ASK

Alten Shopping $\genfrac{}{}{0pt}{}{\text{K}}{\text{C}}$art

Guillermo Rodolfo Ellison

August 5, 2015

INTRODUCTION

ASK is the name of the project. It is based on a pseudo acronym that stands for Alten Shopping K(C)art. Just a marketing idea that could have a bigger commercial impact than ASC.

The purpose of this document is showing a technical background on which the project is constructed. The document also gives precise indications to use the application.

After the introduction there is accurate information that guides through the start up and utilization process. Then the document exposes a set of chosen technologies on which the project has been built. And the conclusion, while making some specific references to the project, is mostly oriented to the general working context at the heart of big software enterprises.

USING THE APPLICATION

1. Launching

The application can be run with:

```
java -jar ask-0.0.1.jar
```

And it can be visualized in a web browser on:

`http://localhost:8080/`

Port 8080 must be free. Enabled username/password credentials are: **c1/c1**, **c2/c2** and **c3/c3**.

2. Technical tips

All classes in the project have a prefix ask, the name of the project, to ensure that applying a filter which starts with “ask” only the project-specific classes are going to be found.

The main class is in the src/main/java folder and its name is com.alten.ask.AskMainApplication.

The project is divided into three main paths: src/main/java, src/main/resources and src/main/webapp. The first contains the java source code, the second is dedicated to the resources like images and properties, and the third includes the CSS styles. This is a standard Maven structure.

3. Bugs

Sometimes the application loses the H2 Database connection. In this case the only solution is to stop/start again.

The application could throw an exception like:

java.io.FileNotFoundException: [...]/tomcat-docbase.[...]/VAADIN/themes/ask/styles.scss.cache[...]

It is a Vaadin bug, the styles should work anyway. Please refer to:

`https://github.com/vaadin/spring/issues/13` for more information.

4. User requirements

These are the operations that an existing user can perform with the application:

a. Log in/log out
b. Change language
c. Add product to cart
d. Go to cart
e. Remove product from cart
f. Purchase the current invoice of the cart
g. Go to invoices
h. See invoice’s details
i. Go to transactions history
j. See transaction’s details

TECHNOLOGIES

5. Maven

This is the build automation technology of the project. Within the pom.xml, which defines a Maven project, there are some dependencies for Spring Boot, Spring Data JPA, H2 Database, Java Mail, Vaadin and JUnit.

6. Spring – Spring Framework – Spring Boot

The Enterprise layer is managed by Spring. The core and the boot are the main modules included, while the core brings the minimum set of libraries needed, the boot brings an embedded Tomcat and allows the entire project to be used from a jar, so neither an Application Server nor a Web Container are required. Because of its higher grade of modularization and its possibility of being run without an Application Server, Spring projects can be run/deployed in many different ways.

7. Vaadin

It is the chosen client and GUI technology. Vaadin is an extension and an improvement of GWT, so it has a wrapped GWT engine inside. GWT, Google Web Toolkit, initially developed by Google and then transformed into an open source community, is based on two great ideas. The first enables a java developer to work only with java and then the GWT engine translates the java source code into HTML plus CSS plus JavaScript, so the java developer does not need to be an expert in other technologies. The second idea is that a developer needs to debug the code in order to be able to write complex applications. So the GWT engines enables debugging with watches and break points by translates the java code into JavaScript. Finally, Vaadin inherits the GWT strengths and improve them by taking care of the GWT gap between the client and the server, and Vaadin also provides a wide and open repository of components.

The connection with Spring can be seen in the AskVaadinUI class, which is the first Vaadin class being run. This class is annotated as @SpringUI, so it is injected by the core of Spring.

8. Vaadin Grid

Threre is a class called AskCompGrid. It is Vaadin Grid wrapper and it is designed to be reusable, so it includes no project logic. This class can be easily ported to another projects and it contains the only grid of the project, then, all HTML tables that can be seen in the browser come from an instance of this class.

9. JUnit

For testing it has been chosen JUnit. The default project test can be run with:

```
java -jar ask-0.0.1.jar test
```

And to force a failing test it can be run:

```
java -jar ask-0.0.1.jar test fail
```

In a maven project usually test cases are launched with maven and classes are in the src/test/java path. In this project the test class is in the src/main/java path so it can be launched programmatically. In the main class it can be found the launching sentence: JUnitCore.runClasses(AskTestAssertions.class);

The test classes do two kind of things, they define the order of the method executions and they do some assertions inside their methods. The order of the executions is indicated with the following annotations: @Ignore, @BeforeClass, @Before, @Test, @After and @AfterClass. In this project they can be found the @Before and @Test. In order to perform the assertions the org.junit.Assert class provides the following methods: assertEquals, assertTrue, assertFalse, assertNotNull, assertNull, assertSame, assertNotSame and assertArrayEquals. In this project it can be found the assertEquals.

