# ANKARA UNIVERSITY

**ENGINEERING FACULTY**

**DEPARTMENT OF COMPUTER ENGINEERING**

****

**INTERNSHIP REPORT**

**Java And Android Projects**

**Gizem HAZAR**

**10290220**

**05.08.2015**

**ABSTRACT**

This report describes the internship that is made under the supervison of Yeliz Gungor at Innova IT Solutions. This internship comprises two main projects; one of project is about Java application and the other project is an Android application. The Java application aim is creating three different platform string files from an Excel file. The Android application aim is solving a Thermo Dynamics problem with using Thermo Dynamics’ tables as the database. Also in this report is focused on the approaches made to solve the problem and explain the results.

**INSTITUTION INFORMATION**

Institutions;

Name : INNOVA IT SOLUTIONS

Deparment (if it can be : FTA (Financial Technical Analysis)

stated)

Address : Odtu Teknokent Telekom ARGE Binasi B Blok

Telephone : 0(312) 201 70 00

E-mail :

Web Page (if exists) : www.innova.com.tr

Innova is a leading software developer and integrator in Turkey which provides innovative software solutions and services, covering the entire project lifecycle from consulting, design, application development and integration to support.

Established in September 1999, Innova serves its bluechip client base from main offices in [İstanbul](http://www.innova.com.tr/istanbul.asp) and[Ankara](http://www.innova.com.tr/ankara.asp), a [manufacturing facility](http://www.innova.com.tr/kioskofis.asp) for kiosks in Istanbul, as well as support offices in 12 cities in Turkey.

Areas of business in Innova include OSS/BSS Systems for Telco operators, Financial Transaction Applications, ERP, CRM and BI systems, portals, custom software development IT systems management and infrastructure, IT security and kiosk systems.

With more than 1000 people serving its clients, Innova has delivered solutions and services to customers in 33 countries to date. Major customers of the company are fixed line and GSM operators in Turkey and abroad, banks, as well as other prominent organizations in the manufacturing, public and service industries.

TABLE OF CONTENTS

**ABSTRACT i**

**INSTITUTION INFORMATION ii**

**TABLE OF CONTENTS iii**

**1. INTRODUCTION 1**

2. JAVA APPLICATION 2

2.1. Introduction to Java 2

2.2. The Aim of the Application 2

2.3. Console Version of the Application 2

2.4. Gui Version of the Application 4

**3. ANDROID APPLICATION** **6**

3.1. Introduction to Android 6

3.2. Basic Terminology 6

3.3. The Aim of the Android Application 8

3.4. The Thermo Dynamic Application 8

4. CONCLUSION 12

BIBLIOGRAPHY 13

1. **INTRODUCTION**

My internship was done under the supervison of Yeliz Gungor. The objective was to design an Android application and Java application.

The purpose of Java application is to create three platform string files (android, Ios, windows phone) from an Excel file. Mainly these files has different extentiens (.properties, .string, .resx) and these files is created by application with their compressed files.

The purpose of Android application is to solve Thermo Dynamic problem easly. The application takes some values for find these values in tables and take the other values which they are related the value then show all results. For this application firsly database is used which inclued many thermo dynamics table. An application was designed, which checked for the user’s attention and proper functioning of the Android phone by checking for the mentioned events.

Web-based [Git](https://en.wikipedia.org/wiki/Git_(software)" \o "Git (software)) repository hosting service also was used in my work which it offers all of the [distributed revision control](https://en.wikipedia.org/wiki/Distributed_revision_control) and [source code management](https://en.wikipedia.org/wiki/Source_code_management) (SCM) functionality of Git as well as adding its own features. So all projects were commit in an Github account.

1. **JAVA APPLICATION**
   1. **Introduction to Java**

Java is a programming language initialiy developed by Sun Microsystems and released as a principal component of Sun Microsystems Java platform Although the language gets much of its syntax from C and C++ it has a less complicated object model ad lesser low-level services. Java applications are typically complicated to byte code (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general purpose , object-oriented language that is specifically designed to have as few implementation dependencies as possible. Java is currently one of the most standart programming languages in use, and is extensively used from application software to web applications.

