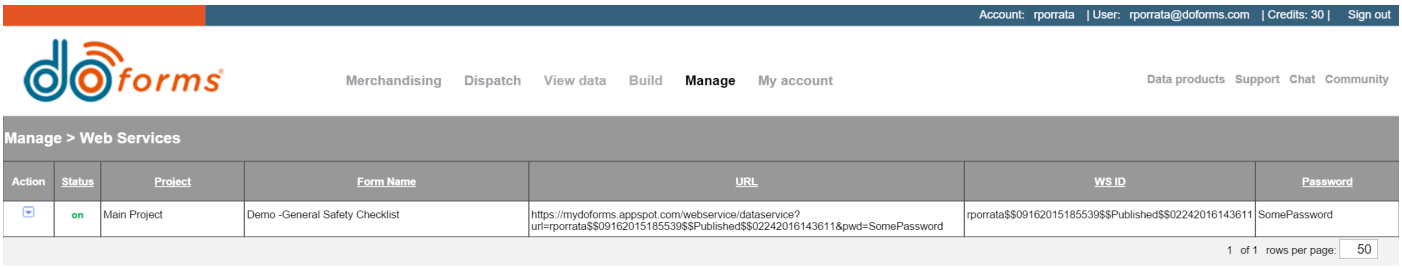
SOAP Services API

The SOAP API provides programmatic access to the doForms portal. These APIs enable you add, update and delete information used by your portal.

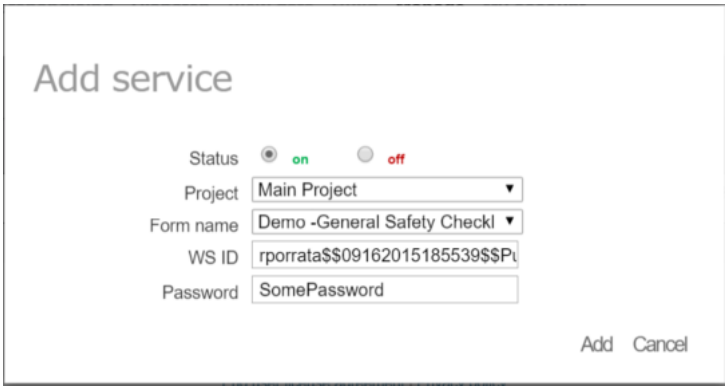
Enabling Web Services

Most of the web services work with the specified forms designed in the doForms portal. Others work with the dispatching, tracking and support data related items - lookup tables for example.

Each published **Project** / **Form** may be assigned a Web Service ID (WSID), and password for the services to work. This is done via the **Manage** > **Web Services** page.



Each service may be enabled and disabled based on the activities planned so you can manage which services should be in use.



These services are quota based, please see the section under Quota Limits section.

Quota Limits

Quota Limits are based on daily usage starting at midnight UTC.

- 50 Get Calls (i.e.: reaching out to retrieve / posting data)
- 50 Dispatch Calls

These limits are based on approx 100 operations per day

Note that during the 7 days immediately following the creation of a web service, we provide higher limits than those above in order to permit adequate testing of your software. This is done on a WSID by WSID basis. So if you need to extend this 7 days you can do so by deleting the doForms web service, and creating a new one for the same project/form (this will generate a new WSID).

Customers requiring higher use limits should contact support@doForms.com.

We reserve the right to change these limits at any time with or without notice in order to maintain the performance and reliability of the doForms website.

Connecting to a Web Service

Detailed instructions on how to connect with a SOAP web service are beyond the scope of this manual. SOAP based services are known to most IT professional experienced in web services.

doForms Web Services strictly follow the W3C SOAP Specification Version 1.2 (HTTP://www.w3.org/TR/soap/). We recommend using a tool such as soapUI (HTTP://www.soapui.org/) for exploring and testing a doForms web service. The WSDL file for a doForms web service looks something like the following, where the [WSID] and [Password] are provided in the Web Services tab (see above).

The current WSDL can be obtained at HTTP://www.mydoForms.com/wss?wsdl. The [URL] provide a quick view of the data in its XForms XML format and can be used to verify that the web service is being used correctly.

