Troubleshooting JSONtore

Overview

Find information to help resolve issues that you might encounter when you use the JSONStore API.

Provide information when you ask for help

It is better to provide more information than to risk not providing enough information. The following list is a good starting point for the information that is required to help with JSONStore issues.

- Operating system and version. For example, Windows XP SP3 Virtual Machine or Mac OSX 10.8.3.
- Eclipse version. For example, Eclipse Indigo 3.7 Java EE.
- JDK version. For example, Java SE Runtime Environment (build 1.7).
- IBM MobileFirst Foundation version. For example, IBM® Worklight® V5.0.6 Developer Edition.
- iOS version. For example, iOS Simulator 6.1 or iPhone 4S iOS 6.0 (deprecated, see Deprecated features and API elements).
- Android version. For example, Android Emulator 4.1.1 or Samsung Galaxy Android 4.0 API Level 14.
- Windows version. For example, Windows 8, Windows 8.1, or Windows Phone 8.1.
- adb version. For example, Android Debug Bridge version 1.0.31.
- Logs, such as Xcode output on iOS or logcat output on Android.

Try to isolate the issue

Follow these steps to isolate the issue to more accurately report a problem.

1. Reset the emulator (Android) or simulator (iOS) and call the destroy API to start with a clean system.
2. Ensure that you are running on a supported production environment.
    - Android >= 2.3 ARM v7/ARM v8/x86 emulator or device
    - iOS >= 6.0 simulator or device (deprecated)
    - Windows 8.1/10 ARM/x86/x64 simulator or device
3. Try to turn off encryption by not passing a password to the init or open APIs.
4. Look at the SQLite database file that is generated by JSONStore. Encryption must be turned off.
    - Android emulator:

```
$ adb shell
$ cd /data/data/com.<app-name>/databases/wljsonstore
$ sqlite3 jsonstore.sqlite
```

    - iOS Simulator:

```
$ cd ~/Library/Application Support/iPhone Simulator/7.1/Applications/<id>/Documents/wljsonstore
$ sqlite3 jsonstore.sqlite
```

    - Windows 8.1 Universal / Windows 10 UWP simulator

```
$ cd C:\Users\<username>\AppData\Local\Packages\<id>\LocalState\wljsonstore
$ sqlite3 jsonstore.sqlite
```

    - **Note:** JavaScript only implementation that runs on a web browser (Firefox, Chrome, Safari, Internet Explorer) does not use an SQLite database. The file is stores in HTML5 LocalStorage.
    - Look at the `searchFields` with `.schema` and select data with `SELECT * FROM <collection-name>;`. To exit sqlite3, type `.exit`. If you pass a user name to the init method, the file is called **the-username.sqlite**. If you do not pass a user name, the file is called **jsonstore.sqlite** by default.
5. (Android only) Enable verbose JSONStore.

```
adb shell setprop log.tag.jsonstore-core VERBOSE
adb shell getprop log.tag.jsonstore-core
```

6. Use the debugger.

Common issues

Understanding the following JSONStore characteristics can help resolve some of the common issues that you might encounter.

- The only way to store binary data in JSONStore is to first encode it in base64. Store file names or paths instead of the actual files in JSONStore.
- Accessing JSONStore data from native code is possible only in IBM MobileFirst Platform Foundation V6.2.0.
- There is no limit on how much data you can store inside JSONStore, beyond limits that are imposed by the mobile operating system.
- JSONStore provides persistent data storage. It is not only stored in memory.
- The init API fails when the collection name starts with a digit or symbol. IBM Worklight V5.0.6.1 and later returns an appropriate error: `4 BAD_PARAMETER_EXPECTED_ALPHANUMERIC_STRING`
- There is a difference between a number and an integer in search fields. Numeric values like `1` and `2` are stored as `1.0` and `2.0` when the type is `number`. They are stored as `1` and `2` when the type is `integer`.
- If an application is forced to stop or crashes, it always fails with error code -1 when the application is started again and the `init` or `open` API is called. If this problem happens, call the `closeAll` API first.
- The JavaScript implementation of JSONStore expects code to be called serially. Wait for an operation to finish before you call the next one.
- Transactions are not supported in Android 2.3.x for Cordova applications.
- When you use JSONStore on a 64-bit device, you might see the following error: `java.lang.UnsatisfiedLinkError: dlopen failed: "..." is 32-bit instead of 64-bit`
- This error means that you have 64-bit native libraries in your Android project, and JSONStore does not currently work when you use these libraries. To confirm, go to **src/main/libs** or **src/main/jniLibs** under your Android project, and check whether you have the x86_64 or arm64-v8a folders. If you do, delete these folders, and JSONStore can work again.

