at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2032)
    at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1566)
    at java.io.ObjectInputStream.readArray(ObjectInputStream.java:1965)
    at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1560)
    at java.io.ObjectInputStream.defaultReadFields(ObjectInputStream.java:2277)
    at java.io.ObjectInputStream.readSerialData(ObjectInputStream.java:2201)
    at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2059)
    at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1566)
    at java.io.ObjectInputStream.readObject(ObjectInputStream.java:426)
    at module6.SerializationDemo.load(SerializationDemo.java:23)
    at module6.SerializationDemo.main(SerializationDemo.java:50)
```



```
SerializationDemo.java
1 package module6;
2
3 import java.io.FileInputStream;
12
13 public class SerializationDemo {
14
15     public static void store(List<Person> list, String filename) throws FileNotFoundException, IOException {
16         try (ObjectOutputStream outstream = new ObjectOutputStream(new FileOutputStream(filename))) {
17             outstream.writeObject(list);
18         }
19     }
20
21     public static List<Person> load(String filename) throws FileNotFoundException, IOException, ClassNotFoundException {
22         try (ObjectInputStream instream = new ObjectInputStream(new FileInputStream(filename))) {
23             return (List<Person>) instream.readObject();
24         }
25     }
26
27     public static void main(String[] args) {
28
29         if (args.length != 2) {
30             System.out.println("Usage: Java module Serialization file_name mode (store or load)");
31             System.exit(-1);
32         }
33
34         Person p1 = new Person("Karl Pearson", new Integer(36), "UK");
35         Person p2 = new Person("John Tukez", new Integer(45), "USA");
36         Person p3 = new Person("Akram MTir", new Integer(45), "TN");
37         Person p4 = new Person("David Smith", new Integer(35), "FR");
38
39         Vector<Person> inList = new Vector<Person>();
40         inList.add(p1);
41         inList.add(p2);
42         inList.add(p3);
43         inList.add(p4);
44
45         try {
46             if ("store".equalsIgnoreCase(args[1])) {
47                 SerializationDemo.store(inList, args[0]);
48                 System.out.println("Objects stored.");
49             } else if ("load".equalsIgnoreCase(args[1])) {
50                 List<Person> outList = SerializationDemo.load(args[0]);
51
52                 for (Person p : outList) {
53                     System.out.println(p);
54                 }
55             }
56         } catch (Exception e) {
57             e.printStackTrace();
58         }
59     }
60 }
```

```
<terminated> SerializationDemo [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 12, 2018, 1:51:38 AM)
Person [name=Karl Pearson, age=36, location=null]
Person [name=John Tukez, age=45, location=null]
Person [name=Akram MTir, age=45, location=null]
Person [name=David Smith, age=35, location=null]
```

```
SerializationDemo.java
42     inList.add(p3);
43     inList.add(p4);
44
45     try {
46         if ("store".equalsIgnoreCase(args[1])) {
47             SerializationDemo.store(inList, args[0]);
48             System.out.println("Objects stored.");
49         } else if ("load".equalsIgnoreCase(args[1])) {
50             List<Person> outList = SerializationDemo.load(args[0]);
51
52             for (Person p : outList)
53                 System.out.println(p.toString());
54         } catch (IOException | ClassNotFoundException e) {
55             e.printStackTrace();
56         }
57     }
58 }
59
60
61 class Person implements Serializable {
62
63     private static final long serialVersionUID = -4756619175145079158L;
64
65     protected String name = null;
66     protected Integer age = null;
67
68     protected transient String location = null;
69
70     public Person() {
71     }
72
73     public Person(String name, Integer age, String location) {
74         this.name = name;
75         this.age = age;
76         this.location = location;
77     }
78
79     @Override
80     public String toString() {
81         return "Person [name=" + name + ", age=" + age + ", location=" + location + "];"
82     }
83
84 }
85
```

Problems Console Javadoc Declaration Terminal Debug