Reading Data

Most users of the doForms web services use them to retrieve data records submitted from mobile devices so that these data records can be integrated into another information system.

In most cases this process centers around one method: "getUnReadData". This method is used for reading data records that have not been previously read by your client application. The "DataFormat" argument determines if data is returned in CSV, JSON or XML.

The "NumberOfRec" determines how many records are returned with each call. Null records values will be returned when there are no more data records to read. We strongly recommend making this value less than 100 for reliability.

In the simplest case, this method is used with the "isAutoUpdate" argument equal to "1". Doing so automatically removes any record that is read with the getUnReadData from the unread data queue so it will not be re-read with subsequent calls of this method. In other cases, you may want to have more control over when a record is marked as read. For example, if performing integrity checks on each returned record. In this case, use the getUnReadData method with the "isAutoUpdate" argument equal to "0". Then use the "markUnReadDataAsRead" method to manually mark successfully read records from the unread record queue.

If you need more control over which records are read than described above, please consider using the getRecordKeyByReceivedDateRange and getDataByRecordKey methods which will enable you to identify and get specific records based on the date:time in UTC when a record was received by the mydoForms website. Alternatively, use the getRecordKeyByDateRange method if you want to do this based on the "Date_Created" which is the date:time that the record was created on the mobile device(not recommended).

With all of the "get" methods above, you may also want to delete data records from the doForms website after they have been read. In this case, use the "deleteDatabyRecordKey" method . Note that even if the record is to be deleted, it must also be marked as "marked as read" to avoid an error if using the getUnReadData method.

IMPORTANT NOTE:

- In doForms there is a distinction between a record that is newly submitted by a mobile device to the doForms website and a previously submitted record that is edited on the doForms website. The getUnReadData method reads both newly submitted records and edited records.
- See the Quota Limits section above before implementing any of the methods.

Downloading Media Files

The output of the web services contains the HTML links for each media source file. With some coding, you can create a program sending an HTTP request via the HTML links to download the media source files automatically.

Integration Tips / Best Practices

Services have a quota limit; depending on the activities done, your calls may be rejected if you hit your quota limit.

Keys are unique across the system and are rather large, for integration, consider using a Fact Table[1] that keeps your keys and submitted data's keys. This will enable clean integration between the two systems.

The receive date & times are the most reliable method preventing overlapping record ranges. Since doForms works in connected ***AND*** disconnected manner, the user may submit records after the period expected.

Below are the current recommended methods provided by the doForms web services API.

Note that in all cases the "[WS ID]" and "[Password]" refer to a specific doForms project/form web service.

The API methods described herein use two different ways of referencing specific records. Please be sure to use to the correct reference when using a specific method:

RecordId – This is the same as the "Form_Record" or "Record Name" field displayed in the View Data and Dispatch tabs and included in exports.
RecordKey – This is an internal database reference that is more efficient than RecordID.

Some of the API methods described here use and/or return two different date:time stamps:

Date_Created – This is the date:time when a record is first saved as complete on a mobile device (or on the website if it was first created there.)
Date_Recieved – This is the date:time when a record was first received by the website. Note that this is the most reliable date:time for specifying "non-overlapping" record ranges.

All date/time values are returned in the UTC/GMT time zone. However you can use the Time Zone parameter that is returned by the newest version of the getDataByRecordKey method to calculate the local Date_Created.

Service Endpoints:

The production end point may be found at this listed site url below.

SITE	URL
Production	https://wss-dot-mydoForms-hrd.appspot.com/wss?wsdl

Service Methods & Parameters

Listing of the service methods and descriptions. Clicking on the link will bring you to the associated section.

Please note the latest method may have a number appended to its name - this would be the revision of the method due to deprecation or changes to the interface. The Depreciated Methods are listed at the end of this document.

