**Lab 7**

The aim of this lab is to develop a shopping cart.

What you should already have:

* A product listing
* A page showing a single product

Customers can select a single product and place it in the shopping cart.

The shopping cart (at the moment) persists for a session.

**The class Sku**

This class represents an item or product (sku = stock keeping unit). The instance variables represent the properties of a sku, e.g. id, title, description, image, price (use Big Decimal).

The constructor is used to create a Sku object. It is useful to that products have an id, e.g. in order to distinguish between two products. There should be a constructor that can be used to create a new product provided an id argument.

Getter and Setter methods for the instance variables, you can use annotations to create these or write these by hand.

A method that returns the price as a String.

Skus have a price, a typical value could be 2.99. double and float aren’t suitable data types for a price. Read [1] and [2]. Why should double and float not be used to represent currency? Use BigDecimal to represent a price.

We need to compare two Sku objects; in this case two Skus are identical if they have the same id value.

Override the equals() and hashCode() methods accordingly.

Read [3] to refresh your memory of the hashCode() and equals() methods.

**The class Shop**

For this task check that in IntelliJ check that the GenerateSerialVersionUID plugin (File|Settings|Plugins|Browse repository) is installed.

Class Shop represents a “shop” from which “sku”s can be selected and placed in a cart.

The Shop class has methods to create a shop from a data source (a JSON file in the first instance). Most importantly it has a Set of Sku’s.

**public** Shop load() {  
 **try** {  
 Shop shop = Shop.*fromJSONString*(IOUtil.*loadStringUTF8*(getClass().getResource(**"/shop/sample-shop.json"**))

**return** shop;  
 } **catch** (IOException e) {  
 **throw new** RuntimeException(e);  
 }  
}

The Shop should be executed (started) by a request to a URL that is passed to a servlet to handle. Create such a servlet.

Later, in order to run it, you need to create a context handler for the new servlet in the Runner class.

**Load a shop from JSON**

The data for the individual Skus comes from a file in JSON (JavaScript Object Notation) notation, see [4] or the notes from lecture 7 for further details.

You need to modify the structure of the JSON file to match the structure (in particular the instance variables) of your Sku (or Product or Item) class.

Refer to [4] for further details on the notation. Essentially, a JSON file is a text file in which the data is marked up as “name : value” pairs, where values can be Strings, numbers, arrays, objects Boolean values.

The Jackson API [5] [6] provides classes for reading from and writing to JSON, see import statements. In order to load the libraries you need to add at least the following dependencies to the pom.xml file:

<**dependency**>  
 <**groupId**>com.fasterxml.jackson.core</**groupId**>  
 <**artifactId**>jackson-core</**artifactId**>  
 <**version**>${jackson.version}</**version**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>com.fasterxml.jackson.core</**groupId**>  
 <**artifactId**>jackson-annotations</**artifactId**>  
 <**version**>${jackson.version}</**version**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>com.fasterxml.jackson.core</**groupId**>  
 <**artifactId**>jackson-databind</**artifactId**>  
 <**version**>${jackson.version}</**version**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>com.fasterxml.jackson.datatype</**groupId**>  
 <**artifactId**>jackson-datatype-guava</**artifactId**>  
 <**version**>${jackson.version}</**version**>  
</**dependency**>

Ensure that the JSON file is placed in the location referenced in Shop (here the reference refers to src\main\resources\shop\sample-shop.json.

**Further reading:**

[1] https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

[2] <http://www.java2s.com/Tutorials/Java/Data_Type/BigDecimal/Java_Tutorial_Java_BigDecimal.htm>

[3]

<http://www.javaworld.com/article/2074996/hashcode-and-equals-method-in-java-object---a-pragmatic-concept.html>

[4] http://www.json.org/

[5] https://github.com/FasterXML/jackson

[6] https://github.com/FasterXML/jackson-docs

**Example class Sku.java**

**package**…;  
  
@Data  
@JsonInclude(JsonInclude.Include.***NON\_NULL***)  
@Accessors(chain=**true**)  
@SuppressWarnings({**"unused"**})  
**public class** Sku **implements** Serializable {  
 @SuppressWarnings(**"unused"**)  
 **static final** Logger ***LOG*** = LoggerFactory.*getLogger*(Sku.**class**);  
 **private static final long *serialVersionUID*** = -4119412393370563146L;  
  
 */\*\* SKUs are unique on their ids. Should check when reading from JSON to make sure this constraint is observed \*/* **private** String **id**;  
 **private** String **title**;  
 **private** String **description**;  
 **private** SkuImage **thumb**;  
 **private** SkuImage **image**;  
 **private** BigDecimal **unitPrice**;  
  
 **private** Sku() {  
 **this**(**""**);  
 }  
  
 **public** Sku(@NonNull String id) {  
 **this**.**id** = id;  
 }  
  
 **public** String unitPriceString() {  
 **return** NumberFormat.*getCurrencyInstance*().format(getUnitPrice());  
 }  
  
 **public** String toJsonString() {  
 **try** {  
 **return new** ShopMapper().writeValueAsString(**this**);  
 } **catch** (IOException e) {  
 **throw new** RuntimeException(e);  
 }  
 }  
  