```
<terminated> SerializationDemo [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/
Person [name=Karl Pearson, age=36, location=null]
Person [name=John Tukez, age=45, location=null]
Person [name=Akram MTir, age=45, location=null]
Person [name=David Smith, age=35, location=null]
```

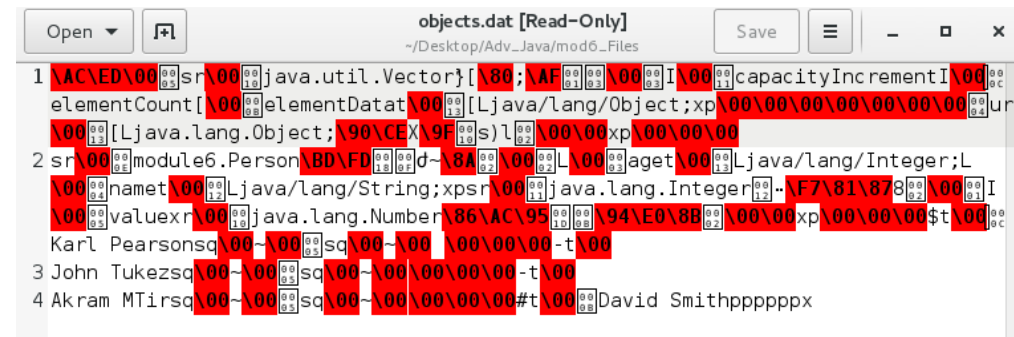
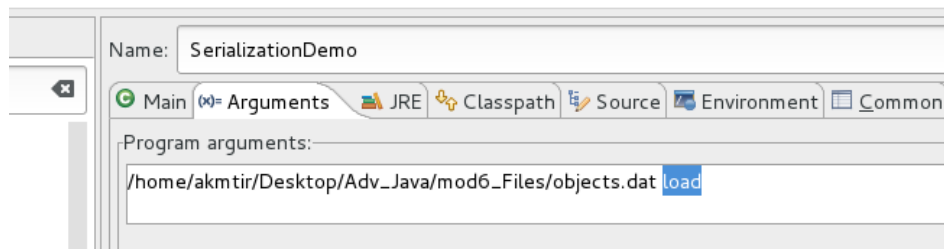
```

60
61 class Person implements Serializable {
62
63     private static final long serialVersionUID = -4756619175145079158L;
64

```

Run Configurations

figurations



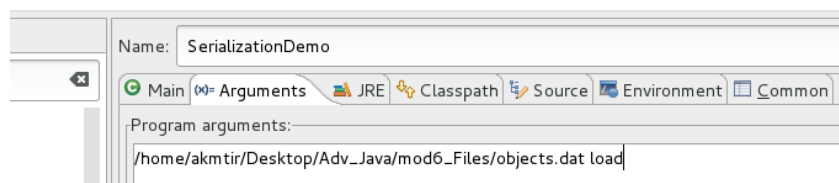
```

45     try {
46         if ("store".equalsIgnoreCase(args[1])) {
47             SerializationDemo.store(inList, args[0]);
48             System.out.println("Objects stored.");
49         } else if ("load".equalsIgnoreCase(args[1])) {
50             List<Person> outList = SerializationDemo.load(args[0]);
51             for (Person p : outList)
52                 System.out.println(p.toString());
53         }
54     } catch (IOException | ClassNotFoundException e) {
55         e.printStackTrace();
56     }
57 }
58
59 }
60
61 class Person implements Serializable {
62
63     private static final long serialVersionUID = 12345; //-4756619175145079158L;
64

```

Run Configurations

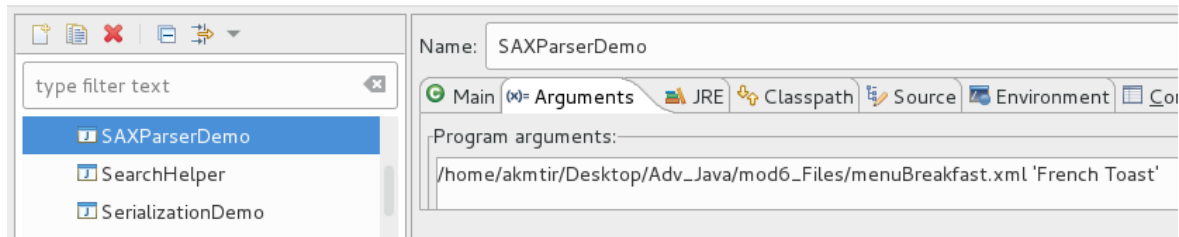
onfigurations



5. For the menu.xml write a programme module6.bkmenu.BKMenuPriceFinder to inquire on the price of a breakfast item. The xml filename and the item name will be supplied by the user.

Create, manage, and run configurations

Run a Java application



