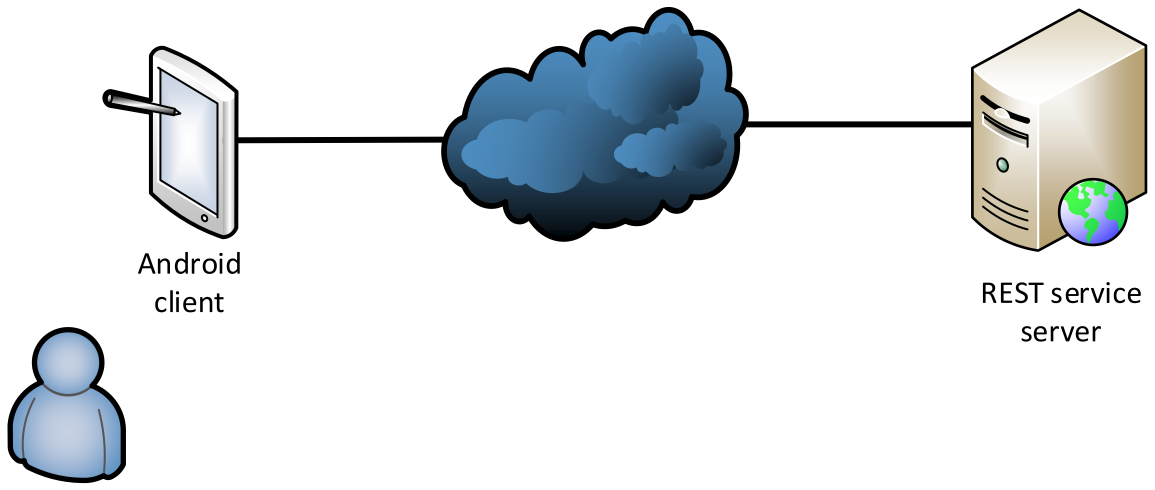
Android REST Demo

This demo contains the implementation of an Android application that calls an external REST service. The service implements several methods using the main HTTP verbs (GET, POST, DELETE and PUT).



The implementation uses a .NET WCF Service configured as a REST endpoint, hosted on a simple Windows console application.

.NET REST Service and Host

The .NET service is built on a Visual Studio s olution (RestDemo) with three projects (RestService, RestHost and Client). All use the .NET 4.6 framework full profile.

The first project contains the service definition (IRestService.cs), as an annotated WCF service, and implementation (RestService.cs). There are five operations: GetUsers (HTTP verb GET) that retrieves all the data in an internal data structure encoded in Json format; a GetUser operation (HTTP verb GET) that receives a single data item in Json format, by the specification of its Id passed in the URL; a AddUser operation (HTTP verb POST) that adds a new item to the resource represented by the service, where the new data is passed by the payload of the POST request in Json format; a DeleteUser operation (HTTP verb DELETE) which deletes an already existent item with its Id passed in the URL; and finally a ChangeUser operation (HTTP verb PUT), which changes the name of an item, passing its Id in the URL and the new name as a payload of the PUT request.

The second project is a simple standard WCF service host to the previous service. The configuration file (App.config in the project, but automatically copied to the bin folder as RestHost.exe.config) is the important aspect. It should be configured with the *webHttpBinding* binding and with the endpoint behavior *webHttp*. The listening address is http://<machine>:8701/Rest/ followed by the template URL of the REST operations.

There is also a console .NET client, requesting several operations, just to see that the service is running correctly.

The RestHost should run in an administrative console to allow the use of the http port in listening mode.

Android client

The Android client invokes (using a menu) the several operations of the service using one of the Android HTTP connectors (the class HttpURLConnection), specifying the address (using the REST template defined by the service), http verb (GET, POST, DELETE or PUT), and the input and output stream operations needed and associated with the HTTP connector. The response and sent data are in Json format. When the response comes it is read as a string and presented on the screen. It could be decoded with the Android classes JSONArray and JSONObject.

The communications run on separated threads different from the user (UI) thread. This is necessary now because Android prevent the call of external communications in the UI thread.

You should notice that the modifications to the controls present in the screen (the layout controls) should always be made in the UI thread and not directly from the comms thread.

In the manifest notice the tag <uses-permission> for the use of external internet connections.