IQMS® Manufacturing Software becomes DELMIAWORKS

3DEXPERIENCE®

GROWING YOUR MANUFACTURING BUSINESS WITH APPLICATION PROGRAMMING INTERFACES (API)

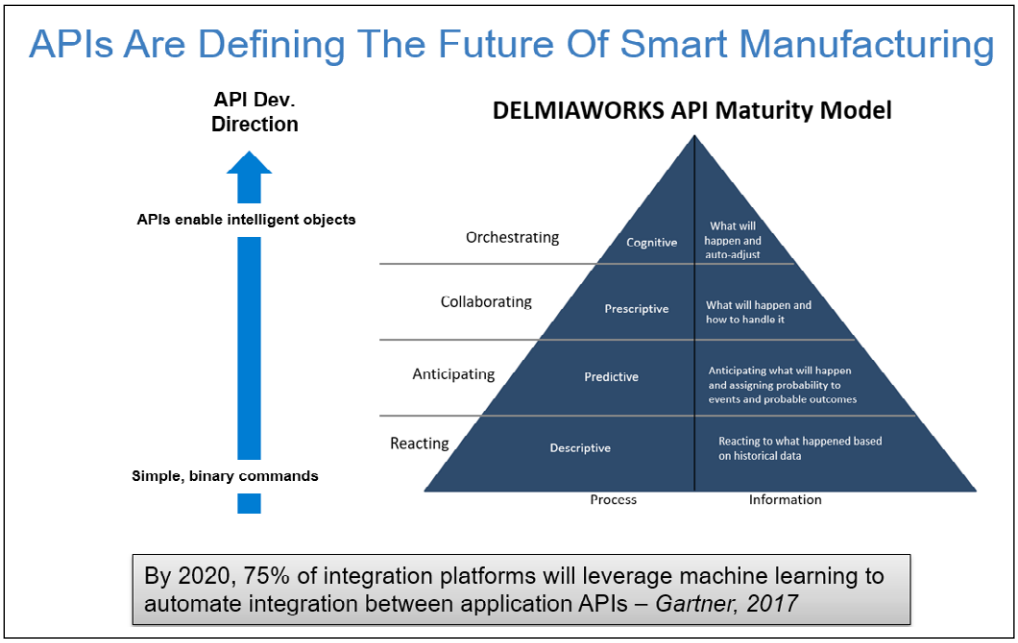
Whitepaper

DASSAULT SYSTEMES

EXECUTIVE SUMMARY

APIs (Application Programmer Interfaces) are the components that enable diverse platforms, apps, and systems to connect and share data with each other. Think of APIs as a set of software modules, tools, and protocols that enable two or more platforms, systems and most commonly, applications to communicate with each other and initiate tasks or processes. APIs are essential for defining and customizing diverse manufacturing processes that scale from integrating with suppliers, supporting transactions, pricing to defining the Graphical User Interfaces (GUIs) of each screen a production system uses. Cloud platform providers all have extensive APIs defined and work in close collaboration with development partners to fine-tune app performance using them. Dassault Systemes DELMIAWORKS and thousands of other companies have APIs available today to give their customers, partners and suppliers' greater agility and flexibility in customizing enterprise systems.

- **Customer needs are driving the most efficient API development programs.** Having a strong focus on the customer and being accountable for how the API's quality turns out is essential which is the design objective DELMIAWORKS is following in creating the APIs referenced in this white paper. Customer-centric development is also forcing APIs to scale up faster, providing contextual intelligence and insight over completing simple tasks. DELMIAWORK's customer-centric APIs are driving greater maturity into development cycles, enabling quicker maturity of API code bases across the board. The following DELMIAWORKS API Maturity Model provides the context of how APIs must progress to provide greater contextual intelligence to enable prescriptive and cognitive workflows.



- **Manufacturers and their IT teams are starting to focus more on unique API consumption strategies first.** Being able to orchestrate different APIs together and enable entirely new manufacturing business processes and models fast is what matters most. Orchestrating APIs and create real-time integration is a challenging task, however, especially between on-premise, legacy systems and cloud platforms and apps that manufacturers are relying on today.
- **APIs are becoming enablers of the next generation of manufacturing business models quickly.** The most complex APIs are being built within manufacturers who have the goal of providing a contextually intelligent real-time experience across all the suppliers they source from and the channels they sell through. This is a daunting task and one of the key factors driving greater maturity as shown in the DELMIAWORKS API Maturity Model.

BENEFITS OF WORKING WITH DELMIAWORK'S API FRAMEWORK

Accelerating Time-to-Value

The DELMIAWORKS API framework is a state-of-the-art, scalable and secure series of cloud integration services that enable IQMS customers to connect on-premise, cloud and Hosted Managed Services (HMS) without requiring any new hardware while also Providien the flexibility of customizing their WebIQ implementations. The DELMIAWORKS API framework delivers exceptional time-to-value by having hundreds of pre-built integrations ready today which require little or no coding to implement. Time-to-value is further increased with the DELMIAWORKS API framework being manageable form any browser, anytime.

Enabling Greater Scale, Speed and Integration Simplicity

Integrating on-premise ERP to cloud or Hosted Managed Services can be accomplished all from a browser using the DELMIAWORKS API framework that has an intuitive, easy-to-use interface that enables administrators, analysts and integration experts to customize systems in minutes. Integrating cloud applications and platforms to on-premise and HMS applications can be accomplished without any hardware either, all from The DELMIAWORKS API framework. The DELMIAWORKS API framework flexes to meet the integration challenges your business has and help you overcome the siloed, disconnected systems that stand in the way of your growth.

