HOBOlink® Web Services V2 Developer's Guide

Onset Computer Corporation
470 MacArthur Blvd.
Bourne, MA 02532
www.onsetcomp.com

Mailing Address:
P.O. Box 3450
Pocasset, MA 02559-3450

Phone: 1-800-LOGGERS (1-800-564-4377) or 508-759-9500
Fax: 508-759-9100
Support: http://www.onsetcomp.com/support/contact

Technical Support Hours: 8AM to 8PM ET, Monday through Friday
Customer Service Hours: 8AM to 5PM ET, Monday through Friday

17382-F

Contents

Section 1: Overview

HOBOlink web services offer the ability for third-party developers and partners to write their own programs to extract HOBO® logger data (U30, RX3000, HOBOmobile®) from the HOBOlink database.

HOBOlink exposes a set of REST web services hosted on Onset's remote servers. These web services are accessed through a third-party software program (the "caller"). The REST web services deliver customized datasets to the caller in several formats (Excel, CSV, HOBOwareCSV, and JSON). This relies on HOBOlink's underlying infrastructure where data is stored as individual readings, such that custom datasets can be extracted and sent to the caller.

Specific capabilities of HOBOlink REST web services include:

- Data from the logger is stored as individual entries in a data warehouse type of database rather than only in binary .dtf files. This provides a tremendous amount of extensibility for allowing customized access to a user's data, both public and private.
- Datasets returned to the caller can span multiple devices, launches, and wraps, as well as being constrained to only a small subset of a deployment.
- Datasets are customizable by time, device serial number, sensor serial number, and/or measurement type.
- Authentication is achieved via a token (provided by Onset) that is passed to HOBOlink as part of the web services call. HOBOlink manages access to the various web services using these tokens. Tokens will be provided free to customers who purchase HOBO devices.

If you wish to discuss the HOBO Remote Monitoring System and web services in more detail, please contact an Onset Computer Application Specialist at (800) 564-4377 or sales@onsetcomp.com.