NAME	DESCRIPTION
addLookupTableData	Add a new Lookup Table. Note that this method is subject to the Use Limits described above.
appendLookupTableData	Append new rows to an existing Lookup Table. Note that this method is subject to the Use Limits described above.
deleteData	Delete records by the record Id
deleteDataByRecordKey	Delete records by the RecordKey
deleteDispatch	Deletes Dispatch records based on Ids
deleteLookupTableData	Delete all rows of an existing Lookup Table.
getDataByRecordKey	Get the detail of a record with the specified RecordKey in a specified format.
getFormRecordCount	Get the total records for a specific form
getFormTemplate2	get the form structure in XML format
getGPSTrackingPoints	Get all the tracking points that are within a certain date range (created and received on the mobile devices) for a specific mobile unit.
getLookupTableData	Get the content of a Lookup Table. Note that this method is subject to the Use Limits described above.
getLookupTableList	Return the Lookup Table list that includes only the Lookup Table key, name, description and the column list.
getMobileUnits	Get a list of all the mobile units connected to an account
getRecordKeyByDateRange	Return a list of the RecordKeys in a given date range.

getRecordKeyByEditedDateRange	Return a list of the RecordKeys in a given date range when the records are edited/updated. time that the record is received by the website.
getRecordKeyByReceivedDateRange	Return a list of the RecordKeys in a given received date range which is the date
getUnReadData	Just download records that have not been previously sent included the new records. Note that this method is subject to the Use Limits.
getWsIDList2	Return the list of active web service WSID's for an account which matching the password
listDispatch	Return a list of Records headers based On Status Flag, only included RecordKey, and base header fields(MobileNumber, Date Completed, Date Submitted).
markUnReadDataAsRead	Mark the unread data as "read" to remove the corresponding record from the getUnReadData call
submissionDispatch	Submit a record to given WebService.
updateLookupTableData	Update an existing Lookup Table. Note that this method is subject to the Use Limits described above.

Data Related

NAME	DESCRIPTION
addLookupTableData	Add a new Lookup Table. Note that this method is subject to the Use Limits described above.
appendLookupTableData2	Append new rows to an existing Lookup Table. Note that this method is subject to the Use Limits described above.
deleteData	Delete records by the record Id
deleteDataByRecordKey	Delete records by the RecordKey
deleteDispatch	Deletes Dispatch records based on Ids
deleteLookupTableData	Delete all rows of an existing Lookup Table.
getDataByRecordKey	Get the detail of a record with the specified RecordKey in a specified format.
getFormRecordCount	Get the total records for a specific form
getFormTemplate2	get the form structure in XML format
getGPSTrackingPoints	Get all the tracking points that are within a certain date range (created and received on the mobile devices) for a specific mobile unit.
getLookupTableData	Get the content of a Lookup Table. Note that this method is subject to the Use Limits described above.
getLookupTableList	Return the Lookup Table list that includes only the Lookup Table key, name, description and the column list.
getMobileUnits	Get a list of all the mobile units connected to an account
getRecordKeyByDateRange	Return a list of the RecordKeys in a given date range.
getRecordKeyByEditedDateRange	Return a list of the RecordKeys in a given date range when the records are edited/updated. time that the record is received by the website.
getRecordKeyByReceivedDateRange	Return a list of the RecordKeys in a given received date range which is the date
getUnReadData	Just download records that have not been previously sent included the new records. Note that this method is subject to the Use Limits.
getWsIDList2	Return the list of active web service WSID's for an account which matching the password
listDispatch	Return a list of Records headers based On Status Flag, only included RecordKey, and base header fields(MobileNumber, Date Completed, Date Submitted).
markUnReadDataAsRead	Mark the unread data as "read" to remove the corresponding record from the getUnReadData call
submissionDispatch	Submit a record to given WebService.
updateLookupTableData	Update an existing Lookup Table. Note that this method is subject to the Use Limits described above.

