MIND STORM SOFTWARE PVT LTD

Google App Engine Training

ExamResults – Step 3 – Datastore Service

In this session, we are going to build the Data Layer for the application. In the previous session, we build the Web Interface for the ExamResults application but it was returning a dummy result. So what we plan to do here is the following:

1) Learn about Datastore Service in App Engine and build the Data Access Layer for Exam Result objects
2) Load the Datastore Service with sample data (4-5 Records)
3) Modify the Exam Results Servlet to query the results given a seat number and return the actual data. We will be using dummy data for the results, since we will be building out the database layer in the next session.

ExamResults – Step 3 – See it in Action

In terms of the UI, it will still be the same, except that the data is now being returned from the Data store layer that we are going to be build.

Refer to **hands-on-exercises/Exam Results App/step 3** for full project source code. You can also import it directly into Eclipse and study the code.

Import the Project

We will import the Eclipse project at this stage so that you have all the working source code for this step.

1. In Eclipse, go to **File → Import**. This will bring up the Import dialog.
2. Select **General → Existing Projects into Workspace** and click on **Next**.
3. For Select root directory, click on browse and go to **c:\appengine-training\hands-on-exercises\Exam Results App\step 3**
4. Ensure that **Copy projects into workspace** is selected.
5. Click on **Finish**.

Modify the ExamResult Entity class

Since are going to make the entity persistable, we need to make a few changes to the ExamResult entity class that we created earlier.

The changes involve adding JDO (Java Data Objects) Annotations to the class to mark the object as persistable and also to identify the key value in the Object. The key value will be the seat number, since that will be a unique way for us to identify the record in the database. Think of a key as having the same concept as a primary key in SQL Table.

The modified source code for the ExamResult.java file is shown below:

```
package com.mindstormsoftware.examresults.entity;

import java.io.Serializable;

import javax.jdo.annotations.IdGeneratorStrategy;
import javax.jdo.annotations.PersistenceCapable;
import javax.jdo.annotations.Persistent;
import javax.jdo.annotations.PrimaryKey;

import com.google.appengine.api.datastore.Key;

@PersistenceCapable
public class ExamResult implements Serializable {

    @PrimaryKey
    @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)
    Key seatNumber;
    String studentName;
    String marks_Math;
    String marks_CommSkills;
    String marks_Programming;
    String marks_ElectronicCircuits;
    String marks_Total;
    String marks_Percentage;//Constructors

//Getter & Setter methods

}
```

The key changes to note are:

1) We have annotated the class with @PersistenceCapable to mark the entity as being persistent. This will be handled internally by the JDO Implementation.
2) We are identifying the seatNumber attribute as the primary key with the @PrimaryKey annotation
3) Also notice that the data type of the seatNumber field has changed from String to Key. This is how App Engine will locate it in its Datastore.

ExamResult Data Access Object

We will now be defining a utility class that follows the Data Access Object pattern. This class will provide utility methods that will be used by other modules in the application to add / modify/ delete /search for Exam Result datastore entities. This will therefore shield them from knowing the Datastore API in App Engine.

The class that you should study here are:

src/com/mindstormsoftware/examresults/dao/ExamResultDAO.java

It contains methods for the following:

```
• List<ExamResult> getExamResults()
• deleteAll()
• add(String seatNumber, String studentName, String marks_Math, String
  marks_CommSkills, String marks_Programming, String
  marks_ElectronicCircuits, String marks_Total, String marks_Percentage)
• remove(String ExamResultKey)
• ExamResult getExamResult(String ExamResultKey)
```

Look at the methods in detail and you will find the Datastore API being used.

Modify the Servlet (ExamResultsServlet.java)

The only change that we need to make to the ExamResultsServlet.java is to change the call from called getDummyResult(seatNumber) to the actual call shown below:

```
ExamResult _result = ExamResultDAO.INSTANCE.getExamResult(seatNumber);
```

Load Initial Data

The first thing to do is to load the application with sample data. The sample data file is present in **WEB-INF\sampleData\examresults.dat**. To load the data, simply invoke http://localhost:8888/loaddata.jsp. On successful execution, this should load the ExamResult entity with 4 records.