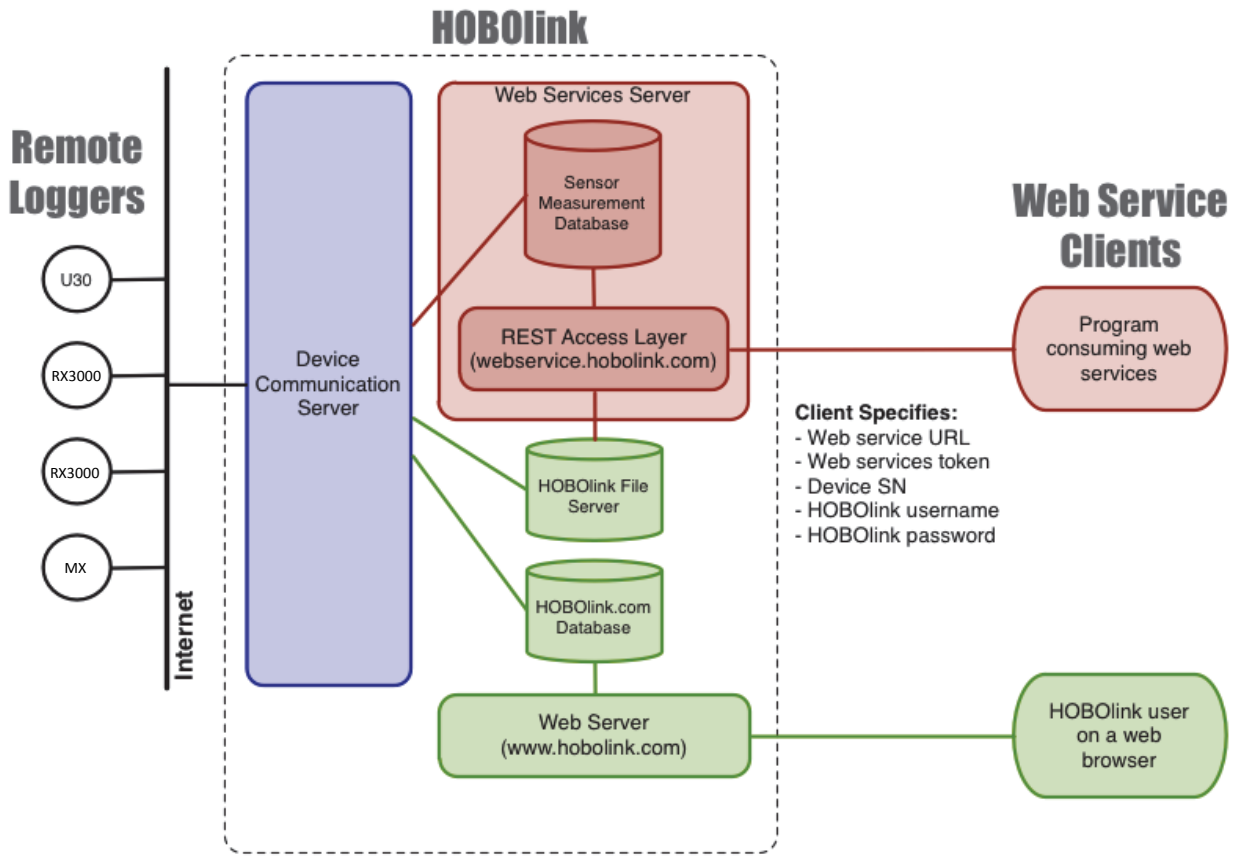
Requirements

- You should be a software engineer with a strong working knowledge of your programming language, its abilities, and its limitations.
- Ideally, you should have experience writing code that consumes REST web services.
- You should have a HOBOlink account and a Remote Monitoring Station and/or a MX series logger, or plans to purchase one in the near future.

Interaction between Web Services, Clients, and Devices

The following diagram shows the interaction between HOBOlink web services, clients, and devices, including:

- The difference between a user hitting HOBOlink and a program hitting the web services.
- How the web services get data from the database, not directly from the devices.
- The parameters the web service program needs to specify.



Section 2: REST Web Services Tutorial

Onset's REST web services allow you to easily obtain the latest data from both public and private loggers and incorporate them into your application. This section explains the URLs for obtaining the data using the REST web services.

