Android SDK

Version notification

The Android Apstrata SDK has been upgraded to a new major version (v 2.0) on the 10th of February 2014. Versions prior to that date have been deprecated. We recommend moving to the newest version.

The **Android Apstrata SDK** is a utility that is provided to developers of Android-based applications. The Android Apstrata SDK wraps the Apstrata REST APIs making it very easy to call these latter from within Android-Java code.

All you need to do is to:

- Drop the client library into you code,
- Select the type of Connection you need to use. A "Connection" is actually a way of signing the calls from your app. to the Apstrata APIs. There are four types of connections:
    - "OwnerConnection", if you need to sign your calls using your Apstrata account owner's credentials,
    - "UserConnection" if you need to sign you calls using the credentials of one of the users defined in your Apstrata account
    - "TokenConnection", if you prefer to use an Apstrata authentication token instead of signing the calls using some of the aforementioned credentials
    - "AnonymousConnection", if requests do not need to be signed.
- Create an instance of the "Client" class passing it the URL of the Apstrata server, the authentication key of the Apstrata application account you are using and the Connection instance you will be using
- Call one of the methods exposed by the Android client, specifying the authentication mode needed (more on authentication with Apstrata)..

Click here to download the Android Apstrata SDK source code. The source code ships with a test application. Note that you need to install the Android SDK.

Click here to download the Android Apstrata SDK jar file and required complementary libraries.

Example

```
String authKey = "A71326F793"; // replace with your authentication key
String baseURL = "https://sandbox.apstrata.com; // replace with the URL to your Apstrata application account

// Example 1, using an OwnerConnection
// Create an instance of OwnerConnection using your Apstrata authentication Key and secret
String secret = "F2A0240C8F61652GD45173BF0EC223X1"; // replace with your secret
Connection ownerConnection = new OwnerConnection(baseUrl, authKey, secret);

// Create an instance of the Apstrata Android Client with the above Connection instance
Client client = new Client(baseURL, authKey, ownerConnection);

// Determine the signature mode to use (Complex or Simple)
Client.AuthMode mode = Client.AuthMode.COMPLEX;

// Prepare the parameters to send according to the requested Apstrata API (In this example, "SaveDocument")
List<NameValuePair> parameters = new ArrayList<NameValuePair>();
parameters.add(new BasicNameValuePair("apsdb.store", "DefaultStore"));
parameters.add(new BasicNameValuePair("apsdb.update", "false"));
parameters.add(new BasicNameValuePair("some_field_to_save", "blah"));

// Invoke the API using the Client instance and signature mode
// note that since we're not passing files, we send "null" in place of the "files" parameter
String response = client.callAPIJson("SaveDocument", parameters, null, mode);

// Example 2, using a TokenConnection
// TokenConnection require the addition of two steps to the scenario above. Since a token needs to be
regularly renewed
// and since it has a limited life-time (cannot renew a token for ever without re-sending credentials),
// we need to explicitly instruct the TokenConnection instance to start refreshing the token, respectively
// stop doing it.

// Let's start by creating an instance of TokenConnection
String username = "someUser@somemail.com"; // You normally would get this from a Login form
String password = "somePassword"; // You normally would get this from a Login form
long tokenExpiry = 60; // The time in seconds before before discarding the token
long tokenLifetime = 120; // The time in seconds before having to resend user credentials to renew the
token
TokenConnection tokenConnection = null;

try {

        tokenConnection = new TokenConnection(baseURL , authKey , username, password, tokenExpiry,
tokenLifetime);

        // Ask the TokenConnection to validate its current token, if not token existed, one is generated
and true is returned
        // (false will be returned in case of passing invalid user credentials)
        boolean isValid = tokenConnection.validateToken();
        if (isValid) {

                // Create an instance of the Apstrata Android Client with the above Connection instance
                client = new Client(baseURL, authKey, tokenConnection);

                // invoke some APIs ...
        }
}catch(Exception e) {
        // Do some exception handling ...
}finally {
        // Terminate the connection if not needed anymore
        if (tokenConnection != null) {
                tokenConnection.terminate();
        }
}
```

Tip

If you need to create a URL to an image file saved in an Apstrata document, use Client.getSignedRequestUrl()

Apstrata Android SDK javadoc