```
SAXParserDemo.java  CatalogueUtils.java
1 package module6;
2
3 import java.util.ArrayList;
12
13 public class SAXParserDemo {
14
15     public static void main(String[] args) throws Exception {
16
17         if (args.length != 2) {
18             System.out.println("Usage: XML SAXParser Demo xmlFileName breakfastName");
19             System.exit(-1);
20         }
21         SAXParserFactory saxparser = SAXParserFactory.newInstance();
22         SAXParser parser = saxparser.newSAXParser();
23
24         BreakfastMenuXMLHandler handler = new BreakfastMenuXMLHandler();
25         parser.parse(args[0], handler);
26         //
27         for (BKMenuItem item : handler.menuItems) {
28             if (item.name.equals(args[1]))
29                 System.out.println(item);
30         }
31     }
32 }
33
34 class BreakfastMenuXMLHandler extends DefaultHandler {
35
36     protected List<BKMenuItem> menuItems = new ArrayList<>();
37     protected BKMenuItem item = null;
38     protected String content = null;
39
40     public void startElement(String uri, String localName, String qName, Attributes attributes) throws SAXException {
41
42         switch (qName) {
43             case "food":
44                 item = new BKMenuItem();
45                 break;
46         }
47     }
48
49     @Override
50     public void endElement(String uri, String localName, String qName) throws SAXException {
51
52         switch (qName) {
53             case "food":
54                 menuItems.add(item);
55                 break;
56             case "name":
57                 item.name = content;
58             case "price":
59                 item.price = content;
60             case "descriptor":
61                 item.descriptor = content;
62             case "calories":
63                 item.calories = content;
64         }
65     }
66 }
```

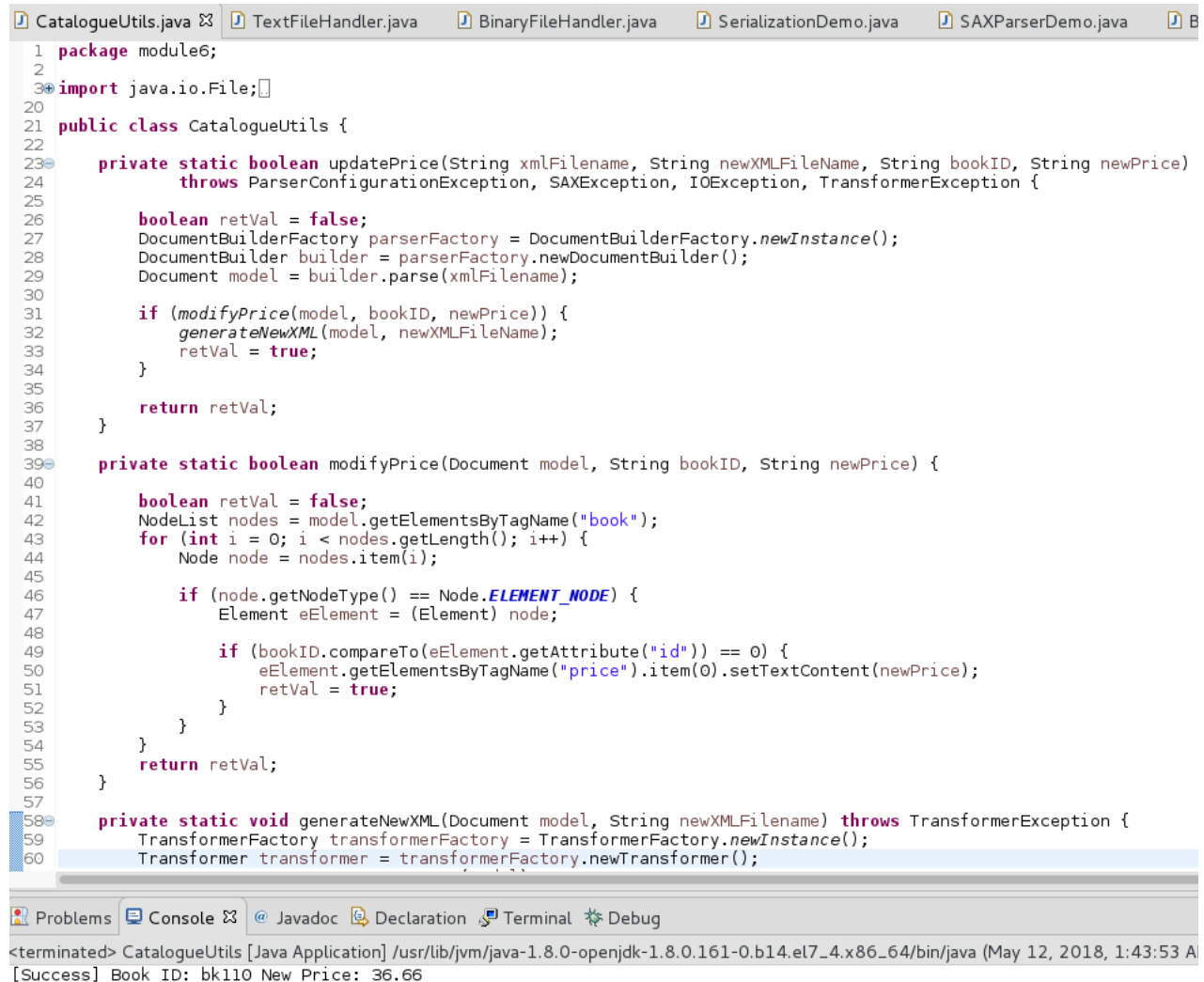
Problems Console Javadoc Declaration Terminal Debug

