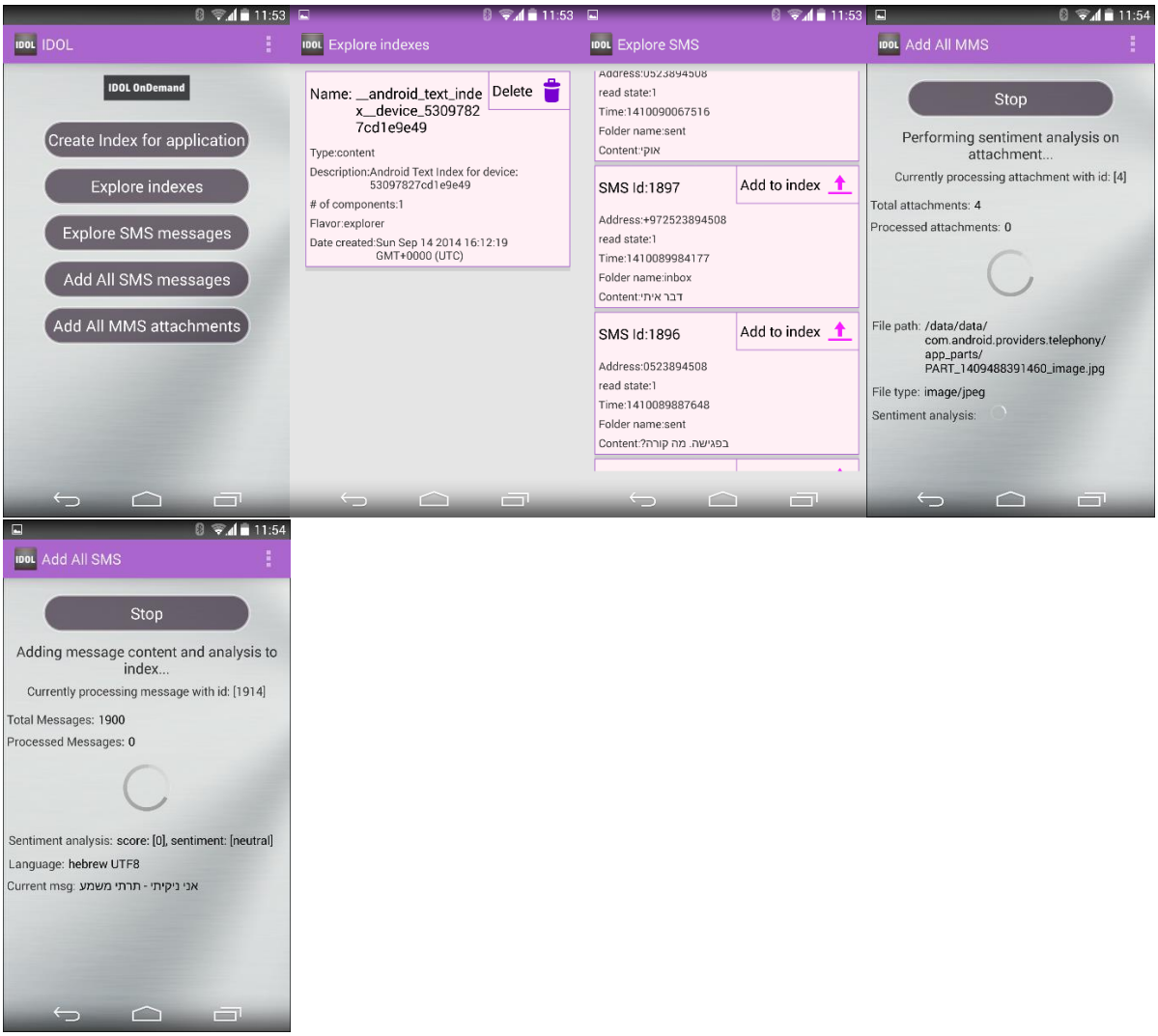
Application overview



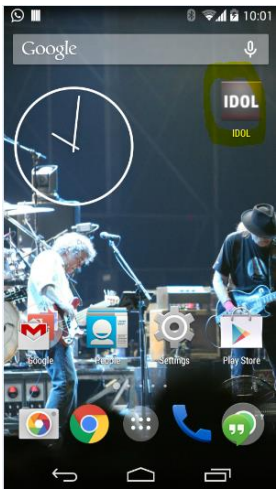
How to deploy the application and code?