The examresults.dat file is a simple text file where each row is a Exam Result for a particular seat. Each field is separated by a semi-colon (;). The file is shown below:

```
226510;Romin Irani;60;70;80;90;300;75
226511;Smarth Behl;70;70;70;70;300;70
226512;Prajyot Mainkar;80;80;80;80;320;80
226513;Jonathan D'Souza;90;90;90;90;360;90
```

The loaddata.jsp file calls a utility class **DataLoader.java** that does the following:

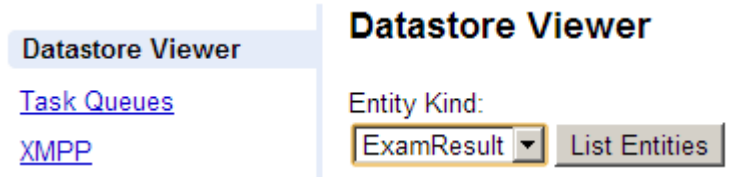
1. Read the data file line by line.

2. Parse out the line into individual attributes and invoke the **ExamResultDAO.*INSTANCE*.add()** method to insert the records one by one.

Test locally and Deploy

Once you made the changes, test out the changes locally first. Stop the Web Application if it is already running by clicking on the Stop icon in the Console window. Restart the application i.e. Right-click on Project and select **Run As → Web Application** and navigate to http://localhost:8888

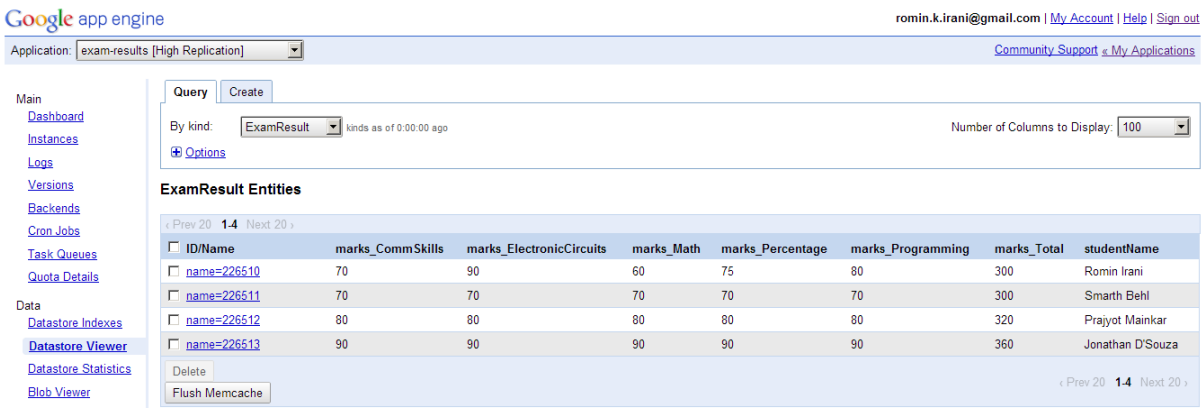
To view the data, you can go to the http://localhost:8888/_ah/admin and you should see the Entity **ExamResult** available as given below:



Click on the **List Entities** link to see the records.

You can also deploy your application to the cloud, if you wish. To deploy to the cloud, right-click on project and then **Google → Deploy to App Engine.**

Remember to run the http://<appid>.appspot.com/loaddata.jsp file on the server so that it is also populated with dummy data. To view the data, login to Administrative Console in your app and go to Data → Datastore Viewer. You should see records populated as given under:



Next Session

In the next session, we shall see how to add XMPP support to the application. Users will then be able to get exam results directly from the Google Talk application.