Service Methods

getWsIDList2
Return the list of active web service WSID's for an account which matching the password

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getWsIDList2>
 <arg0_java_lang_String>[Account Name]</arg0_java_lang_String>
 <arg1_java_lang_String>[WS Password]</arg1_java_lang_String>
 <arg2_int>[ResponseFormat]</arg2_int>
 </ser:getWsIDList2>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [Account Name] is the Account name or Website name
- [WS Password] is the WS Password
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

getFormTemplate2
Get the form structure in XML format

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getFormTemplate2>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 </ser:getFormTemplate2>
 </soapenv:Body>
</soapenv:Envelope>
```

checkValidWebservice
Check if a WS is valid or not

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:checkValidWebservice>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
</ser:checkValidWebservice>
 </soapenv:Body>
</soapenv:Envelope>
```

Return: true: the WS is valid false: the WS is invalid

getFormRecordCount
Get the total records for a specific form

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getFormRecordCount>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
</ser:getFormRecordCount>
```

```
 </soapenv:Body>
</soapenv:Envelope>
```

Return: number of records

getUnReadData

Just download records that have not been previously sent included the new records. Note that this method is subject to the quota limits.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getUnReadData>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_int>[DataFormat]</arg2_int>
 <arg3_int>[NumberOfRec]</arg3_int>
 <arg4_int>[isAutoUpdate]</arg4_int>
 </ser:getUnReadData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [DataFormat]: 1=CSV, 2=XML
- [NumberOfRec]: the number of records to be downloaded. Due to the existing limits from GAE, we recommend to just read less than 100 records for each call.
- [isAutoUpdate]: 1-the downloaded records will be automatically set to "read" and will not be returned in next calls; 0-the downloaded records are still kept as "unread"
- Note: If using isAutoUpdate=0, you should use the markUnReadDataAsRead method to manually mark the records as read in order to avoid service limitations on number of records read daily.

markUnReadDataAsRead

Mark the unread data as "read" to remove the corresponding record from the getUnReadData call

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
<soapenv:Header/>
 <soapenv:Body>
 <ser:markUnReadDataAsRead>
 <arg0_java_lang_String>[RecordKey]</arg0_java_lang_String>
 </ser:markUnReadDataAsRead>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [RecordKey]: The RecordKey which is extracted from the "@recordKey" field returned getUnReadData method.

getRecordKeyByReceivedDateRange

Return a list of the RecordKeys in a given received date range which is the date:time that the record is received by the website.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getRecordKeyByReceivedDateRange>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[From Date]</arg2_java_lang_String>
 <arg3_java_lang_String>[To Date]</arg3_java_lang_String>
```

```
 </ser:getRecordKeyByReceivedDateRange>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [From Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format
- [To Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format

getRecordKeyByEditedDateRange
Return a list of the RecordKeys in a given date range when the records are edited/updated.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getRecordKeyByEditedDateRange>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[From Date]</arg2_java_lang_String>
 <arg3_java_lang_String>[To Date]</arg3_java_lang_String>
 </ser:getRecordKeyByEditedDateRange>
 </soapenv:Body>
</soapenv:Envelope>
```

Where: - [From Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format - [To Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format

getRecordKeyByDateRange
Return a list of the RecordKeys in a given date range.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getRecordKeyByDateRange>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[From Date]</arg2_java_lang_String>
 <arg3_java_lang_String>[To Date]</arg3_java_lang_String>
 </ser:getRecordKeyByDateRange>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [From Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format
- [To Date]: the UTC date:time in MM/dd/yyyy HH:mm:ss format

getDataByRecordKey
Get the detail of a record with the specified RecordKey in a specified format. This newest version of the getDataByRecordKey method also returns the Time Zone for the record Date_Created (which is always returned in UTC/GMT). Note that this method is subject to the Use Limits described above.

Note: getDataByRecordKey7 is the latest Version of this call

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getDataByRecordKey>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
```

```
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[RecordKey]</arg2_java_lang_String>
 <arg3_int>[ResponseFormat]</arg3_int>
 <arg4_int>[DatetimeFormat]</arg4_int>
 </ser:getDataByRecordKey>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [RecordKey]: The RecordKey which is extracted from the "@recordKey" field returned by the "getRecordKeyByDateRange" method
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON
- [DatetimeFormat]:
- - "0" : yyyy-MM-ddTHH:mm:ss (UTC/GMT)
- - "1" : MM/dd/yyyy HH:mm:ss (UTC/GMT)

submissionDispatch
Submit a record to given WebService.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:submissionDispatch>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[recordsToSubmit]</arg2_java_lang_String>
 </ser:submissionDispatch>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [recordsToSubmit]is a CSV data string including headers and corresponding values concatenated by @END_LINE;

