JSONStore - Java API

Follow along with the code sample

1. Download the compressed file with the code sample that is associated with this tutorial.
2. Open the Basic.java "JSONStoreAPINativeAndroid/src/com/worklight/jsonstorenativeandroid/tests/Basic.java" file. The sample image provides context.
3. Run the application by using Android JUnit. Right-click the application, select **Run As** and then click **Android JUnit Test**.

> **Note:** The code sample uses the built-in JUnit Test framework of Android. Explaining how it works is beyond the scope of this tutorial.

Code Sample Walkthough

```
        try {
            Context ctx = getContext();
            WLJSONStore.getInstance(ctx).destroy();
            List<JSONStoreCollection> collections =
                    new LinkedList<JSONStoreCollection>();
            JSONStoreCollection people = new JSONStoreCollection("people"
);
            people.setSearchField("name", SearchFieldType.STRING);
            people.setSearchField("age", SearchFieldType.INTEGER);
            collections.add(people);
            WLJSONStore.getInstance(ctx).openCollections(collections);
            JSONObject data1 = new JSONObject("{age: 20, name: 'carlos'}");
            JSONObject data2 = new JSONObject("{age: 30, name: 'mike'}");
            people.addData(data1);
            people.addData(data2);
            List<JSONObject> results = people.findAllDocuments();
            assertEquals(
                new JSONObject("{_id: 1, "
                        + "json: {'name' : 'carlos', 'age' : 20}}")
                    .toString(),
                results.get(0)
                    .toString());
            assertEquals(
                new JSONObject("{_id: 2, "
                        + "json: {name: 'mike', age: 30}}")
                    .toString(),
                results.get(1)
                    .toString());
        }
        catch (JSONStoreException ex) {
            throw ex;
        }
        catch (Throwable t) {
            throw t;
        }
```

The `destroy` API removes all JSONStore content from the application. It is used here to start with no data. Doing it this way ensures that the output is predictable in the code sample.

> **Note:** Explaining Context (`ctx`) is beyond the scope of this module. For more information about that object, see the Android API documentation.

To persist data, you must first define at least one collection. These collections are entities that hold data. You can see here the definition of a collection that is called `people`.

Search fields are fields that are indexed inside a collection. You can use those fields when you search for data that is inside a collection.
You can see here the definition of two search fields:

- `name` (string)
- `age` (integer)

The data types, such as `string`, `integer`, `number`, `boolean`, are used to better store input data.

The `open` API is used to open one or more collections. If the collection was never opened before, a file is created on the file system to persist data inside the collection. Before the operation finishes, an accessor to that file is created.
The accessor allows the caller to call collection-level APIs such as `add` and `findAl`, which are shown later in this code sample walkthrough.

The data that is stored inside the `people` collection is defined here. Notice that the data is a hardcoded array of two JSON objects with key value pairs for `name` and `age`. This data can be acquired from multiple sources (for example: Network Request, File I/O, User Input).

The collection accessor provides access to store data inside the `people` collection. The input data must be in JSON format.

There are a couple of different ways to find documents inside a JSONStore collection (for example: `find`, `findById`). The easiest way, and the way that is shown here, is by using the `findAll` API. This method returns all the data that is stored inside a collection.

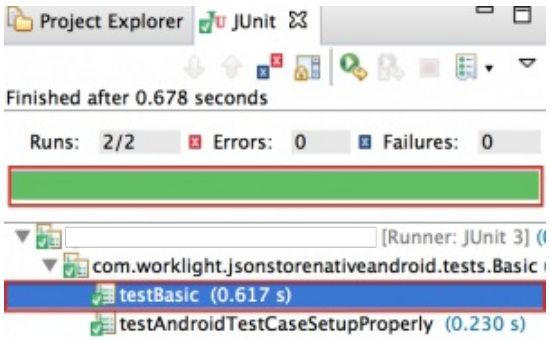
Data that is stored inside a collection is called a document. Documents have `_id` and `json` key value pairs. The `_id` pair is an internal identifier that is added automatically when data is added. The json pair contains all the data that was added.

If an error occurs, a `JSONStoreException` object is thrown. The exception contains information about the error.

Sample application

Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/JSONStoreAPIBasicsProject.zip) Studio project.
Click to download (http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/JSONStoreAPINativeAndroidNativeProject.zip) Native project.

Expected Output



To execute the tests, right-click **Project** > **Run As** > **Android JUnit Test.**
When the tests are executed, the output looks similar to the sample image.
The green bar above the tests indicates that everything is working as expected.

For more information

For more information about JSONStore, see the product user documentation.