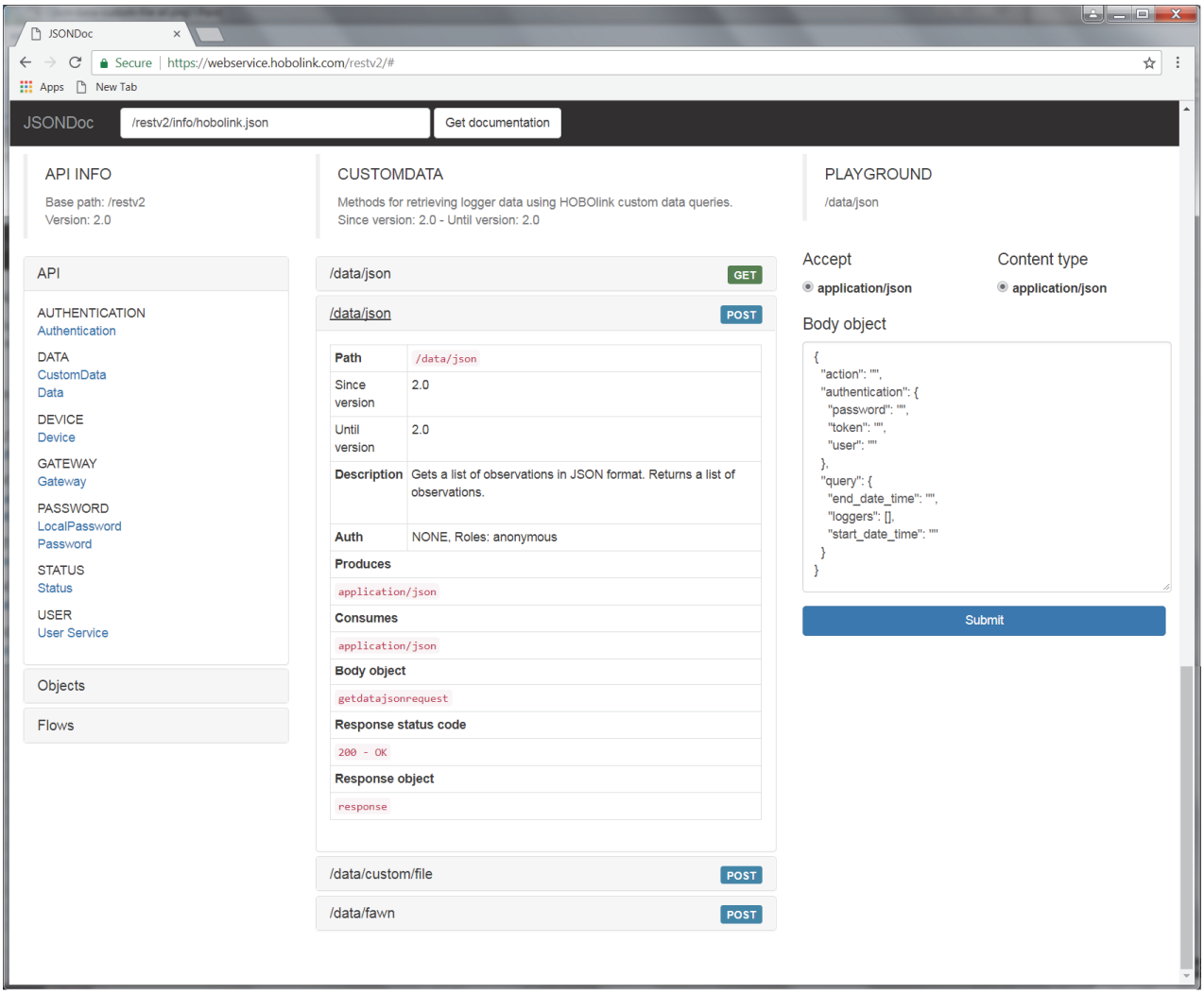
The REST web services are described in a JSONDoc website shown in the following sections. To access this site, go to https://webservice.hobolink.com/restv2 and click the Get documentation button. The JSONDoc site provides a playground section that allows you to interact with the web services and your data without writing any code.

Getting Data from Specific Loggers Over a Period of Time

Under API, click /data/json. Enter the password, token, and user for authentication. Under query, define the start and end date for the data you want to access from loggers.