Store internals

See an example of how JSONStore data is stored.

The key elements in this simplified example:

- `_id` is the unique identifier (for example, AUTO INCREMENT PRIMARY KEY).
- `json` contains an exact representation of the JSON object that is stored.
- `name` and age are search fields.
- `key` is an extra search field.

_idkeynameageJSON
1 c carlos99 {name: 'carlos', age: 99}
2 t tim 100{name: 'tim', age: 100}
When you search by using one of the following queries or a combination of them: `{_id : 1}, {name: 'carlos'}`, `{age: 99}, {key: 'c'}`, the returned document is `{_id: 1, json: {name: 'carlos', age: 99} }`.

The other internal JSONStore fields are:

- `_dirty`: Determines whether the document was marked as dirty or not. This field is useful to track changes to the documents.
- `_deleted`: Marks a document as deleted or not. This field is useful to remove objects from the collection, to later use them to track changes with your backend and decide whether to remove them or not.
- `_operation`: A string that reflects the last operation to be performed on the document (for example, replace).

JSONStore errors

JavaScript

JSONStore uses an error object to return messages about the cause of failures.

When an error occurs during a JSONStore operation (for example the `find`, and `add` methods in the `JSONStoreInstance` class) an error object is returned. It provides information about the cause of the failure.

```
var errorObject = {
  src: 'find', // Operation that failed.
  err: -50, // Error code.
  msg: 'PERSISTENT_STORE_FAILURE', // Error message.
  col: 'people', // Collection name.
  usr: 'jsonstore', // User name.
  doc: {_id: 1, {name: 'carlos', age: 99}}, // Document that is related to the failure.
  res: {...} // Response from the server.
}
```

Not all the key/value pairs are part of every error object. For example, the doc value is only available when the operation failed because of a document (for example the `remove` method in the `JSONStoreInstance` class) failed to remove a document.

Objective-C

All of the APIs that might fail take an error parameter that takes an address to an NSError object. If you don not want to be notified of errors, you can pass in `nil`. When an operation fails, the address is populated with an NSError, which has an error and some potential `userInfo`. The `userInfo` might contain extra details (for example, the document that caused the failure).

```
// This NSError points to an error if one occurs.
NSError* error = nil;

// Perform the destroy.
[JSONStore destroyDataAndReturnError:&error];
```

Java

All of the Java API calls throw a certain exception, depending on the error that happened. You can either handle each exception separately, or you can catch `JSONStoreException` as an umbrella for all JSONStore exceptions.

```
try {
  WL.JSONStore.closeAll();
}

catch(JSONStoreException e) {
  // Handle error condition.
}
```

List of error codes

Error code	Description
-100 UNKNOWN_FAILURE	Unrecognized error.
-75 OS*SECURITY*FAILURE	This error code is related to the requireOperatingSystemSecurity flag. It can occur if the destroy API fails to remove security metadata that is protected by operating system security (Touch ID with passcode fallback), or the init or open APIs are unable to locate the security metadata. It can also fail if the device does not support operating system security, but operating system security usage was requested.
-50 PERSISTENT*STORE*NOT_OPEN	JSONStore is closed. Try calling the open method in the JSONStore class class first to enable access to the store.
-48 TRANSACTION*FAILURE*DURING_ROLLBACK	There was a problem with rolling back the transaction.
-47 TRANSACTION*FAILURE*DURING*REMOVE*COLLECTION	