An explanation video can be found here:
https://www.youtube.com/watch?v=qO-RJmCoRjU

For testing purposes:

To install the app please use an android device running with minimum version of KitKat (API 19).

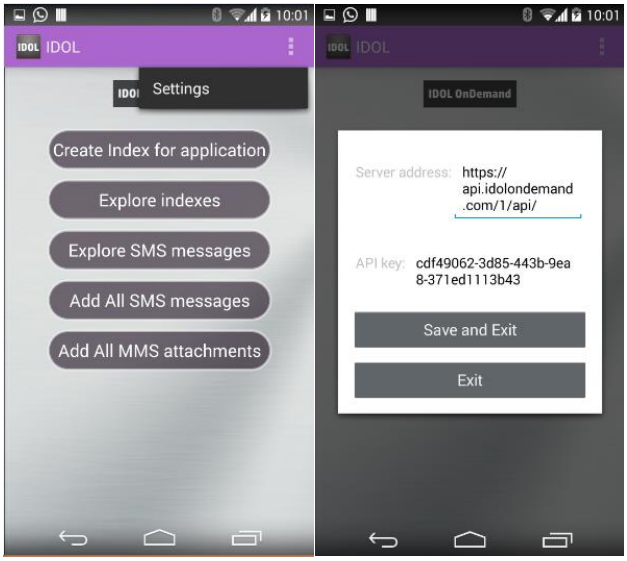
Open the APK file found in the root project folder (“TopCoder-IDOL\IDOL-release.apk”). The phone will ask for permission to install the app.



When you open the app the default server and API key will be as follow:

https://api.idolondemand.com/1/api/, "cdf49062-3d85-443b-9ea8-371ed1113b43".

You will be able to change it via settings:



Note: Don't forget to create an index ☺

For development/debug purposes:

Requirements:

IDE: Android Studio Beta 0.8.6

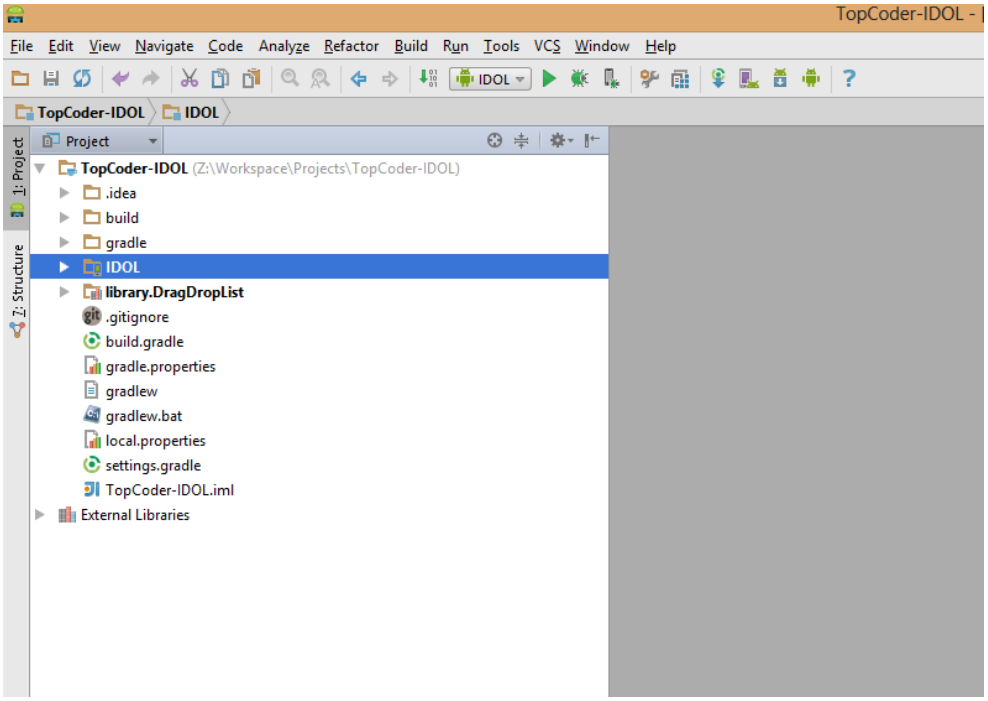
SDK: Android SDK v20

JAVA JDK: JDK1.7

Extracting the project files:

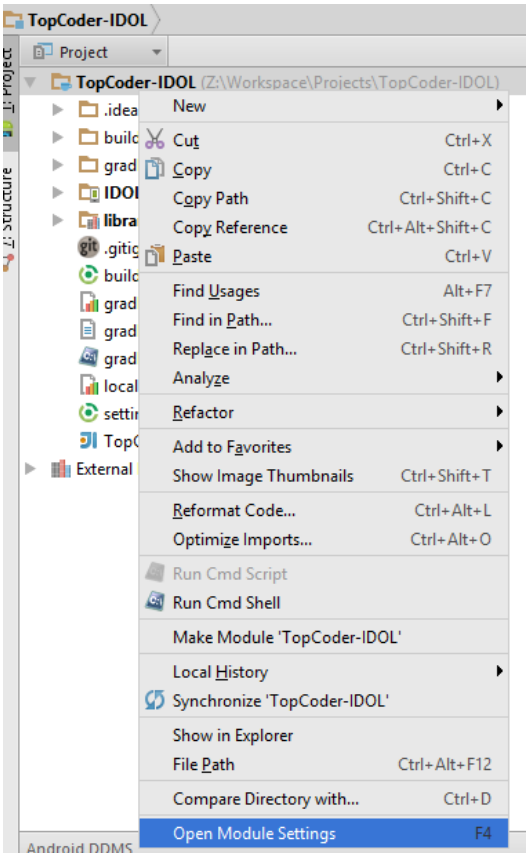
The attached ZIP project ("**TopCoder-IDOL.7z**") contains the module "IDOL" – the application code.

Open the project directory as a project in Android Studio:

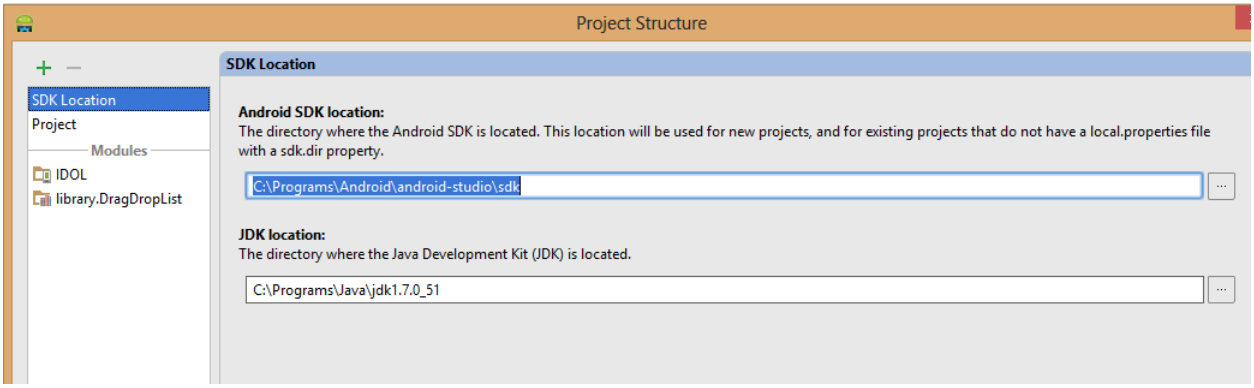


Make sure the project settings contains the right android-sdk and JDK folder.

Right-click the project and choose “Open Module Settings”:

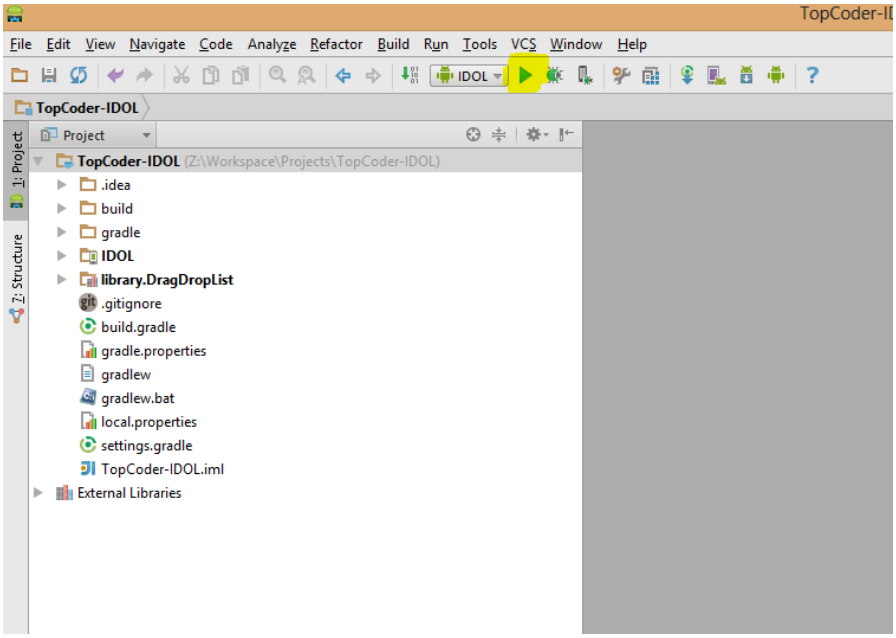


Then edit the sdk and jdk folders:



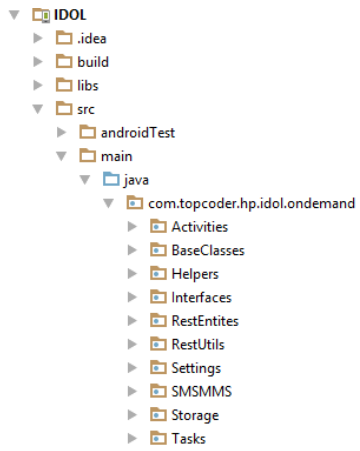
Try to run the application. There should be no errors.

Note: there should be a usb-debugging-on android device with minimum API 19 (KitKat) connected to the machine OR an emulator running.

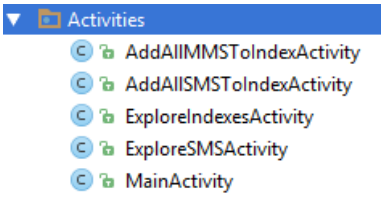

TopCoder-I
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
IDOL
TopCoder-IDOL
Project
1: Project
7: Structure
TopCoder-IDOL (Z:\Workspace\Projects\TopCoder-IDOL)
.idea
build
gradle
IDOL
library.DragDropList
.gitignore
build.gradle
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle
TopCoder-IDOL.iml
External Libraries

A bit about the code…

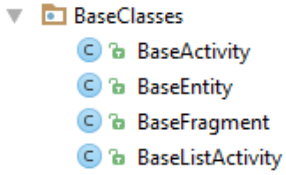
Java code folders structure:



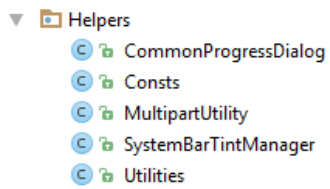
1. **Activities**
   Android activities of the application. Such as main activity, explore indexes activity, etc.
   Activities use Tasks to retrieve and share data with the server.
   Activities use layout xml objects (found in the 'res' folder) to display components on the screen.
   They also handle user interaction action (such as clicks, swipes and more).



2. **Base Classes**
   This folder contains the base classes of the application. For example there is a base class for activities and it adjusts the style of each activity that extends it.



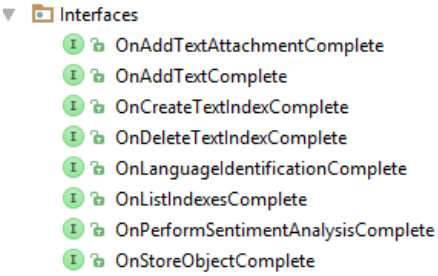
3. **Helpers**
   Helpers classes. Members and methods (Usually static) that are being used widely by other classes of the application.
   For example 'writeToLog(String msg)' function in the 'Utilities' class.



The class Utilities also exposes the method ‘Handle Exception’ which handles unexpected exceptions of the application.