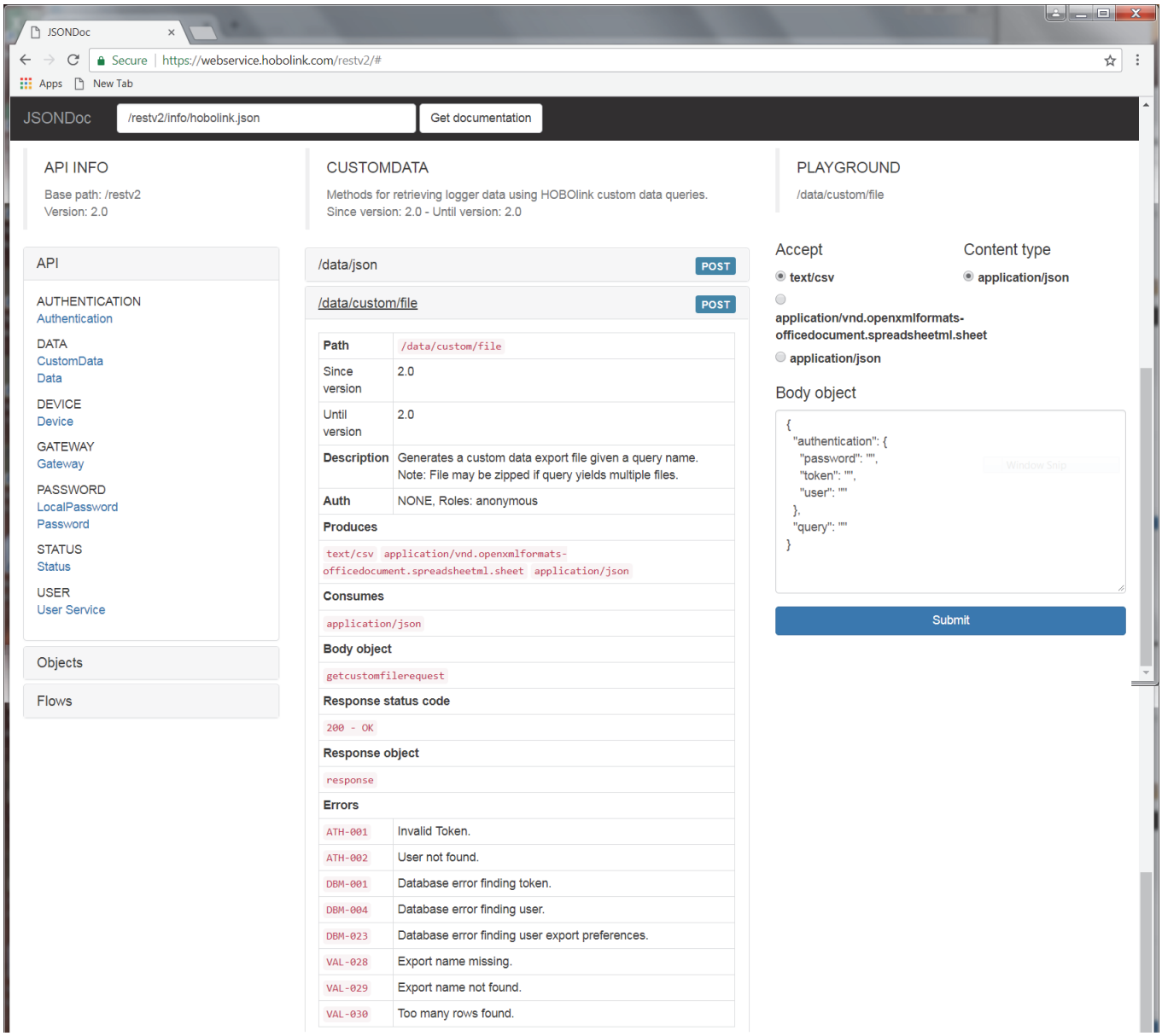
Click Objects in the left menu and then click "getdataquery" for instructions on the format of "end_date-time" and "start_date_time." List loggers (stations) by serial number, comma-delimited.

Notes: The data available is from active databases; archived data is not available for retrieval. Only one to two months of data will be retrieved at a time. This method is recommended for real-time data acquisition. For older data, use /data/custom/file described in the next section.



Getting Data Using a Custom HOBOlink Export

Under API, click /data/custom/file. You will need to enter the password, token, and user for authentication and the HOBOlink custom data export name for the query. See the next section for details on saving a query in HOBOlink.

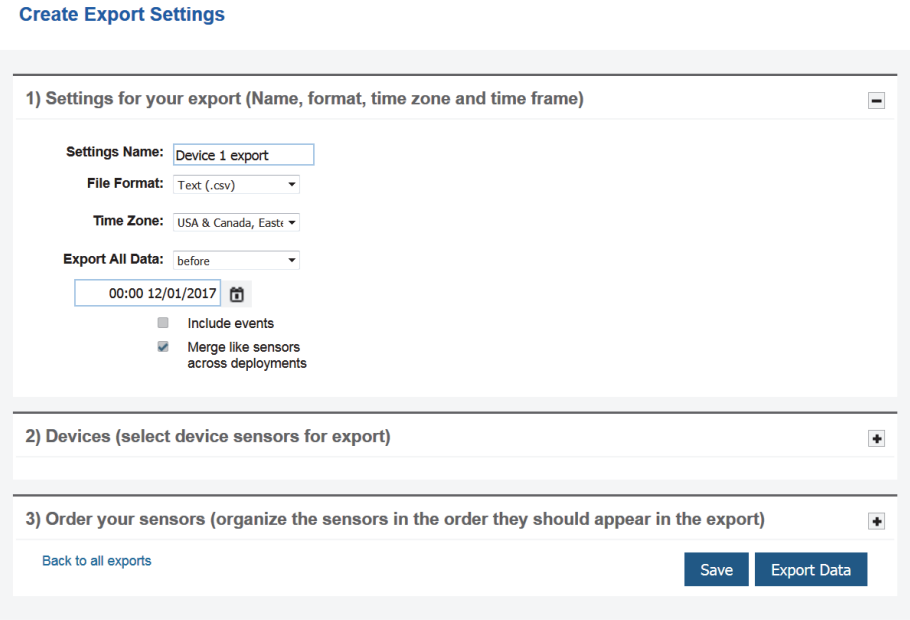


Setting up a HOBOlink Custom Data Export

To use the CustomData web service you must set up a custom data export in HOBOlink. To do this:

1. Click Data and then Exports. Click Create New Export.

2. Complete the details in all three sections on the Create Export Settings page below (see HOBOlink Help for details).



3. Save the export and then click Export Data to ensure the data results are as expected.

Once the export has the data you want, test it with the JSONDoc site in the previous section using your token, user info, and query (export).

Use the example code in Section 3 to call the web service programmatically.

Getting the Latest Data File for a U30 Device

If a device has been made public in HOBOlink (via HOBOlink User Settings), anyone can access its data via the public URL at http://webservice.hobolink.com/restv2/public/devices. Public services can be executed without authentication or the need to use SSL.

If a device has not been made public, its data can only be accessed via a private URL, which is http://webservice.hobolink.com/restv2/private/devices. Private services require authentication and must be executed over SSL. The authentication scheme is Basic Authentication over SSL (secure sockets layer). An authenticated user can only access data for devices that the user has registered in his or her HOBOlink account, or have been made public.

The data file service public URL follows this convention:

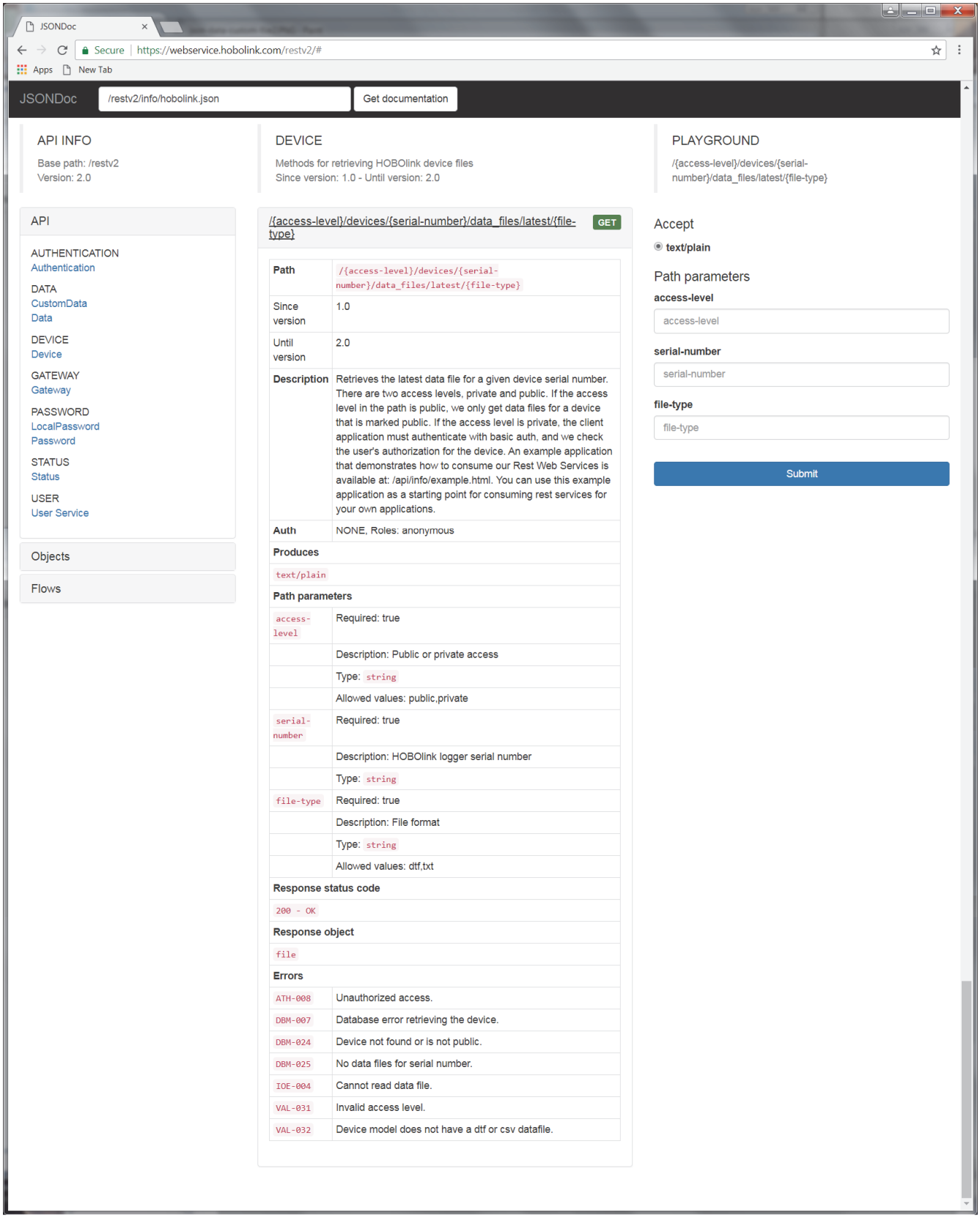
http://webservice.hobolink.com/restv2/public/devices/<serial_number>/data_files/latest/<data_file_type: dtf or txt>

