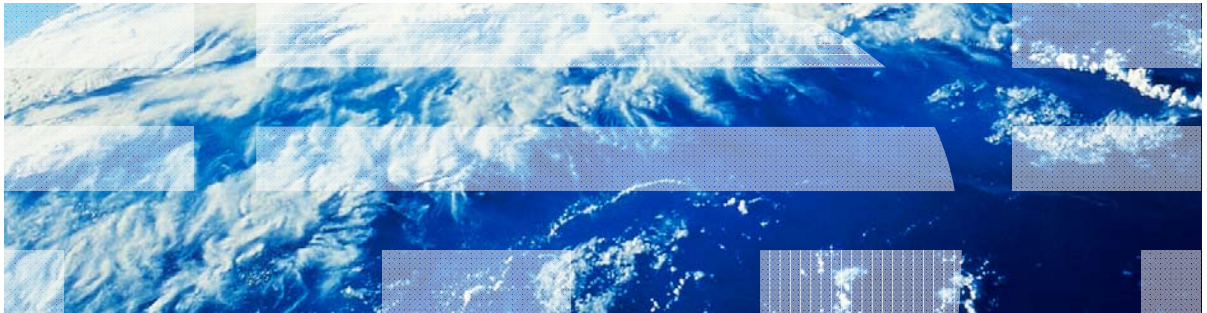


IBM Worklight V6.0.0 Getting Started

JSONStore – Common JSONStore Usage



Trademarks

- IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Worklight is a trademark or registered trademark of Worklight, an IBM Company. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)” at www.ibm.com/legal/copytrade.shtml.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
- Other company products or service names may be trademarks or service marks of others.
- This document may not be reproduced in whole or in part without the prior written permission of IBM.

About IBM®

- See <http://www.ibm.com/ibm/us/en/>

Introduction

- This module comprises two parts:
 - Basic usage: basic tasks that you can perform on a local JSONStore collection, such as **add** and **find**.
 - Advanced usage: some advanced tasks that, depending on your business requirements for data storage, you might find useful, such as **security**.
- The code samples that are available in this module are short on purpose.
 - Complete examples are available in the sample application that accompanies this module.
 - The screen captures that are used in this module derive from the sample application.

Agenda

- Basic Usage

- Initialize
- Get
- Add
- Find
- Replace
- Remove
- Remove Collection
- Destroy

- Advanced Usage

- Security
- Multiple user support
- Worklight adapter integration
- Enhance

Initialize – overview

- Use **init** to start one or more JSONStore collections.
- Starting or provisioning a collection means creating the persistent storage that contains collections and documents, if it does not exist.
- If the persistent storage is encrypted and a correct password is passed, the necessary security procedures to make the data accessible are run.
- For optional features that you can enable at initialization time, see **Security**, **Multiple user support**, and **Worklight adapter integration**, in the second part of this module.

Initialize – Example

```
var collectionName = 'people';

//Object that defines all the collections
var collections = {};

//Object that defines the 'people' collection
collections[collectionName] = {};

//Object that defines the Search Fields for the 'people' collection
collections[collectionName].searchFields = {name: 'string', age: 'integer'}

WL.JSONStore.init(collections)
    .then(function () {
        //handle success
    })
    .fail(function (errorObject) {
        //handle failure
    });
```

Initialize (Login/Open)

Get – Overview

- Use **get** to create an accessor to the collection.
- You must call **init** before you call **get**, otherwise the result of **get** is undefined.

Get – Example

```
var collectionName = 'people';  
var people = WL.JSONStore.get(collectionName);
```

- The variable **people** can now be used to perform operations on the **people** collection such as:
 - add
 - find
 - replace

Add – Overview

- Use **add** to store data as documents inside a collection.

Add – Example

```
var data = {name: 'carlos', age: 99};  
var collectionName = 'people';  
var options = {}; //default  
WL.JSONStore.get(collectionName)  
    .add(data, options)  
    .then(function () {  
        //handle success  
    })  
    .fail(function (errorObject) {  
        //handle failure  
    });
```

Find – Overview

- Use **find** to locate a document inside a collection by using a query.
- Use **findAll** to retrieve all the documents inside the collection.
- Use **findById** to search by the document unique identifier.
- The default behavior for **find** is to do a fuzzy search.