```
<terminated> SAXParserDemo [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 12, 2018, 2:15:59 AM)
BKMenuItem [name=French Toast, price=$4.50, descriptor=Thick slices made from our homemade sourdough bread, calories=600]
```

6. As an exercise check the other methods of `org.xml.sax.helpers.DefaultHandler` and also the interfaces that it implements.

7. Write a programme to update the price of a book based on the book ID.

- The input xml is `catalogue.xml`
- Don't change the `catalogue.xml`, create a new xml instead.



```
1 package module6;
2
3 import java.io.File;
4
5 public class CatalogueUtils {
6
7     private static boolean updatePrice(String xmlFilename, String newXMLFileName, String bookID, String newPrice)
8         throws ParserConfigurationException, SAXException, IOException, TransformerException {
9
10         boolean retVal = false;
11         DocumentBuilderFactory parserFactory = DocumentBuilderFactory.newInstance();
12         DocumentBuilder builder = parserFactory.newDocumentBuilder();
13         Document model = builder.parse(xmlFilename);
14
15         if (modifyPrice(model, bookID, newPrice)) {
16             generateNewXML(model, newXMLFileName);
17             retVal = true;
18         }
19
20         return retVal;
21     }
22
23     private static boolean modifyPrice(Document model, String bookID, String newPrice) {
24
25         boolean retVal = false;
26         NodeList nodes = model.getElementsByTagName("book");
27         for (int i = 0; i < nodes.getLength(); i++) {
28             Node node = nodes.item(i);
29
30             if (node.getNodeType() == Node.ELEMENT_NODE) {
31                 Element eElement = (Element) node;
32
33                 if (bookID.compareTo(eElement.getAttribute("id")) == 0) {
34                     eElement.getElementsByTagName("price").item(0).setTextContent(newPrice);
35                     retVal = true;
36                 }
37             }
38         }
39         return retVal;
40     }
41
42     private static void generateNewXML(Document model, String newXMLFilename) throws TransformerException {
43         TransformerFactory transformerFactory = TransformerFactory.newInstance();
44         Transformer transformer = transformerFactory.newTransformer();
45     }
46 }
```

Problems Console Javadoc Declaration Terminal Debug

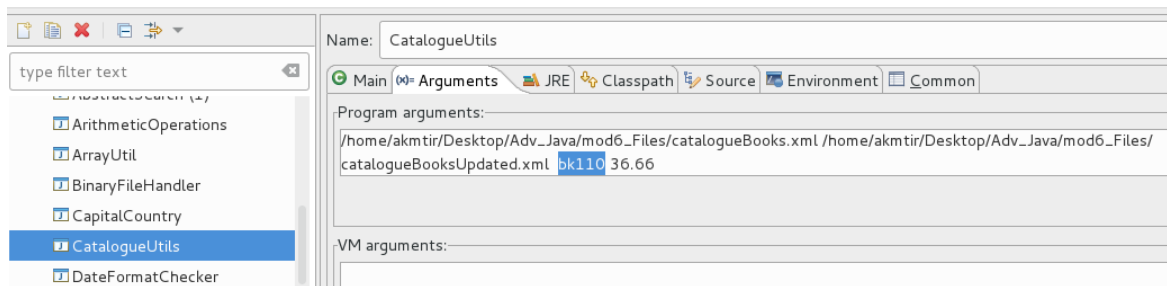
<terminated> CatalogueUtils [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7\_4.x86\_64/bin/java (May 12, 2018, 1:43:53 A  
[Success] Book ID: bk110 New Price: 36.66