and the data file service private URL follows this convention:

http://webservice.hobolink.com/restv2/private/devices/<serial_number>/data_files/latest/<data_file_type: dtf or txt>

where <serial_number> is the device serial number, and <data_file_type: dtf or txt> is either dtf (to get the dtf file) or txt (to get the text file).

On JSONDoc, click Device under API for more details and to test the web service.



Example URLS

Getting the latest .dtf file for a public device with serial number 123456:
http://webservice.hobolink.com/restv2/public/devices/123456/data_files/latest/dtf

Getting the latest .txt file for a public device with serial number 123456:
http://webservice.hobolink.com/restv2/public/devices/123456/data_files/latest/txt

Getting the latest .dtf file for a private device with serial number 123456:
https://webservice.hobolink.com/restv2/private/devices/123456/data_files/latest/dtf

Getting the latest .txt file for a private device with serial number 123456:
https://webservice.hobolink.com/restv2/private/devices/123456/data_files/latest/txt

Testing the Code

The URLs in the previous section point to a stable server with production user devices. It is strongly recommended that you get your code running, tested, and debugged against the HOBOlink development environment before switching to a stable server. To use the development instance of our REST web services for testing, use this URL for public devices:

http://webservice-dev.hobolink.com/restv2/public/devices/1216485/data_files/latest/dtf

and this URL for private devices:

https://webservice-dev.hobolink.com/restv2/private/devices/1216485/data_files/latest/dtf

Section 3: Sample REST Code

This section describes an example application that demonstrates how to consume the REST web services using Java or Visual Basic. If you plan to write your client in either language, you can use the example applications as a starting point for consuming REST web services for your own applications. If you plan to write your client in another language, refer to documentation provided by your API that describes how to consume REST web services in that language. Onset does not provide support for writing your own web service clients outside of what is contained in this document.

Overview

You can access Onset's REST web services using any language you choose, provided you have an understanding of how to access HTTP content in that language. While the example client provided is written in Java and Visual Basic, many other mainstream languages, such as Perl, have libraries you can use to consume RESTful web services.

The Java example uses a library called the Jersey Client API for RESTful Web Services. For information about the Jersey Client API, visit https://jersey.dev.java.net/ or search for "Consuming REST Web Services with Jersey" on the web. You may also use other Java libraries, such as Apache HttpClient at http://hc.apache.org/httpclient-3.x/.

To download the source code for this example application, go to:
https://webservice.hobolink.com/restv2/info/ExampleRestWebServicesClient.zip

Note: Onset supports the example code, but does not provide additional support with programming or building your application.

What the Example Application Does

The example code calls the custom data file web service to run the given data query (export) and download the resulting files in text or Excel format. After the example application retrieves the files, it saves them to the local file system.