10. FindBugs

The project was verified with the last version of the Eclipse plugin FindBugs and it shows no messages. Anyway, there are some special cases in which a message is not indicating a source code weakness. For example, in the method AskModel.newCurrentInvoice(AskUser) FindBugs will indicate that the askUser parameter could be null if the second check “askUser == null” is removed. It is wrong, it is a FindBugs’ bug. As the validateNotNull call is going to throw an exception whether the askUser parameter is null, at the point of the second check this parameter cannot be null.

11. Spring Data JPA – Hibernate – DAO

The persistence technology chosen is Spring Data JPA. So the persistence layer is JPA compliant and the implementation is Hibernate.

Another point that deserves to be mentioned is the Spring DAO template generation. Spring provides a parametrized interface named CrudRepository<T, ID> which allows developers to define their DAO interfaces, for this project the DAOs can be found in the com.alten.ask.model.orm package. Through these interfaces Spring provides a full set of CRUD (Create, Read, Update and Delete) methods to manipulate entities of type T with ids of type ID. Besides, it become easy to write some new methods and some new JPQL queries.

12. H2 Database

The choice of the database is related to the possibility of using an SQL database without the need of an installation. It is an in-memory database and it can also be saved to the file system. The application will create and update a file named askDB.mv.db into the user folder or in the application's folder. The properties can be found in the file application-2.properties which is loaded by Spring in the main class AskMainApplication through the annotation @PropertySources(value = { @PropertySource("classpath:/application-2.properties") }).

13. Logger

The chosen logger is java.util.logging.Logger. Every class that uses the logger does not include the Logger attribute, but a wrapper one called AskLogger. This architectural/design choice enables the possibility of saving the messages to the database each time a message is logged and there is a user in the session. Doing that the messages are persisted in one point and the logger still does its normal job.

The java logger is configured to save the messages to a file called asklogging.txt which is going to be overwritten each time the application starts.

The logger will also save a transaction each time a user logs in and it will log an action each time the user does something.

14. Exception

In the main package it can be found the project exception class AskException. Then, every time a try-catch block catches an AskException, the application does not log the stack trace. Anyway, the exception is always being logged in the AskException class when the class is instantiated. This choice is based on the need of keeping track of all thrown project exceptions. Thus, whether a programmer does log the exception, it is going to be logged twice, but if the programmer does not do it, the exception is going to be logged anyway.

15. I18N – Internationalization

All the labels have been externalized in files named with the format strings[_xx].properties and these files are internationalized with the native java.util.Locale by doing Locale.setDefault in the method AskLogic.setDefaultLocale. If the user changes the language and the application is stopped and launched again, the system should remember the last choice.

16. CSS

The styles can be easily attached with Vaadin and Maven, they only need a three-step configuration. First, the file structure which can be seen in the src/main/webapp/VAADIN/themes folder. Then the theme needs to be set as in the method AskVaadinUI.init. And finally the styles can be used as in the method AskCompSections.doAction, aComponent.setStyleName.

17. Email – Java Mail

By default the application will not send any email because the password of the account is not set. Besides, by default it will not try to use a proxy because these properties are under comment (#). If the email account is configured, an email will be sent each time an invoice is purchased.

The application is prepared to detect and load any property present in the file email.properties. The minimum set of properties required to get the email working is composed by the first three properties in the file: to, mail.smtp and mail.smtp.passwd. Please refer to that file (email.properties) for detailed information about some specific properties.

CONCLUSION

Doing a project like these could be a very good experience in order to learn or become faster in terms of how technologies need to be integrated. Even more, some enterprises have a step-by-step **tutorial** to help new employees acquire the minimum required concepts. Nevertheless, in a real scenario there is barely such disponibility of time. Thus, usually big enterprises do agree with their technical architects one or more of three possible working contexts.

In the first one, which is easier, the architect prepares a **template** application with a set of already-configured technologies that can be used as a starting point in a specific project. Templates are very useful because they solve configuration problems in advance and because they enable conventions allowing the enterprise to develop an internal culture like “that’s the way we do things around here”. A simple example within this project could be the pom dependency for java mail plus the email.properties file.

In the second working context the architect develops a **framework** that also includes a set of already-configured technologies and which is usually a library or an engine embedded in the main application. This framework runs online (with the application) and does some complex operations that otherwise should be faced for each new technical project. A simple example within this project could be the AskEmailSender class.

Finally, in the third working context they can be found the **source generators**. These are meta softwares with a high degree of abstraction which are able to produce some parts of other softwares. They run offline (while the application is being constructed) and they create enormous quantities of no-error-well-designed-high-performant source code. An example within this project could be found in the entities in the com.alten.ask.model.orm package. They were created on a 98% with the Hibernate Tools of the JBoss plugin for Eclipse. They totalize 1530 lines of source code and this is a small project with only nine entities.