Error code	Description
-46 TRANSACTION_FAILURE_DURING_DESTROY	Cannot call destroy while there are transactions in progress.
-45 TRANSACTION_FAILURE_DURING_CLOSE_ALL	Cannot call closeAll while there are transactions in place.
-44 TRANSACTION_FAILURE_DURING_INIT	Cannot initialize a store while there are transactions in progress.
-43 TRANSACTION_FAILURE	There was a problem with transactions.
-42 NO_TRANSACTION_IN_PROGRESS	Cannot commit to rolled back a transaction when there is no transaction is progress
-41 TRANSACTION_IN_POGRESS	Cannot start a new transaction while another transaction is in progress.
-40 FIPS_ENABLEMENT_FAILURE	Problem getting the file information from the file system.
-24 JSON_STORE_FILE_INFO_ERROR	Problem replacing documents from a collection.
-23 JSON_STORE_REPLACE_DOCUMENTS_FAILURE	Problem removing documents from a collection.
-22 JSON_STORE_REMOVE_WITH_QUERIES_FAILURE	Problem storing the Data Protection Key (DPK).
-21 JSON_STORE_STORE_DATA_PROTECTION_KEY_FAILURE	Problem indexing input data.
-20 JSON_STORE_INVALID_JSON_STRUCTURE	Check that the types that you are passing to the searchFields are stringinteger,number, orboolean.
-12 INVALID_SEARCH_FIELD_TYPES	An operation on an array of documents, for example the replace method can fail while it works with a specific document. The document that failed is returned and the transaction is rolled back. On Android, this error also occurs when trying to use JSONStore on unsupported architectures.
-11 OPERATION_FAILED_ON_SPECIFIC_DOCUMENT	The accept function that the user provided returned false.
-10 ACCEPT_CONDITION_FAILED	To use offset, you must also specify a limit.
-9 OFFSET_WITHOUT_LIMIT	Validation error, must be a positive integer.
-8 INVALID_LIMIT_OR_OFFSET	Validation error (Must be [A-Z] or [a-z] or [0-9] only).
-7 INVALID_USERNAME	To log out, a JSONStore user must call the closeAll method first. There can be only one user at a time.
-6 USERNAME_MISMATCH_DETECTED	A problem with the destroy method while it tried to delete the file that holds the contents of the store.
-5 DESTROY_REMOVE_PERSISTENT_STORE_FAILED	Problem with the destroy method while it tried to clear the keychain (iOS) or shared user preferences (Android).
-4 DESTROY_REMOVE_KEYS_FAILED	Passed the wrong password to an encrypted store.
-3 INVALID_KEY_ON_PROVISION	