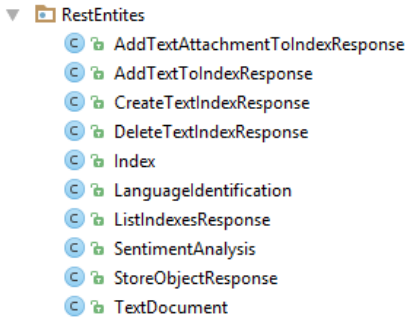
4. **Interfaces**

Interfaces used by classes in the application.
The main use of interfaces are to pass anonymous functions between async calls so that the call will have a callback function when it ‘wants’ to communicate with its caller.



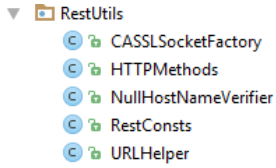
5. **RestEntites**

Rest entities that are passed between the client (application) and the server.
Includes request/response entities that are used by different layers of the application.



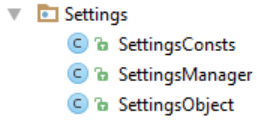
6. **RestUtils**

HTTP REST methods related classes to work with the IDOL Rest API.
For example: this folder contains the url addresses and it exposes a class that builds requested url’s according to input parameters – ‘URLHelper.java’ .



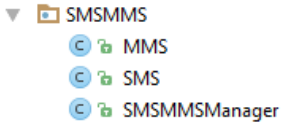
7. **Settings**

This class manages the setting of the application.
It manages the current configurable settings (Server address, API key).
It automatically saves settings offline and has default values.



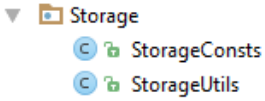
8. **SMSMMS**

   This folder contains two entities (MMS, SMS) and a manager.
   The manager exposes function that are used to read sms/mms content.
   It is used by the SMS/MMS related activities and tasks.



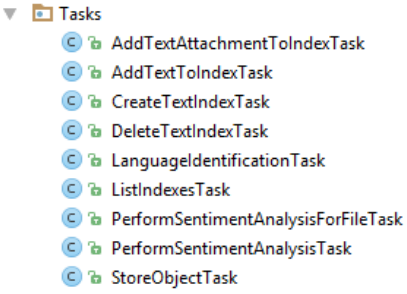
9. **Storage**

   This class exposes methods to read/write files to Android internal storage.
   Mainly used by the Settings layer.

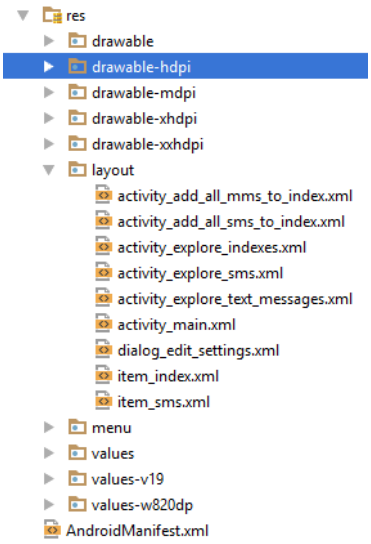


10. **Tasks**

    The tasks folder contains all the server – requests tasks that the application uses.
    For example – Add a text index, add a file to a text index, etc.
    It uses async calls to the server and uses the Rest layers to construct the requests and the responses.



Resource Folder:

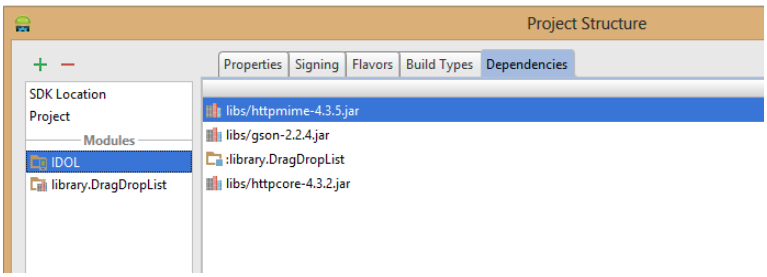


The resource folder contains all the layout xml objects, strings resources, images and more UI related files.

For example: The layout folder contains the layout xml's for the different activities of the application. The values folder contains the file 'strings.xml' which contains most the strings used by the UI layer and layout xml files.

External libraries:

The project uses the following libraries and JARs:



1. **library.DragDropList**
   a library used for constructing lists in the explore indexes/sms activities.
2. **Httpmime, httpcore** – Apache classes used for making java REST calls.
3. **Gson** – JSON serializer.