Spring Exercises

Exercise 1: The BeanFactory

In this exercise you will need to implement a List<Item> sorter that corresponds to the following interface:

```
public interface Sorter {
        void addItem(Item item);
        void setList(List<Item> unsorted);
        void setComparator(Comparator<Item> comparator);
        void sort();
        List<Item> getItems();
}
```

The method *sort* should sort the list of items according to the Comparator provided. The Item class is a simple Java Bean with the properties: *name* and *price*;

The implementation should be injected with the type of the List: *ArrayList* or *LinkedList* and of the Comparator<Item> which can be *by Name* or *by Price*.

Test your class with the four combinations (*ArrayList* or *LinkedList* against *by Name* or *by Price*).

The test data (meaning items) should be added from the xml files by using the *org.springframework.beans.factory.config.MethodInvokingFactoryBean* bean.

Step 2:

You should change the Sorter implementation class to generate a bunch of Items according to a property: *itemCount*. The items should be generated from a Factory class which will produce items with the name *Item1, Item2, etc.…*

The Sorter class should use a lookup-method called getItemFactory

Note: create the factory as Spring factory bean.

Again, test your class with different configurations. If done correctly, **you should not change the test code in this step**.

EX2: Spring AOP

- Create a benchmark interceptor that will benchmark the Sorter from ex1.
- Because the sorting takes too much time, you will need to provide an OneWay advice that upon invocation of the sort method will run it on a new thread. Use custom pointcut to match only the sort method (on any class).
- Create a Mixin that adds an interface *Sorted* to the Sorter class with the method isSorted() that will return whether the sort method was invoked. Use the *DefaultIntroductionAdvisor*.

Note: You should use autoproxying in this exercise.
Think, is the **order** of the interceptors in this exercise important?

EX3: Spring JDBC

In this exercise you should create a BookstoreAdmin class that should implement the following interface:

```
public interface BookStoreAdmin {
        void addNewBook(Book book);
        List<Book> showBooks();
        List<Book> showBooksBellow(double price);

        void addCustomer(Customer customer);
        List<Customer> showCustomers();
        List<Customer> showCustomersByName(String name);

}
```

Book and Customer classes represent the BOOKS and CUSTOMERS tables respectively.

Your class should extend the JdbcDaoSupport class.

EX4: Spring Hibernate

Change your DAO implementation from the previous exercise to use Hibernate.

Note that your test code should not be changed!