Error code	Description
-2 PROVISION*TABLES*SEARCH*FIELDS*MISMATCH	Search fields are not dynamic. It is not possible to change search fields without calling the destroy method or the removeCollection method before you call the init or openmethod with the new search fields. This error can occur if you change the name or type of the search field. For example: {key: 'string'} to {key: 'number'} or {myKey: 'string'} to {theKey: 'string'}.
-1 PERSISTENT*STORE*FAILURE	Generic Error. A malfunction in native code, most likely calling the init method.
0 SUCCESS	In some cases, JSONStore native code returns 0 to indicate success.
1 BAD*PARAMETER*EXPECTED_INT	Validation error.
2 BAD*PARAMETER*EXPECTED_STRING	Validation error.
3 BAD*PARAMETER*EXPECTED_FUNCTION	Validation error.
4 BAD*PARAMETER*EXPECTED*ALPHANUMERICS*TRING	Validation error.
5 BAD*PARAMETER*EXPECTED_OBJECT	Validation error.
6 BAD*PARAMETER*EXPECTED*SIMPLE*OBJECT	Validation error.
7 BAD*PARAMETER*EXPECTED_DOCUMENT	Validation error.
8 FAILED*TO*GET*UNPUSHED*DOCUMENTS*FROM*DB	The query that selects all documents that are marked dirty failed. An example in SQL of the query would be: SELECT * FROM [collection] WHERE _dirty > 0.
9 NO*ADAPTER*LINKED*TO*COLLECTION	To use functions like the push and load methods in the JSONStoreCollection class, an adapter must be passed to the init method.
10 BAD*PARAMETER*EXPECTED*DOCUMENT*OR*ARRAY*OF_DOCUMENTS	Validation error
11 INVALID*PASSWORD*EXPECTED*ALPHANUMERICS*TRING*WITH*LENGTH*GREATER*THAN_ZERO	Validation error
12 ADAPTER_FAILURE	Problem calling WL.Client.invokeProcedure, specifically a problem in connecting to the MobileFirst Server adapter. This error is different from a failure in the adapter that tries to call a backend.
13 BAD*PARAMETER*EXPECTED*DOCUMENT*OR_ID	Validation error
14 CAN*NOT*REPLACE*DEFAULT*FUNCTIONS	Calling the enhance method in the JSONStoreCollection class to replace an existing function (find and add) is not allowed.
15 COULD*NOT*MARK*DOCUMENT*PUSHED	Push sends the document to an adapter but JSONStore fails to mark the document as not dirty.

Error code	Description
16 COULD*NOT*GET*SECURE*KEY	To initiate a collection with a password there must be connectivity to the MobileFirst Server because it returns a 'secure random token'. IBM® Worklight® V5.0.6 and later allows developers to generate the secure random token locally passing {localKeyGen: true} to the init method via the options object.
17 FAILED*TO*LOAD*INITIAL*DATA*FROM*ADAPTER	Could not load data because WL.Client.invokeProcedure called the failure callback.
18 FAILED*TO*LOAD*INITIAL*DATA*FROM*ADAPTER*INVALID*LOAD_OBJ	The load object that was passed to the init method did not pass the validation.
19 INVALID*KEY*IN*LOAD*OBJECT	There is a problem with the key used in the load object when you call the add method.
20 UNDEFINED*PUSH*OPERATION	No procedure is defined for pushing dirty documents to the server. For example: the init method (new document is dirty, operation = 'add') and the push method (finds the new document with operation = 'add') were called, but no add key with the add procedure was found in the adapter that is linked to the collection. Linking an adapter is done inside the init method.
21 INVALID*ADD*INDEX_KEY	Problem with extra search fields.
22 INVALID*SEARCH*FIELD	One of your search fields is invalid. Verify that none of the search fields that are passed in are *id,json,*deleted, or _operation.
23 ERROR*CLOSING*ALL	Generic Error. An error occurred when native code called the closeAll method.
24 ERROR*CHANGING*PASSWORD	Unable to change the password. The old password passed was wrong, for example.
25 ERROR*DURING*DESTROY	Generic Error. An error occurred when native code called the destroy method.
26 ERROR*CLEARING*COLLECTION	Generic Error. An error occurred in when native code called the removeCollection method.
27 INVALID*PARAMETER*FOR*FIND*BY_ID	Validation error.
28 INVALID*SORT*OBJECT	The provided array for sorting is invalid because one of the JSON objects is invalid. The correct syntax is an array of JSON objects, where each object contains only a single property. This property searches the field with which to sort, and whether it is ascending or descending. For example: {searchField1 : ∾€œASC∾€}.

Error code	Description
29 INVALID*FILTER*ARRAY	The provided array for filtering the results is invalid. The correct syntax for this array is an array of strings, in which each string is either a search field or an internal JSONStore field. For more information, see Store internals.
30 BAD*PARAMETER*EXPECTED*ARRAY*OF_OBJECTS	Validation error when the array is not an array of only JSON objects.
31 BAD*PARAMETER*EXPECTED*ARRAY*OF*CLEAN*DOCUMENTS	Validation error.
32 BAD*PARAMETER*WRONG*SEARCH*CRITERIA	Validation error.