 @Override  
 **public int** hashCode() {  
 **return id**.hashCode();  
 }  
  
 @Override  
 **public boolean** equals(Object o) {  
 **if** (o **instanceof** Sku) {  
 Sku oSku = (Sku)o;  
 **return id**.equals(oSku.**id**);  
 }  
 **return false**;  
 }  
  
}

**Example class Shop.java**

**package** …;  
  
**import** com.fasterxml.jackson.annotation.JsonInclude;  
**import** lombok.Data;  
**import** lombok.NonNull;  
**import** org.slf4j.Logger;  
**import** org.slf4j.LoggerFactory;  
  
**import** java.io.IOException;  
**import** java.io.Serializable;  
**import** java.util.LinkedHashSet;  
**import** java.util.Set;  
  
@Data  
@JsonInclude(JsonInclude.Include.***NON\_NULL***)  
@SuppressWarnings(**"unused"**)  
**public class** Shop **implements** Serializable {  
 **static final** Logger ***LOG*** = LoggerFactory.*getLogger*(Shop.**class**);  
 **private static final long *serialVersionUID*** = -4478190721183244953L;  
  
 **private** String **shopName**;  
 **private** Set<Sku> **skus**;  
  
 **public static** Shop fromJSONString(@NonNull String s) {  
 **try** {  
 **return new** ShopMapper().readValue(s, Shop.**class**);  
 } **catch** (IOException e) {  
 **throw new** RuntimeException(e);  
 }  
 }  
  
 **public** Shop() {  
 **skus** = **new** LinkedHashSet<>();  
 }  
  
 **public** Shop(@NonNull String storeName) {  
 **this**();  
 **this**.**shopName** = storeName;  
 }  
  
 **public** Shop(@NonNull Shop shop) {  
 **this**.**shopName** = shop.**shopName**;  
 **this**.**skus** = **new** LinkedHashSet<>(shop.**skus**);  
 }  
  
 **public** Shop addSku(@NonNull Sku sku) {  
 **skus**.add(sku);  
 **return this**;  
 }  
  
  
 **public** Sku getSku(@NonNull String id) {  
 **for** (Sku sku : **skus**) {  
 **if** (id.equals(sku.getId())) {  
 **return** sku;  
 }  
 }  
 **return null**;  
 }  
  
 **public** String toJSONString() {  
 **try** {  
 **return new** ShopMapper().writeValueAsString(**this**);  
 } **catch** (Exception e) {  
 **throw new** RuntimeException(e);  
 } } }

**Example class Shop Mapper**

**package**…;  
  
**import** com.fasterxml.jackson.annotation.JsonAutoDetect;  
**import** com.fasterxml.jackson.databind.DeserializationFeature;  
**import** com.fasterxml.jackson.databind.ObjectMapper;  
**import** com.fasterxml.jackson.databind.module.SimpleModule;  
**import** com.fasterxml.jackson.databind.ser.std.ToStringSerializer;  
**import** org.slf4j.Logger;  
**import** org.slf4j.LoggerFactory;  
  
**import** java.math.BigDecimal;  
  
  
**class** ShopMapper **extends** ObjectMapper {  
 @SuppressWarnings(**"unused"**)  
 **static final** Logger ***LOG*** = LoggerFactory.*getLogger*(ShopMapper.**class**);  
 **private static final long *serialVersionUID*** = -3817083726149967662L;  
  
 ShopMapper() {  
 SimpleModule module = **new** SimpleModule();  
 module.addSerializer(BigDecimal.**class**, **new** ToStringSerializer());  
 registerModule(module);  
 setVisibility(getVisibilityChecker().withFieldVisibility(JsonAutoDetect.Visibility.***ANY***));  
 configure(DeserializationFeature.***FAIL\_ON\_UNKNOWN\_PROPERTIES***, **false**);  
 }  
}

