# Summary

1. For pure java user, you need only the BoxJavaLibraryV2 package.
2. For android user, you need BoxAndroidLibraryV2 package, which is an android library, and the BoxJavaLibraryV2 package, which is a pure java package and is referred by the BoxAndroidLibraryV2.

# Known Issues

1. Not all the features are supported, for example trash and search.

# API Calls Usage Example

**// authenticate the client, only OAuth is supported. More details can be found**

**// in next part**

**boxClient.authenticate(oAuthView, autoRefreshToken, listener);**

**// Get file info.**

**BoxAndroidFile boxFile = boxClient.getFilesManager().getFile(fileId, null);**

**// Get file info, require sha1, description to be in the resulted object**

**BoxDefaultRequestObject requestObj =**

**(new BoxDefaultRequestObject()).addField(BoxFile.FIELD\_SHA1)**

**.addField(BoxFile.FIELD\_DESCRIPTION);**

**BoxAndroidFile boxFile = boxClient.getFilesManager().getFile(fileId, requestObj);**

**// Get the 20th to 50th children items(total 30 items) of a folder, requiring**

**// etag, description, name to be in the resulted object.**

**BoxFolderRequestObject requestObj =**

**BoxFolderRequestObject.getFolderItemsRequestObject(30, 20)**

**.addField(BoxFolder.FIELD\_NAME)**

**.addField(BoxFolder.FIELD\_DESCRIPTION)**

**.addField(BoxFolder.FIELD\_ETAG);**

**BoxAndroidCollection collection =**

**boxClient.getFoldersManager().getFolderItems(folderId, requestObj);**

// Upload a new file.

**BoxFileUploadRequestObject requestObj =**

**BoxFileUploadRequestObject.uploadFileRequestObject(parent, “name”, file);**

**List< BoxAndroidFile > bFiles =**

**boxClient.getFilesManager().uploadFiles(requestObj);**

**// Upload a new file with a upload progress listener.**

**BoxFileUploadRequestObject requestObj =**

**BoxFileUploadRequestObject.uploadFileRequestObject(parent, “name”, file)**

**.setListener(listener));**

**List<BoxAndroidFile> bFiles =**

**boxClient.getFilesManager().uploadFiles(requestObj);**

**// Download a file.**

**boxClient.getFilesManager().downloadFile(fileId, null);**

**// Delete a file, only delete if etag matches.**

**BoxFileRequestObject requestObj =**

**BoxFileRequestObject.deleteFileRequestObject()**

**.setIfMatch(etag);**

**boxClient.deleteFile(fileId, requestObj);**

**.**

# Simple Authentication Example (OAuth)

**// This is the code in your activity to get OAuth, to get more details, check**

**// the sample app.**

**// An OAuthActivity is provided in the sdk, include it in your manifest.**

**Intent intent = OAuthActivity.createOAuthActivityIntent(this, clientId,**

**clientSecret);**

**startActivityForResult(intent);**

**// Then override the onActivityResult();**

**@Override**

**public void onActivityResult(int requestCode, int resultCode, Intent data) {**

**if (resultCode == Activity.RESULT\_CANCELED) {**

**// handle oauth failure case.**

**String failMessage = data.getStringExtra(OAuthActivity.ERROR\_MESSAGE);**

**// Here implement your own logic to handle this fail message.**

**handleFail(failMessage);**

**} else {**

**BoxAndroidClient client =**

**data.getParcelableExtra(OAuthActivity.BOX\_CLIENT);**

**// Now you can use the client, pass it around, make api calls.**

**youOwnMethod(client);**

**}**

**}**

# Advanced Authentication Example (OAuth)

**// This is the code in your activity to get OAuth, to get more details, check**

**// the OAuth sample app.**

**// An customized view is provided in the sdk, include it in your layout and**

**// use findViewById to get it.**

**oauthView = (OAuthWebView) findViewById(R.id.oauthview);**

**oauthView.initializeAuthFlow(boxClient, this);**

**boxClient.authenticate(oauthView, autoRefreshOAuth, getOAuthFlowListener());**

**// Create a listener listening to OAuth flow. The most important part you need**

**// to implement is onAuthFlowEvent and catch the OAUTH\_CREATED event, this**

**// indicates OAUth flow is done, the BoxClient is authenticated and you can**

**// start making api calls.**

**Private OAuthWebViewListener getOAuthFlowListener() {**

**Return new OAuthWebViewListener() {**

**@Override**

**public onAuthFlowEvent(final IAuthEvent event, final IAuthFlowMessage message) {**

**// authentication is done, you can start using BoxClient now, for**

**// example, making an api call to download a file.**

**}**

**}**

**}**

**// Get an OAuth object (You can get this and save it and use this OAuth object**

**// to authenticate the client at a later time or in a different activity)**

**BoxOAuthToken oauthObject = boxClient.getAuthData();**

**// Re-authenticate using the previously obtained OAuth object.**

**boxClient.authenticate(oauthObject);**

# Tips / pain points

1. There are two sets of DAO objects, for example BoxFile and BoxAndroidFile, they belong to java sdk and android sdk, however when you are using android sdk and get a collection, although the real object is of type BoxAndroidFile, the item in the collection is defined by it’s super type so you will need to cast.
2. getFolder method in FoldersManager returns a folder object, which has a item collection field, this is supposed to be mini-formats of all the children inside this folder, however it could be incomplete, if you want to get a complete list of children, use getFolderItems method instead.