Find – Example

```
var query = {name: 'carlos'};
var collectionName = 'people';
var options = {
  exact: false, //default
  limit: 10 //returns a maximum of 10 documents, default: return every match
};

WL.JSONStore.get(collectionName)

  .find(query, options)

  .then(function (arrayResults) {
    //arrayResults = [{_id: 1, json: {name: 'carlos', age: 99}}]
  })

  .fail(function (errorObject) {
    //handle failure
  });
```

Search Field	
Limit (optional)	Offset (optional)
Find By Name (Fuzzy Search)	
Find By Age (Exact Search)	
Find All	
Enter Id	Find (by id)

Replace – Overview

- Use **replace** to modify documents inside a collection.
- The field that you use to perform the replacement is **_id**, the document *unique identifier*.

Replace – Example

```
var document = {_id: 1, json: {name:
  'carlitos', age: 99}};

var collectionName = 'people';

var options = {}; //default
```

```
WL.JSONStore.get(collectionName)
```

```
.replace(document, options)
```

```
.then(function () {
    //handle success
})
```

```
.fail(function (errorObject) {
    //handle failure
});
```

- This example assumes that the document `{_id: 1, json: {name: 'carlos', age: 99}}` is in the collection, and shows how to replace it with one where the name is 'carlitos'.

Remove – Overview

- Use **remove** to delete a document from a collection.
- Documents are not *erased* from the collection until you call **push**.
 - For more information, see the **Worklight adapter integration** section, later in this module.

Remove – Example

```
var query = {_id: 1};
var collectionName = 'people';
var options = {exact: true}; //default: false

WL.JSONStore.get(collectionName)

.remove(query, options)

.then(function () {
    //handle success
})

.fail(function (errorObject) {
    //handle failure
});
```


Remove Collection – Overview

- Use **removeCollection** to delete all the documents that are stored inside a collection.
 - This operation is similar to dropping a table in database terms.

Remove Collection – Example

```
var collectionName = 'people';

WL.JSONStore.get(collectionName)

.removeCollection()

.then(function () {
    //handle success
})

.fail(function (errorObject) {
    //handle failure
});
```

A rounded rectangular button with a light gray gradient and a thin border. The text "Remove Collection" is centered in a dark gray, sans-serif font.

Remove Collection

Destroy – Overview

- Use **destroy** to remove the following data:
 - All documents
 - All collections
 - All stores
 - See **Multiple user support** later in this module.
 - All JSONStore metadata and security artifacts
 - See **Security** later in this module.

Destroy – Example

```
var collectionName = 'people';

WL.JSONStore.destroy()

.then(function () {
    //handle success
})

.fail(function (errorObject) {
    //handle failure
});
```

Destroy Everything

Agenda

- Basic Usage
 - Initialize
 - Get
 - Add
 - Find
 - Replace
 - Remove
 - Remove Collection
 - Destroy
- Advanced Usage
 - Security
 - Multiple user support
 - Worklight adapter integration
 - Enhance

Security – Overview

- You can secure all the collections in a store by passing a password to the **init** function.
- If no password is passed, the documents of all the collections in the store are not encrypted.
- Data encryption is only available on Android and iOS environments.
- Some security metadata are stored:
 - In the Keychain, on iOS
 - In Shared Preferences, on Android
- The store is encrypted with a 256-bit Advanced Encryption Standard (AES) key. All keys are strengthened with Password-Based Key Derivation Function 2 (PBKDF2).
- Use **closeAll** to lock access to all the collections until you call **init** again.
 - If you think of **init** as a login function, you can think of **closeAll** as the corresponding logout function.
- Use **changePassword** to change the password.

Security – Example

```
var collections = {  
  people: {  
    searchFields: {name: 'string'}  
  }  
};  
  
var options = {  
  password: '123'  
};  
  
WL.JSONStore.init(collections, options)  
  
.then(function () {  
  //handle success  
})  
  
.fail(function (errorObject) {  
  //handle failure  
});
```

Multiple user support – Overview

- You can create multiple stores that contain different collections in a single Worklight application.
- The **init** function can take an options object with a user name.
- If no user name is given, the default user name is “**jsonstore**”.