For example:

```
"@mobileNumber","string_question","integer_question","decimal_question","date_question","Time_question","Date_Time_question","select_mult
```

The "@mobileNumber" is a system field. If this field is blank or omitted, the submitted record will be set to "pending". Otherwise, this record will be automatically set to "sent" and delivered to the mobile unit whose mobile number is set in this field.

listDispatch
Return a list of Records headers based On Status Flag, only included RecordKey, and base header fields(MobileNumber, Date Completed, Date Submitted).

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:listDispatch>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_int>[DispatchStatus]</arg2_int>
 <arg3_int>[ResponseFormat]</arg3_int>
 </ser:listDispatch>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [DispatchStatus]: 1=Pending, 2=Scheduled, 3=Sent, 4=Received, 5=Viewed, 6=Rejected, 7=Completed, 100=All

- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

deleteDataByRecordKey
Delete records by the RecordKey

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:deleteDataByRecordKey>
 <arg0_java_lang_String>[RecordKeys]</arg0_java_lang_String>
 </ser:deleteDataByRecordKey>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [RecordKeys]: The list of the RecordKey separated by a comma (",").

The RecordKey which is extracted from the "@recordKey" field returned by the "getRecordKeyByDateRange" method.

Returns: true: the WS is valid false: the WS is invalid

deleteData
Delete records by the record Id

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:deleteData>
 <arg0_java_lang_String>[Record IDs]</arg0_java_lang_String>
 </ser:deleteData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [Record IDs] The list of the record ID separated by a comma (","). Note that the record ID is the "Form_Record" field on the View Data tab.

Return: 0: Success -1: System Error -2: No record found

deleteDispatch
Deletes Dispatch records based on Ids

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:deleteDispatch>
 <arg0_java_lang_String>[WS ID]</arg0_java_lang_String>
 <arg1_java_lang_String>[Password]</arg1_java_lang_String>
 <arg2_java_lang_String>[RecordKeys]</arg2_java_lang_String>
 <arg3_int>[ResponseFormat]</arg3_int>
 </ser:deleteDispatch>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [RecordKeys]: the list of the RecordKey concatenated by a comma
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

getLookupTableList

Return the Lookup Table list that includes only the Lookup Table key, name, description and the column list.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getLookupTableList>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_int>[ResponseFormat]</arg3_int>
 </ser:getLookupTableList>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

An example of a reponse in CSV format

```
[<?xml version="1.0" encoding="UTF-8" standalone=?>
<Root version="1.0">
<StatusCode></StatusCode>
<ErrorMessage>Success</ErrorMessage>
<Data>
<error_message/>
<error_code/>
<Result>
<![CDATA[key, tableName, description, columnsName
agIteWRvZm9ybXNyFQsSDkxvb2t1cFRhYmxlTXN0GIIGDA,"DS1","Sample 1","STATION,STATION_NAME,ELEVATION,LATITUDE,LONGITUDE,DATE,HLY-CLDHNORMAL,Co
agIteWRvZm9ybXNyFQsSDkxvb2t1cFRhYmxlTXN0GJEGDA,"DS2","Sample 2","STATION,STATION_NAME,ELEVATION,LATITUDE,LONGITUDE,DATE,HLY-CLDHNORMAL,Co
</Data>
</Root>]
```

getLookupTableData

Get the content of a Lookup Table. Note that this method is subject to the Use Limits described above.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getLookupTableData>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[LU KEY]</arg3_java_lang_String>
 <arg4_int>[OFFSET]</arg4_int>
 <arg5_int>[LIMIT]</arg5_int>
<arg6_int>[ResponseFormat]</arg6_int>
 </ser:getLookupTableData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [LU KEY]: The Lookup Table key which returned by getLookupTableList
- [OFFSET]: The row position will be returned
- [LIMIT]: The number of rows will be returned (***Note: return all rows if [LIMIT] is -1.***)
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

