COSC 3P91

Lab 6

In this lab you are going to use XML in an example. The XML schema files needed for this lab are provided separately.

1. Add the `COSC3P91.jar` file as a library to your project so that you can use the classes and methods discussed in class. Add the XSD folder to your project, i.e., add the folder to the project folder.
2. Implement a class `Person` that has the following fields `firstName`, `middleName`, `lastName` of type `String`, `age` of type `int`, and `address` of type `String`. Add an appropriate constructor and make the class an instance of XMLObject. You can use plain strings to implement `toXMLString()`, i.e., you do not need to use a `StringBuilder`. Refer to the example below and the XML schema file for the syntax of a person in XML.
   Example of a `Person` in XML:
   ```
   <Person firstName="Holden" middleName="James"
   lastName="Ridley" age="28" address="15 Main Street"/>
   ```
3. Implement a class `XMLNodeConverterPerson` that is an `XMLNodeConverter` for `Person`. The `convertXMLNode(Node node)` method should return `null` if the `node` is not a node of a `Person`. Make this class a **Singleton**, i.e., use the **Singleton** design pattern for this class.
4. Implement a static method `read(Reader source)` in `Person` that returns a `Person` based on the XML input `source`.
5. Test your implementation. For this set the XSD path of the `XMLReader` in the `main` method appropriately. If your XSD folder is in the same folder `src`, then it should be `XMLReader.setXSDPath("./XSD/");`
6. Implement a class `PersonList` that is an `ArrayList` of `Persons`. Make this class an instance of `XMLObject`.
   Example of a `PersonList` in XML:
   ```
   <PersonList>
   <Person firstName="John" middleName="" lastName="McDonald"
   age="42" address="12 Riverside Drive"/>
   ```

```
<Person firstName="Michael" middleName="J" lastName="Fox"
age="63" address="unkown"/>
<Person firstName="Holden" middleName="James"
lastName="Ridley" age="28" address="15 Main Street"/>
</PersonList>
```

7. Implement a class `XMLNodeConverterPersonList` that is an `XMLNodeConverter` for `PersonList`. The `convertXMLNode(Node node)` method should return `null` if the `node` is not a node of a `PersonList`. Make this class a **Singleton**, i.e., use the **Singleton** design pattern for this class.
8. Implement a static method `read(Reader source)` in `PersonList` that returns a `PersonList` based on the XML input `source`.
9. Test your implementation.
10. Implement a enumeration type `DepartmentType` that has the three values `ACADEMIC`, `ADMINISTRATIVE`, and `OTHER`.
11. Implement a class `Department` that has the following fields `depName` of type `String`, `depMembers` of type `PersonList`, and `depType` of type `DepartmentType`. Add an appropriate constructor and make the class an instance of XMLObject.
    Example of a `Department` in XML:
    ```
    <Department name="COSC" type="ACADEMIC">
    <PersonList>
    <Person firstName="John" middleName="" lastName="McDonald"
    age="42" address="12 Riverside Drive"/>
    <Person firstName="Michael" middleName="J" lastName="Fox"
    age="63" address="unkown"/>
    <Person firstName="Holden" middleName="James"
    lastName="Ridley" age="28" address="15 Main Street"/>
    </PersonList>
    </Department>
    ```
12. Implement a class `XMLNodeConverterDepartment` that is an `XMLNodeConverter` for `Department`. The `convertXMLNode(Node node)` method should return `null` if the `node` is not a node of a `Department`. Make this class a **Singleton**, i.e., use the **Singleton** design pattern for this class.
13. Implement a static method `read(Reader source)` in `Department` that returns a `Department` based on the XML input `source`.
14. Test your implementation.