* 1. **The Aim of the Application**

The purpose of Java application is create three platform string files from an Excel file (Excel file must be formed with key and value pair for different languages sheets. example Excel file is attached on CD.). It produces files which these are properties extension for Android, string extension for iOS and resx extension for Windows Phone platform. For create these files, the basic project consists of three classes and in this project some external jar file is used for read Excel file (Apache POI –the Java API).

* 1. **Console Version of the Application**

Console Version is the first version of the application. After the java application is run, runnable jar file can be created and when the file is dropped on CMD or Terminal it can run very easily. In the Java codes, first of all Excel file path and a destination path are taken from user in “main“ class. Then the file is read in “readFromExcel” class and kept all key and value pairs in a “TreeMap” in a method with following code (Figure 1.1) and this process is applied number of sheets times which depends on Excel file includes how many sheets in a workbook.

After all key-value pairs are kept in a global TreeMap, other methods are called for every format to produce files with their extension correctly.

TreeMap<String, String> properties = **new** TreeMap<String, String>();

. . .

FileInputStream file = **new** FileInputStream(inputFile);

XSSFWorkbook workbook = **new** XSSFWorkbook(file);

XSSFSheet sheet;

**for** (**int** i = 0; i < workbook.getNumberOfSheets(); i++) {

properties.clear();

sheet = workbook.getSheetAt(i);

Iterator<Row> rowIterator = sheet.iterator();

**while** (rowIterator.hasNext()) {

. . . .

**while** (cellIterator.hasNext()) {

. . . .

**switch** (cell.getCellType()) {

**case** Cell.***CELL\_TYPE\_NUMERIC***:

Double val = cell.getNumericCellValue();

String vl = String.*valueOf*(val);

properties.put(key, vl);

**break**;

**case** Cell.***CELL\_TYPE\_STRING***:

String value = cell.getStringCellValue();

properties.put(key, value);

**break**;

}}}

workbook.close();

file.close();

createFolder();

createLanguageFolder(sheet.getSheetName());

writeToPropertiesFile(sheet.getSheetName());

writeToStringFile(sheet.getSheetName());

writeToResxFile(sheet.getSheetName());

Figure 1.1. Saving key-value pair and create files with different extension

Every file has different format for their working platform so after all methods are worked all files are created with the correct format easily. Then the object is created which it belongs to “AppZip” class to organize all files for their platform in compressed files. After the “addFiletoZip” method called, files which they're in different formats (.properties,.string,.resx) will be located under relational platform folder with their zip files in the path that is taken from user (Figure 1.3).

* 1. **Gui Version of the Application**

Gui is the second version of the application, with this version Java Swing is used.Swing is a [GUI](https://en.wikipedia.org/wiki/Graphical_user_interface) [widget toolkit](https://en.wikipedia.org/wiki/Widget_toolkit) for [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). With gui version of the application again the project can be run on CMD or Terminal with runnable jar file nonetheless the project can be run a any compiler (like Eclipse IDE).

In this version ,after jar file is run, a file chooser is opened and an excel file and a target folder can be select instead of drop the path on terminal. Excel filter is added to file chooser for prevent from wrong file chosing. Some files extension (like .xlsx, .xls, .xlt, .xltx) are accepted from the file chooser.

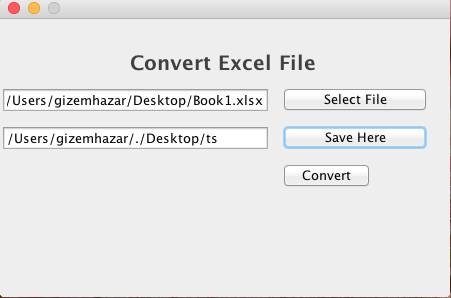


Figure 1.2 Gui Version of the Application

After Excel path and destination path are chosen, files are located under relational platform folder with their compressed files.