Also, for U30 devices only, the example calls the private Onset REST device web service to get the latest data files for a given device's serial number in two formats. The first file retrieved from the service in the example is a binary .dtf file and the second is in text format (.csv). After the example application retrieves the files, it saves them to the local file system. This example also demonstrates how to authenticate over SSL (secure sockets layer) to the private Onset REST web services.

Note: There is also a public device web service that does not require authentication or SSL. To use this public service, skip the steps for SSL and authentication. The use of the client web service is the same. The only requirement for accessing a device's data from the public service is that the device has to have been made public. To make a device public, log onto HOBOlink, click the User Settings tab, and check the appropriate box under Preferences > By Device.

Running the Example Java Application

Requirements:

- Maven2 to build the project. Check the version of Maven installed by typing **mvn -version** on the command line. If you don't already have maven installed, visit http://maven.apache.org.
- Java 1.5 or later. Check the version of Java installed by typing **java –version** on the command line.

Build the Source and Run

1. Download the source at https://webservice.hobolink.com/restv2/info/ExampleRestWebServicesClient.zip
2. Extract the zip file to any location.
3. Open the file src/main/java/com/onset/test/rest/webservices/OnsetRestWebServiceExample.java in an IDE (NetBeans, Eclipse, IntelliJ, etc.), or import the pom.xml file.
4. Run the following Maven command:
   mvn compile
5. Run the main class in the IDE. You should see "success" printed out on the console.

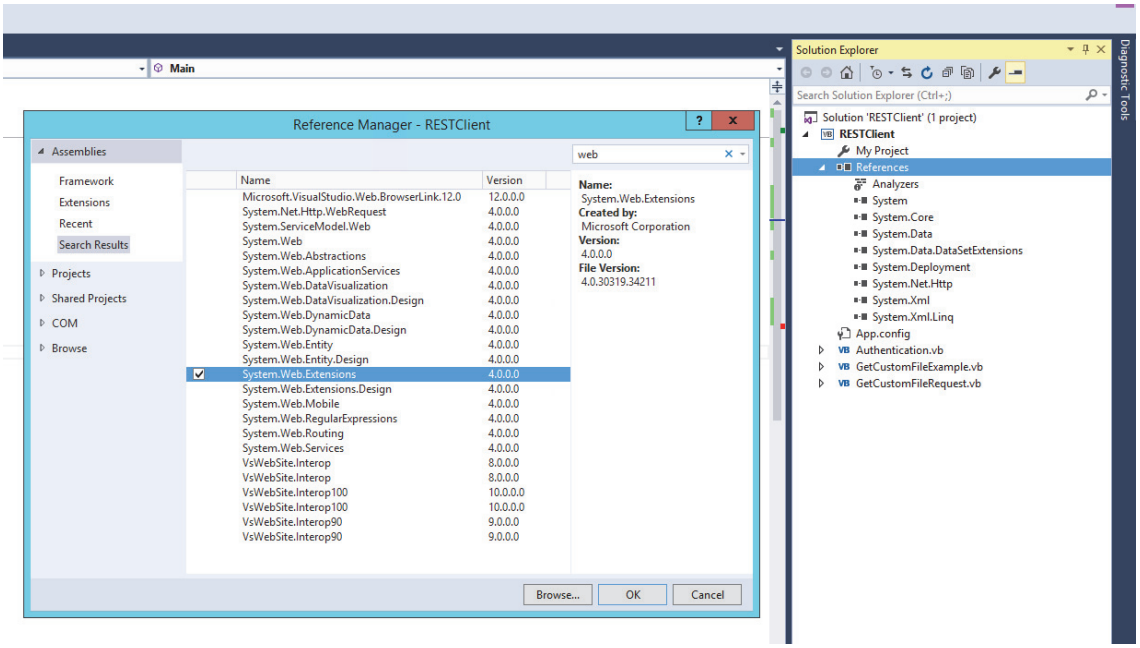
Running the Example Visual Basic Application

Requirements:

- Visual Studio

Build the Source and Run

1. Download the source at https://webservice.hobolink.com/restv2/info/ExampleRestWebServicesClient.zip
2. Extract the zip file to any location.
3. Create a new Visual Studio project.
4. Add the three VB files from the zip file (under src/main/vb) to your project.
5. Add the reference library **System.Web.Extensions** to the project as shown below.