**Example JSON data file**

{  
 **"shopName"**: **"Botanic Gardens Shop"**,  
 **"storeCurrency"**: **"GBP"**,  
 **"skus"**: [  
 {  
 **"id"**: **"0715320211"**,  
 **"title"**: **"Hillier Gardener's Guide to Trees and Shrubs"**,  
 **"description"**: **"Unique in its detailed and comprehensive coverage of temperate-zone trees and shrubs, and backed by the world-famous Hillier name, this book provides great expertise in a highly illustrated, eminently usable package."**,  
 **"thumb"**: {  
 **"width"**: 57,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I/61Bf4KgTcxL.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 381,  
 **"height"**: 500,  
 **"url"**: **"/images/shop/images/I/61Bf4KgTcxL.jpg"** },  
 **"unitPrice"**: **"20"** },  
 {  
 **"id"**: **"0881929743"**,  
 **"title"**: **"Conifers of the World: The Complete Reference"**,  
 **"description"**: **"Researched for more than three decades, this definitive work provides up-to-date descriptions of all the true conifers of the world, including 545 species of trees and shrubs. Written for accessibility to both horticultural and botanical audiences, it is the first comprehensive update of conifer taxonomy in nearly a century.\n\n Noted conifer taxonomist James E. Eckenwalder discusses the relationships among the groups, practical usages, champion trees, fossil occurrences, and biology.\n\n New identification guides for the families and genera are based whenever possible on foliage features and thus should be easier to use than traditional conifer keys, which focus on seasonal, and often microscopic, cone characters.\n\n Eckenwalder shares the reasoning behind his taxonomic decisions, many of which are unique to this book, reflecting a comprehensive reevaluation of conifer classification. He also outlines the features sought in cultivars of each genus, particular cultivation concerns, and conifers recommended for cultivation under various conditions and to achieve different effects.\n\n Some 3,000 cultivars have been available in recent times, more than five times the total number of conifer species. Several hundred original illustrations include drawings of the seed cones for all genera as well as for representative species.\n\n Maps of the natural distribution of each genus allow for easy comparison of ranges. Handsome black-and-white photographs of species in their natural habitats and attractive color photos further enrich the volume. More than 100 images reproduce foliage of many genera as an aid in identification.\n\n With its unprecedented attention to detail and extensive bibliography, this major work is an essential reference for botanists, naturalists, and horticulturists."**,  
 **"thumb"**: {  
 **"width"**: 56,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I//413IMKVLPwL.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 389,  
 **"height"**: 500,  
 **"url"**: **"/images/shop/images/I/61PCBvxbHGL.jpg"** },  
 **"unitPrice"**: **"45"** },  
 {  
 **"id"**: **"1908931515"**,  
 **"title"**: **"The Story of Glasgow's Botanic Gardens"**,  
 **"description"**: **""**,  
 **"thumb"**: {  
 **"width"**: 53,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I/619FmMeAGNL.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 354,  
 **"height"**: 500,  
 **"url"**: **"/images/shop/images/I/619FmMeAGNL.jpg"** },  
 **"unitPrice"**: **"14.99"** },  
 {  
 **"id"**: **"0300096747"**,  
 **"title"**: **"Glasgow (Pevsner Architectural Guides: Buildings of Scotland)"**,  
 **"description"**: **"Glasgow has a wide array of architectural treasures: the greatest medieval cathedral in Scotland; fragments of a seventeenth- and eighteenth-century 'merchant city'; the well-preserved heart of a planned new town, Blythswood; a city centre dense with Victorian and Edwardian commercial buildings; stately nineteenth-century terraces lining the Great Western Road and picturesquely crowning Woodlands Hill; opulent villas in suburbs like Pollokshields and Kelvinside; and streets of tenements from the workaday to the grand.\n\n The twentieth century has encircled the city with a broad belt of public housing, and this too has a fascinating history that encompasses garden suburbs, early experiments in high-rise, comprehensive redevelopments and new interpretations of the tenement tradition. Charles Rennie Mackintosh and Alexander 'Greek' Thomson are, of course, internationally known, but the exceptional talents of Glasgow's many other architects, such as Charles Wilson, James Salmon Jr. and Jack Coia, have helped to shape the city's distinctive character."**,  
 **"thumb"**: {  
 **"width"**: 41,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I/41T5PFESN3L.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 261,  
 **"height"**: 475,  
 **"url"**: **"/images/shop/images/I/41T5PFESN3L.jpg"** },  
 **"unitPrice"**: **"35"** },  
 {  
 **"id"**: **"1902831047"**,  
 **"title"**: **"Kibble's Palace"**,  
 **"description"**: **"Eric Curtis, former keeper of the Botanic Garden, has researched the beginnings in the life of Glasgow entrepreneur and eccentric, John Kibble.'"**,  
 **"thumb"**: {  
 **"width"**: 48,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I/515TMJY1SWL.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 321,  
 **"height"**: 500,  
 **"url"**: **"/images/shop/images/I/515TMJY1SWL.jpg"** },  
 **"unitPrice"**: **"7.99"** },  
 {  
 **"id"**: **"117278583X"**,  
 **"title"**: **"Trees and shrubs hardy in the British Isles"**,  
 **"description"**: **"This is a reproduction of a book published before 1923. This book may have occasional imperfections...."**,  
 **"thumb"**: {  
 **"width"**: 55,  
 **"height"**: 75,  
 **"url"**: **"/images/shop/images/I/31vmBJ9TNaL.\_SL75\_.jpg"** },  
 **"image"**: {  
 **"width"**: 381,  
 **"height"**: 500,  
 **"url"**: **"/images/shop/images/I/51xIyIq%2BClL.jpg"** },  
 **"unitPrice"**: **"23.99"** }  
 ]  
}