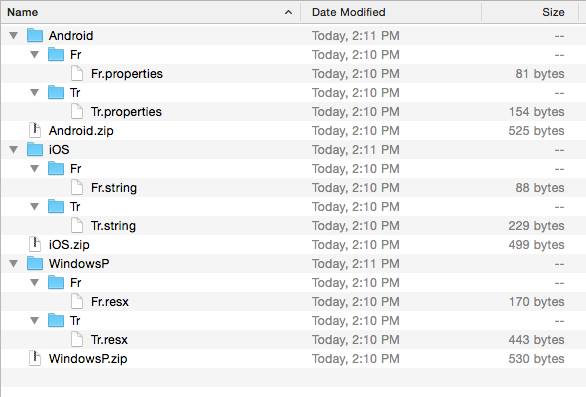


Figure 1.3 Relational Platform Folder and Zip Files

1. **ANDROID APPLICATION**
2. **Introduction to Android**

Android is an open-source software stack created for a wide array of devices with different form factors. Android applications are written in the Java programming language. The Android SDK tools compile the code along with any data and resource filesinto an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Once installed on a device, each Android application lives in its own security sandbox:

• The Android operating system is a multi-user Linux system in which each application is a different user.

• By default, the system assigns each application a unique Linux user ID (the ID is used only by the system and is unknown to the application). The system sets permissions for all the files in an application so that only the user ID assigned to that application can access them.

• Each process has its own virtual machine (VM), so an application’s code runs in isolation from other applications.

• By default, every application runs in its own Linux process. Android starts the process when any of the application’s components need to be executed, then shuts down the process when it’s no longer needed or when the system must recover memory for other applications.

1. **Basic Terminology**

Activity - An activity represents a single screen with a user interface. Each activity is given a window in which to draw its user interface. The window typically fills the screen, but may be smaller than the screen and float on top of other windows. An activity is implemented as a subclass of Activity.

Service - A service is a component that runs in the background to perform longrunning operations or to perform work for remote processes. A service does not provide a user interface. Another application component can start a service and it will continue to run in the background even if the user switches to another application. Additionally, a component can bind to a service to interact with it and even perform interprocess communication (IPC).

The Manifest file - The Android Manifest file identifies any user permissions the application requires, such as Internet access or read-access to the user’s contacts, declares the minimum API Level required by the application, based on which APIs the application uses, declares hardware and software features used or required by the application, such as a camera, bluetooth services, or a multitouch screen and 8 API libraries the application needs to be linked against (other than the Android framework APIs), such as the Google Maps library.

MVC Architecture – Model View Controller is an architecture that separates the representation of information from the user’s interaction with it. The model consists of application data and business rules, and the controller mediates input, converting it to commands for the model or view. A view can be any output representation of data, such as a chart or a diagram. Multiple views of the sama data are possible, such as a pie chart for management and a tabular view for accountants. The central idea behind MVC is code reusability and seperation of concerns.

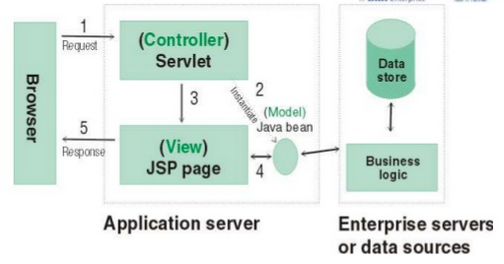


Figure 1.4 MVC Architechture

1. **The Aim of the Android Application**

The purpose of the Android application is solving Thermo Dynamic problem with using Thermo Dynamic tables so SQLite Databese is used in this project to organize all tables. For solving the problem some values are received from user. Values can be changeable, there is no certain rules that only two or three values must enter. Firslty fluid type is chosen from the user then user can enter Temperature value or Pressure value or two values could be selected from the user also. Therefore the architecture that provides all probabilities was developed.

1. **The Thermo Dynamic Application**

First of all, all tables were editted in Excel and added to database browser for sqlite to create a database. “SqliteBrowser” was used in this application to create a database. Then this database was added to asset folder of the application so when the application runs in any device firstly the database will be copied from asset folder and located at data folder of the application.

Then a splash screen was added to the application, splash screen is an activity that will show for set time when the application is starting and after set time period redirect to application main screen and prevents the user from using the application, then an animation which it provides moving the image in the middle of the screen from bottom to up, was added. In addition, a media player was added to splash screen with using MediaPlayer object. It provides to playback control the audio file. Then when the splash screen time is up, Activity screen is called.

In MainActivity class has activity screen (activity\_main.xml) and values are taken from the user here.