6. Build the project. Once the example is running against the HOBOlink development server, contact Technical Support for a token. You can then switch the URL, token, user, password, and query name variables to the stable server.

Section 4: Reference

Error Codes

Configuration

- CFG-001 – Error loading logging properties file.
- CFG-002 – Error loading the webservice properties file.

Database

- DBM-001 – Database error validating token.
- DBM-002 – Database error retrieving communication plans by VAR.
- DBM-003 – Database error retrieving HOBOlink status.
- DBM-004 – Database error retrieving the user.
- DBM-005 – No active deployments for user.
- DBM-006 – Database error retrieving active deployments for user.
- DBM-007 – Database error retrieving the device by serial number.
- DBM-008 – Database error retrieving the communications plan.
- DBM-009 – Database error saving the device.
- DBM-010 – Database error retrieving the deployment by device.
- DBM-011 – No valid communication plans for the respective VAR.
- DBM-012 – No subscribers found.
- DBM-013 – Database error retrieving list of subscribers.
- DBM-014 – Database error retrieving subscriber.
- DBM-015 – Database error retrieving credit type.
- DBM-016 – Database error saving subscriber.
- DBM-017 – Database error retrieving subscriber's webservice.
- DBM-018 – Database error retrieving webservice.
- DBM-019 – Database error saving subscriber's webservice.
- DBM-020 – No text files for deployment.
- DBM-021 – Database error retrieving latest text file for deployment.
- DBM-022 – Logger communications plan is terminated.
- DBM-023 – Database error retrieving user's export preferences.
- DBM-024 – Device not found or is not public.
- DBM-025 – No data files for serial number.

Authentication

- ATH-001 – Invalid token.
- ATH-002 – Invalid user.

- ATH-003 – Encryption algorithm error during authentication.
- ATH-004 – Invalid password.
- ATH-005 – Invalid device serial number.
- ATH-006 – Invalid communications plan.
- ATH-007 – Invalid status.
- ATH-008 – Device not owned by respective user.

Email

- EML-001 – Error sending plan renewal confirmation email.

I/O

- IOE-001 – Could not read text file.
- IOE-002 – Error communicating with Sensor Observation Service.
- IOE-003 – The Sensor Observation Service reported an error.
- IOE-004 – Cannot read data file.

Validation

- VAL-001 – Time period start time is null.
- VAL-002 – Time period start time is in the future.
- VAL-003 – No data found in specified time range.
- VAL-004 – Device serial number or user name is required.
- VAL-005 – Unsupported preset time period.
- VAL-006 – Unsupported indeterminate value for time period.
- VAL-007 – A start time is required for time period with the "during" operator.
- VAL-008 – An end time is required for time period with the "during" operator.
- VAL-009 – A start time is required for time period with the "after" operator.
- VAL-010 – An end time is required for time period with the "before" operator.
- VAL-011 – Invalid time period operator.
- VAL-012 – An operator is required with a time period.
- VAL-013 – Sensor information is required with "greater than" filter operator.
- VAL-014 – The value is required with "greater than" filter operator.
- VAL-015 – Sensor information is required with "less than" filter operator.
- VAL-016 – The value is required with the "less than" filter operator.
- VAL-017 – Sensor information is required with "greater than or equal to" filter operator.
- VAL-018 – The value is required with the "greater than or equal to" filter operator.
- VAL-019 – Sensor information is required with "less than or equal to" filter operator.
- VAL-020 – The value is required with the "less than or equal to" filter operator.

- VAL-021 – Sensor information is required with "equal to" filter operator.
- VAL-022 – The value is required with the "equal to" filter operator.
- VAL-023 – Sensor information is required with "not equal to" filter operator.
- VAL-024 – The value is required with the "not equal to" filter operator.
- VAL-025 – Invalid filter operator.
- VAL-026 – An operator is required with a filter.
- VAL-027 – Request is null.
- VAL-028 – Invalid data query.
- VAL-029 – Data query not found.
- VAL-030 – Too many rows found for query.
- VAL-031 – Invalid access level.
- VAL-032 – Device model does not have a .dtf or .csv data file.