```
CatalogueUtils.java TextFileHandler.java BinaryFileHandler.java SerializationDemo.java SAXParserDemo.java
42     NodeList nodes = model.getElementsByTagName("book");
43     for (int i = 0; i < nodes.getLength(); i++) {
44         Node node = nodes.item(i);
45
46         if (node.getNodeType() == Node.ELEMENT_NODE) {
47             Element eElement = (Element) node;
48
49             if (bookID.compareTo(eElement.getAttribute("id")) == 0) {
50                 eElement.getElementsByTagName("price").item(0).setTextContent(newPrice);
51                 retVal = true;
52             }
53         }
54     }
55     return retVal;
56 }
57
58 private static void generateNewXML(Document model, String newXMLFilename) throws TransformerException {
59     TransformerFactory transformerFactory = TransformerFactory.newInstance();
60     Transformer transformer = transformerFactory.newTransformer();
61     DOMSource source = new DOMSource(model);
62     StreamResult result = new StreamResult(new File(newXMLFilename));
63     transformer.transform(source, result);
64 }
65
66
67 public static void main(String[] args)
68     throws ParserConfigurationException, SAXException, IOException, TransformerException {
69
70     if (args.length != 4) {
71         System.out.println("Usage: Java XML DOMParserAlterPriceDemo xml_File_Name book_id new_price");
72         System.exit(-1);
73     }
74
75     boolean retVal = CatalogueUtils.updatePrice(args[0], args[1], args[2], args[3]);
76
77     if (retVal)
78         System.out.println("[Success] Book ID: " + args[2] + " New Price: " + args[3]);
79     else
80         System.out.println("[Failure] Book ID: " + args[2] + " does not exist.");
81 }
82
83 }
84
85
Problems Console Javadoc Declaration Terminal Debug
<terminated> CatalogueUtils [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 12, 2018, 1:3
[Success] Book ID: bk110 New Price: 36.66
```

## Create, manage, and run configurations

Run a Java application



catalogueBooks.xml	catalogueBooksUpdated.xml
89	
90	
91	
92	
93	
94	
95	
96	
97	
98	

```

89  </book>
90  <book id="bk110">
91    <author>O'Brien, Tim</author>
92    <title>Microsoft .NET: The Programming Bible</title>
93    <genre>Computer</genre>
94    <price>36.95</price>
95    <publish_date>2000-12-09</publish_date>
96    <description>Microsoft's .NET initiative is explored in
97    detail in this deep programmer's reference.</description>
98  </book>

```

catalogueBooks.xml	catalogueBooksUpdated.xml
85	
86	
87	
88	
89	
90	
91	
92	
93	
94	
95	
96	
97	

```

85  <description>After an inadvertent trip through a Heisenberg
86  Uncertainty Device, James Salway discovers the problems
87  of being quantum.</description>
88  </book>
89  <book id="bk110">
90    <author>O'Brien, Tim</author>
91    <title>Microsoft .NET: The Programming Bible</title>
92    <genre>Computer</genre>
93    <price>36.66</price>
94    <publish_date>2000-12-09</publish_date>
95    <description>Microsoft's .NET initiative is explored in
96    detail in this deep programmer's reference.</description>
97  </book>

```