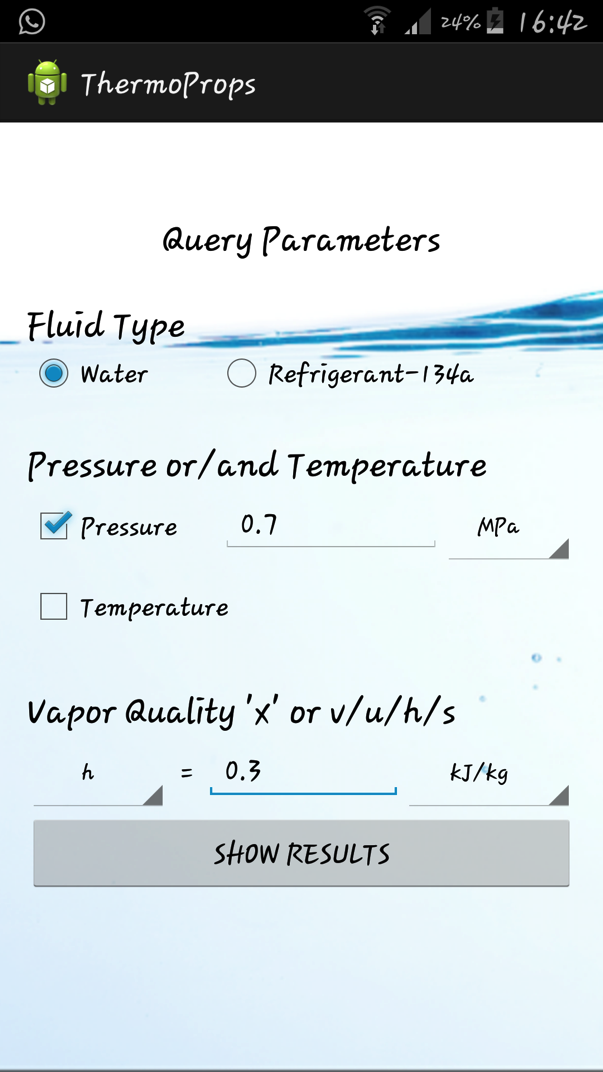
 

Figure 1.5 Activity Screen witt selection states

In this screen user have more than one choice to select values. User can be select Pressure and Temperature values together or can chose one of them and then have to select “x, v, u, h and s“ values.

For instance if user chose only pressure or temperature value than must select “x, v, u, h and s “ values (Figure 1.5) or if user choose pressure and temperature value there is no need to chose any extra value.(Figure 1.5). Then when the show results button is clicked, all values that are entered by the user are taken and first of all controll which values are selected with “fetchQueryInputs” methods. MVC architecture was used in this project to control and check these values.

In MVC architecture the object of “Mdl\_QueryInputs” class is created in “mainActivity” class and is send to “ResultActivity” class when the show button is clicked. This object keeps all informations which are taken from the database. If the values that are entered from the user have already in database table (firstly checks table a4), all column values are taken from database to Mdl\_QueryInputs”s object with helping “QueryHelper” class. If the values that are not exist in database, the application firstly checks whether Temperature or Pressure values are in between any two numbers. If it is then an interpolation coefficient is calculated in “QueryHelper” class and all other values (for every column) are calculated with this coefficient then these informations keeps “mQueryInputs” (Mdl\_QueryInputs”s object). (For Instance assume Temperature value is 17.5 °C, this value is between 17 and 18 °C (Figure 1.6), with these information the coefficient is calculated with interpolation rules.)

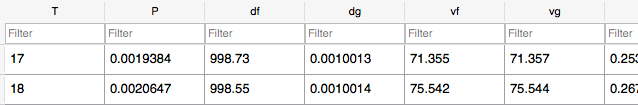


Figure 1.6 Table a4 in Database (only the values of 17-18 °C is shown)

If the values are not in table a4 then it means that fluid type is changed so the fluid type is calculated with thermo dynamics rules and decided type compressed or superheated instead of saturated. If fluid type is saturated table a5 and all tables which are related table a6 are checked. If fluid type is compressed table a7 is checked and again interpolation coefficient method is used.