Multiple user support – Example

```
var collections = {  
  people: {  
    searchFields: {name: 'string'}  
  }  
};  
  
var options = {  
  username: 'carlos'  
};  
  
WL.JSONStore.init(collections, options)  
  
.then(function () {  
  //handle success  
})  
  
.fail(function (errorObject) {  
  //handle failure  
});
```

Worklight adapter integration – Overview

- This section assumes that you are familiar with Worklight adapters.
- Worklight adapter integration is optional, and provides ways to:
 - Send data from a collection to a Worklight adapter.
 - Get data from a Worklight adapter into a collection.
- You can achieve these goals by using functions such as **WL.Client.invokeProcedure** or **jQuery.ajax** if you need more flexibility.
- In Worklight Studio, you can use a new wizard to create a template JavaScript file.
 - This creation is based on search fields that are selected from the back-end system that you used to communicate with the adapter.

Adapter implementation – Example

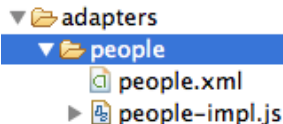
- First, create a Worklight adapter, and define its procedures to get, add, replace, and remove data (such as **getPeople**, **addPerson**, **replacePerson**, and **removePerson**).
- Next, implement these procedures.
 - The following example only returns hard-coded data and logs for simplicity.
 - For more information about how to create a Worklight adapter, see the modules under category 4, *Worklight server-side development*, in the IBM Worklight Information Center, at http://pic.dhe.ibm.com/infocenter/wrklght/v6r0m0/topic/com.ibm.worklight.help.doc/start/c_gettingstarted.html.

```
function getPeople () {
    return { peopleList : [{name: 'carlos', age: 100},
                          {name: 'tim', age: 99}] };
}

function addPerson (data) {
    return WL.Logger.debug('Got data from
JSONStore to ADD: ' + data);
}

function replacePerson (data) {
    return WL.Logger.debug('Got data from
JSONStore to REPLACE: ' + data);
}

function removePerson (data) {
    return WL.Logger.debug('Got data from
JSONStore to REMOVE: ' + data);
}
```



Initialize a collection that is linked to a Worklight adapter – Example

- Start the people collection by calling **init** and passing adapter metadata (adapter name, procedure names).
- In the example, the key is *peopleList* because the goal is to store {name: 'carlos', age: 100} and {name: 'tim', age: 99} as two separate documents.

```
var collections = {
  people: {
    searchFields: {
      name: 'string',
      age: 'integer'},
    //-- Start adapter metadata
    adapter : {
      name: 'people',
      add: 'addPerson',
      remove: 'removePerson',
      replace: 'replacePerson',
      load: {
        procedure: 'getPeople',
        params: [],
        key: 'peopleList'
      }
    }
  }
  //-- End adapter metadata
}; // (continued on the right column)
```

```
// (continued from the left column)

//Initialize
WL.JSONStore.init(collections)

.then(function () {
  //handle success
})

.fail(function (errorObject) {
  //handle failure
});
```

user (optional)

pass (optional)

Initialize (Login/Open)

Load data from a Worklight adapter – Example

- When **load** is called, JSONStore uses some metadata about the adapter (such as **name** and **procedure**), which you previously passed to **init**, to determine what data to get from the adapter and eventually store it.

```
WL.JSONStore.get('people').load()  
.then(function () {  
    //handle success  
})  
.fail(function (errorObject) {  
    //handle failure  
});
```

Load Data From Adapter

Get Push Required – Example

- Calling **getPushRequired** returns an array of so-called “dirty documents”, which are documents that have local modifications that do not exist on the back-end system.
 - These documents are sent to the Worklight adapter when **push** is called.

```
WL.JSONStore.get('people').getPushRequired()  
.then(function () {  
    //handle success  
})  
.fail(function (errorObject) {  
    //handle failure  
});
```

- To stop JSONStore from marking the documents as “dirty”, pass the option **{push: false}** to **add**, **replace**, and **remove**.

Get Push Required

Push – Example

- **push** sends the document that changed to the correct Worklight adapter procedure (for example: **addPerson** is called with a document that was added locally).
 - This mechanism is based on the last operation that is associated with the document that changed and the adapter metadata that is passed to init.

```
WL.JSONStore.get('people').push()  
.then(function (res) {  
    //handle success  
    //res is an empty array if all documents reached the server  
    //res is an array of error responses if some documents failed  
    to reach the server  
})  
.fail(function (errorObject) {  
    //handle failure  
});
```

Push Changes to Adapter

Enhance – Overview

- Use **enhance** to extend the core API to fit your needs, by adding functions to a collection prototype.