An example of a reponse in CSV format

```
<?xml version="1.0" encoding="UTF-8" standalone=?>
<Root version="1.0">
<StatusCode></StatusCode>
<ErrorMessage>Success</ErrorMessage>
<Data>
<error_message/>
<error_code/>
<Result><![CDATA["STATION","STATION_NAME","ELEVATION","LATITUDE","LONGITUDE","DATE","HLY-CLDH-NORMAL","Completeness Flag","HLY-HTDHNORMAL
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 00:00",,,"375",
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 01:00",,,"380",
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 02:00",,,"384",]]></Result>
</Data>
</Root>
```

addLookupTableData

Add a new Lookup Table. Note that this method is subject to the Use Limits described above.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:addLookupTableData>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[LU NAME]</arg3_java_lang_String>
 <arg4_java_lang_String>[DESCRIPTION]</arg4_java_lang_String>
 <arg5_java_lang_String>[DATA SOURCE]</arg5_java_lang_String>
 <arg6_int>[ResponseFormat]</arg6_int>
 </ser:addLookupTableData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [LU NAME]: The Lookup Table name
- [DESCRIPTION]: The Lookup Table description
- [DATA SOURCE]: The datasource in CSV format. The @END_LINE; is used as a carriage return for each row and the first row is the header.
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

For example:

```
"STATION","STATION_NAME","ELEVATION","LATITUDE","LONGITUDE","DATE","HLY-CLDH-NORMAL","Completeness Flag","HLY-HTDH-NORMAL","Completeness
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 00:00",,,"375",@END_LINE;
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 01:00",,,"380",
```

updateLookupTableData

Update an existing Lookup Table. Note that this method is subject to the Use Limits described above.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:updateLookupTableData>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[LU NAME]</arg3_java_lang_String>
 <arg4_java_lang_String>[DESCRIPTION]</arg4_java_lang_String>
 <arg5_java_lang_String>[LU KEY]</arg5_java_lang_String>
 <arg6_java_lang_String>[DATA SOURCE]</arg6_java_lang_String>
 <arg7_int>[ResponseFormat]</arg7_int>
 </ser:updateLookupTableData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [LU NAME]: The Lookup Table name
- [DESCRIPTION]: The Lookup Table description
- [LU KEY]: The Lookup Table key which returned by getLookupTableList
- [DATA SOURCE]: The datasource in CSV format. The @END_LINE; is used as a carriage return for each row and the first row is the header. Note: the number of fields in the new datasource must be same withthe current datasource.
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

For example (Data Source List):

```
"STATION","STATION_NAME","ELEVATION","LATITUDE","LONGITUDE","DATE","HLY-CLDH-NORMAL","Completeness Flag","HLY-HTDH-NORMAL","Completeness
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 00:00",,,"375",@END_LINE;
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 01:00",,,"380",
```

appendLookupTableData2

Append new rows to an existing Lookup Table. Note that this method is subject to the Use Limits described above.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:appendLookupTableData>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[LU NAME]</arg3_java_lang_String>
 <arg4_java_lang_String>[DESCRIPTION]</arg4_java_lang_String>
 <arg5_java_lang_String>[LU KEY]</arg5_java_lang_String>
 <arg6_java_lang_String>[DATA SOURCE]</arg6_java_lang_String>
 <arg7_int>[ResponseFormat]</arg7_int>
 </ser:appendLookupTableData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms

- [PASSWORD]: The password which used to login into mydoForms
- [LU NAME]: The Lookup Table name
- [DESCRIPTION]: The Lookup Table description
- [LU KEY]: The Lookup Table key which returned by getLookupTableList
- [DATA SOURCE]: The datasource in CSV format. The @END_LINE; is used as a carriage return for each row and the first row is the header. Note: the number of fields in the new datasource must be same with the current datasource.
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