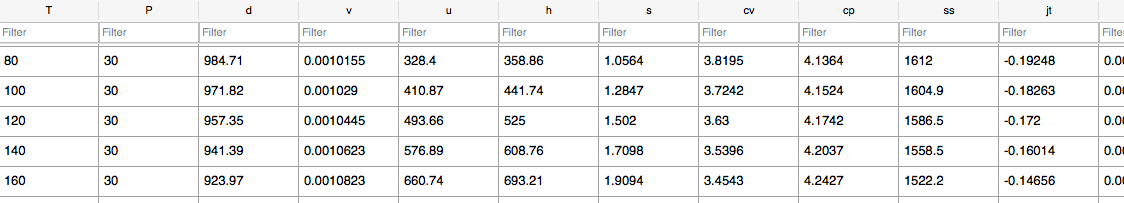


Figure 1.7 Table a7\_30

After all these database control steps no matther which table is used (every table has different size of column Figure 1.7) all information which are taken from database with help of cursor structure, are kept in mQueryInput object and finally all values are shown in an expandable list view in “activity\_results.xml” (Figure 1.8)

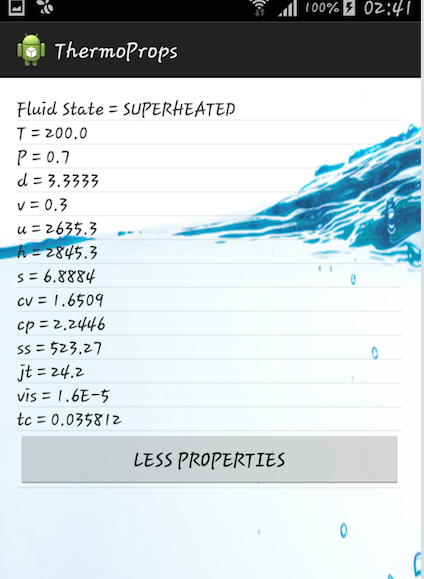
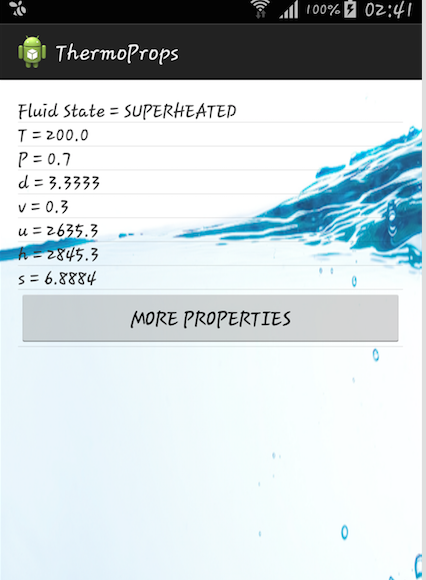


Figure 1.8 Activity\_result screen (All values are shown in a list).

**4. CONCLUSION**

End of the internship, Java and Android applications have been completed successfully and results were according to expectations.

In Java application Excel file was read and different platform files were created.Then these files (.properties,.string,.resx) was located under relational platform folder with their compressed files. In Android application all states were tested and matched all results values with database values. All projects were added Web-based repository hosting service (GitHub).

Overall, the results of my internship was very satisfied. Coding skills were improved by dint of this internship. Using a database with a Android project is learned. Sqlite query skills were improved.

**BIBLIOGRAPHY**

DB BROWSER, 2003 Db Browser for SQLite <http://sqlitebrowser.org/> Access Date 12.08.2015

MICROSOFT, 2015. Web Site, Microsoft Virtual Academy. <http://www.microsoftvirtualacademy.com/training-courses/implementing-entity-framework-with-mvc> Access Date 12.08 2015

WIKIPEDIA, 2001 Web Site, Android (operating system) <https://en.wikipedia.org/wiki/Android_(operating_system>) Access Date 11.08.2015

WIKIPEDIA, 2001 Web Site Java (programming language) <https://en.wikipedia.org/wiki/Java_(programming_language>) Access Date 10.08.2015

WIKIPEDIA, 2001 Web Sites Model-Viev-Controller <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

Access Date 11.08.2015