Enhance – Example

- The example shows how use **enhance** to add the function **getValue** that works on the **keyvalue** collection.
 - It takes a **key** (string) as its only parameter and returns a single result.
 - The call to **init** and how to use **enhance** to get a **put** method is shown in the code sample that is associated with this module.

//Definition:

```
var collectionName = 'keyvalue';
WL.JSONStore.get(collectionName).enhance(
  'getValue', function (key) {
    var deferred = $.Deferred(),
        collection = this;

    //Do an exact search for the key
    collection.find({key: key}, {exact:
true, limit: 1})
      .then(deferred.resolve,
deferred.reject);
    return deferred.promise();
  }); // (continued on the right column)
```

```
// (continued from the left column)
//Usage:
var key = 'myKey';
WL.JSONStore.get(collectionName).getValue
  (key)
  .then(function (res) {
    //res contains an array of documents
    //with the results from the find
  })
  .fail(function () {
    //handle failure
  });
```

Initialize (Login/Open)

Key

Value

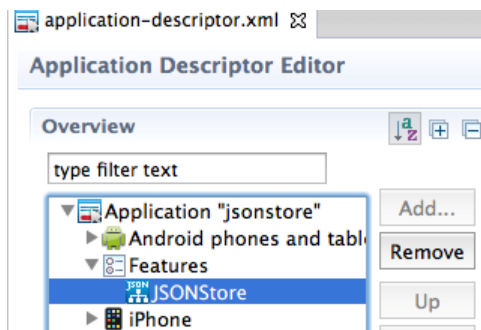
Add Key/Value Pair

Key

Find Value

Sample

- The sample for this training module can be found in the Getting Started page of the IBM Worklight documentation website at <http://www.ibm.com/mobile-docs>.
- To use the sample for this module, you must install IBM Worklight V6.0.0 Interim Fix (IF) 201307011413 or later.
- Make sure that the application enables the JSONStore feature.



For more information

- For more information about JSONStore (such as fuzzy search, Worklight Studio wizard, and other topics that are introduced in this module), see the JSONStore documentation in the IBM Worklight Information Center, at:
 - http://pic.dhe.ibm.com/infocenter/wrklight/v6r0m0/topic/com.ibm.worklight.help.doc/devref/c_jsonstore_overview.html
 - http://pic.dhe.ibm.com/infocenter/wrklight/v6r0m0/topic/com.ibm.worklight.help.doc/devref/c_wl_jsonstore_core_api.html for the API

Notices

- Permission for the use of these publications is granted subject to these terms and conditions.
- This information was developed for products and services offered in the U.S.A.
- IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.
- IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
 - IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.
- For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:
 - Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan
- **The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.**
- This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.
- Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
- IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.
- Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:
 - IBM Corporation
Dept F6, Bldg 1
294 Route 100
Somers NY 10589-3216
USA

- Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.
- The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.
- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

COPYRIGHT LICENSE:

- This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
- Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:
 - © (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp., enter the year or years. All rights reserved.

Privacy Policy Considerations

- IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.
- Depending upon the configurations deployed, this Software Offering may use session cookies that collect session information (generated by the application server). These cookies contain no personally identifiable information and are required for session management. Additionally, persistent cookies may be randomly generated to recognize and manage anonymous users. These cookies also contain no personally identifiable information and are required.
- If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent. For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy>, and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the sections entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Support and comments

- For the entire IBM Worklight documentation set, training material and online forums where you can post questions, see the IBM website at:
 - <http://www.ibm.com/mobile-docs>
- **Support**
 - Software Subscription and Support (also referred to as Software Maintenance) is included with licenses purchased through Passport Advantage and Passport Advantage Express. For additional information about the International Passport Advantage Agreement and the IBM International Passport Advantage Express Agreement, visit the Passport Advantage website at:
 - <http://www.ibm.com/software/passportadvantage>
 - If you have a Software Subscription and Support in effect, IBM provides you assistance for your routine, short duration installation and usage (how-to) questions, and code-related questions. For additional details, consult your IBM Software Support Handbook at:
 - <http://www.ibm.com/support/handbook>
- **Comments**
 - We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.
 - For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.
 - When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state.
 - Thank you for your support.
 - Submit your comments in the IBM Worklight forums at:
 - <https://www.ibm.com/developerworks/mobile/worklight/connect.html>
 - If you would like a response from IBM, please provide the following information:
 - Name
 - Address
 - Company or Organization
 - Phone No.
 - Email address

Thank You