For example:

```
"STATION","STATION_NAME","ELEVATION","LATITUDE","LONGITUDE","DATE","HLY-CLDH-NORMAL","Completeness Flag","HLY-HTDH-NORMAL","Completeness
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 00:00",,,"375",@END_LINE;
"GHCND:USW00003947","KANSAS CITY INTERNATIONAL AIRPORT MO US","306.3","39.2972","-94.7306","20100101 01:00",,,"380",
```

deleteLookupTableData
Delete all rows of an existing Lookup Table.

```
<soapenv:Envelope
xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:deleteLookupTableData>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[LU KEY]</arg3_java_lang_String>
 <arg4_int>[ResponseFormat]</arg4_int>
 </ser:deleteLookupTableData>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [LU KEY]: The Lookup Table key which returned by getLookupTableList
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

getMobileUnits
Get a list of all the mobile units connected to an account

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getMobileUnits>
 <arg0_java_lang_String>>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</</arg2_java_lang_String>
 <arg3_int>[ResponseFormat]</arg3_int>
 </ser:getMobileUnits>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms

- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON

getGPSTrackingPoints
Get all the tracking points that are within a certain date range (created and received on the mobile devices) for a specific mobile unit.

```
<soapenv:Envelope xmlns:soapenv="HTTP://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ser="services.wss.portal.doForms.mdt.com">
 <soapenv:Header/>
 <soapenv:Body>
 <ser:getGPSTrackingPoints>
 <arg0_java_lang_String>[ACCOUNT NAME]</arg0_java_lang_String>
 <arg1_java_lang_String>[EMAIL]</arg1_java_lang_String>
 <arg2_java_lang_String>[PASSWORD]</arg2_java_lang_String>
 <arg3_java_lang_String>[MOBILE KEY]</arg3_java_lang_String>
 <arg4_java_lang_String>["FROM" DATE]</arg4_java_lang_String>
 <arg5_java_lang_String>["TO" DATE]</arg5_java_lang_String>
 <arg6_int>[ResponseFormat]</arg6_int>
 </ser:getGPSTrackingPoints>
 </soapenv:Body>
</soapenv:Envelope>
```

Where:

- [ACCOUNT NAME]: your doForms account name
- [EMAIL]: The email address which used to login into mydoForms
- [PASSWORD]: The password which used to login into mydoForms
- [MOBILE KEY]: The mobile key which returned by the getMobileUnits method
- ["FROM" DATE]: the date:time when the GPS values are created and received on the mobile device
- ["FROM" DATE]: the date:time when the GPS values are created and received on the mobile device
- [ResponseFormat]: 1=CSV, 2=XML, 3=JSON, 4=KML, 5=GPX

Depreciated Methods

Although deprecated methods remain in the API, their use is discouraged, and deprecation may indicate that the feature will be removed in the future.

Features are deprecated — rather than immediately removed — in order to provide backward compatibility, and give programmers who have used the feature enough time to bring their code into compliance with the new standard.

The following methods are depreciated:

- appendLookupTableData
- deleteData
- getFormTemplate
- getFormDataWithDateRange
- getDataByRecordKey
- getDataByRecordKey2
- getDataByRecordKey3
- getDataByRecordKey4
- getDataByRecordKey5
- getDataByRecordKey6
- getGPSTrackingPointsByMobileNumber
- getRecordKeyByDateRange
- getRecordKeyByDateRange2
- getRecordKeyByEditedDateRange
- getRecordKeyByEditedDateRange2
- getRecordKeyByReceivedDateRange
- getRecordKeyByReceivedDateRange2
- getUnReadData
- getUnReadData2

- getWsIDList
- getWsIDList2
- submissionDispatch

Sample Code

Please note these sample are not meant for production use, rather, they are used to showcase how certain calls are made within an application.

Review the documentation and services needed prior to using any code listed here for your needs.

LANGUAGE	URL
Java	HTTP://beta.mydoForms-hrd.appspot.com/support/doForms_WS_Java_Sample.zip
VB.Net	HTTP://beta.mydoForms-hrd.appspot.com/support/doForms_WS_VBNet_Sample.zip
PHP	HTTP://beta.mydoForms-hrd.appspot.com/support/doForms_WS_PHP_Sample.zip

Modification History

- 2016.02.20
  - Updated documentation based on changes made from 2013 documents
  - Updated deprecated functions section

Links

1. https://www.google.com/search?q=fact+table