Support Diverse Integration Patterns to Form a Single Data Source

The DELMIAWORKS API framework is designed to give you the freedom you need to enable diverse integration patterns across shop floors, production centers and distributed offices globally. And the DELMIAWORKS API framework supports all key integration patterns including application-to-application, pub-sub, real-time and event-driven web services, streaming, batch, ETL integrations and more. The DELMIAWORKS API framework supports asynchronous process execution, workflow orchestration and more.

Enterprise Security

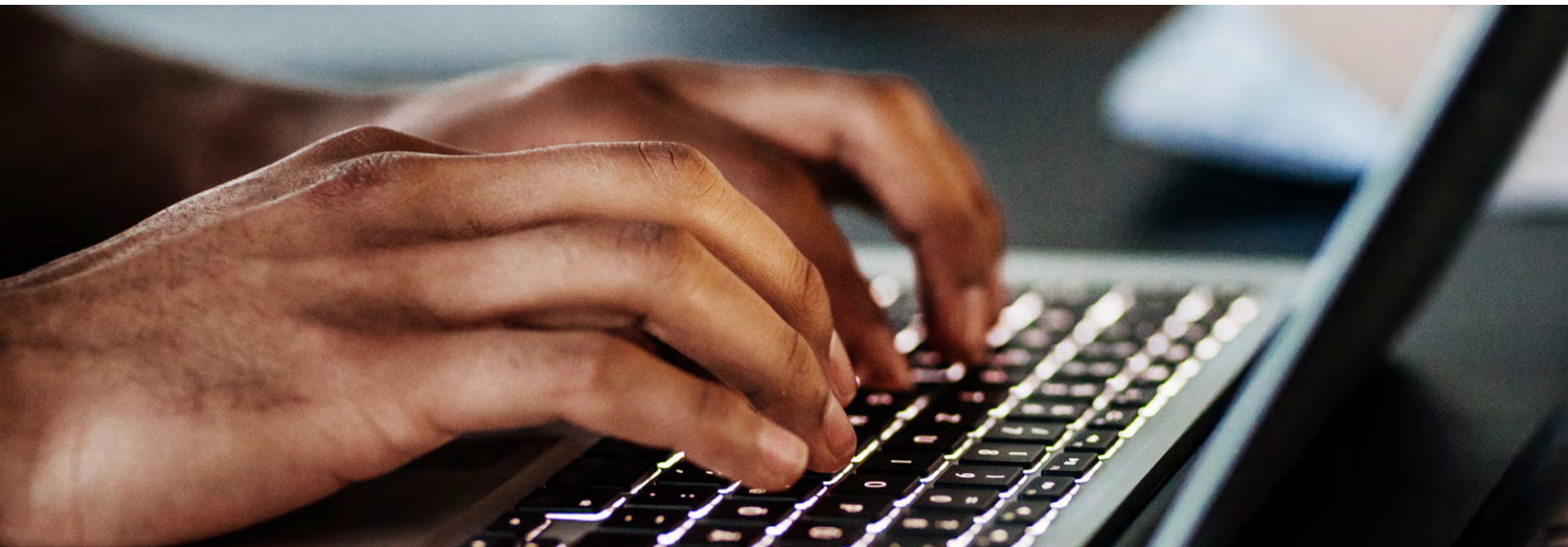
The DELMIAWORKS API framework has multilayered data security designed in, ensuring every threat surface today and in the future is protected from break and hack attempts. DELMIAWORKS realizes identities are the new security perimeter of any business, which is why our data security architecture is built to scale and protect every threat surface of your growing business.

Fast time-to-value with no hardware investment

The DELMIAWORKS API framework enables you to integrate cloud, Hosted Managed Services (HMS) and on-premise applications with no hardware investment. Our centralized management and monitoring tools further increase efficiency and simplify reporting and audit trails. All DELMIAWORKS solutions also support data mapping, error resolution, and all key integration patterns.

Say Goodbye to Swivel-Chair Integration and Hello to the DELMIAWORKS API framework

Every manufacturer has had their share of data capture errors as a result of swivel chair integration or swiveling from one system to another to input production data. Those manual errors and days of frustration are over with the DELMIAWORKS API framework. Our series of integration solutions help your business to run more accurately, efficiently and securely than ever before.



EXPLORING DELMIAWORK'S WEBAPI SOFTWARE DEVELOPMENT KIT

The DELMIAWORKS WebAPI Software Development Kit (SDK) comes with all the tools needed to empower developers to write custom applications with the DELMIAWORKS manufacturing ERP and MES software solution. Most modern programming platforms and languages (Android, ASP.net, Windows, Visual Studio (.Net)) can be used to develop applications.

The WebAPI SDK includes a specific list of interfaces that allow your programmer to encapsulate core DELMIAWORKS logic with the JSON data exchange format for a lightweight exchange of information between the WebAPI and your applications. In addition to the web services/routines, the WebAPI SDK comes with a complete help file system and full code example projects to ensure that the process is a success.

In combination with the WebAPI SDK tools, DELMIAWORKS offers hourly support with its specialized Automation Department for any questions, issues and development surrounding the WebAPI. Additionally, developers can source the valuable contributions of fellow DELMIAWORKS users on the MyDELMIAWORKS online discussion forum.

Key Benefits

- Easily write your own custom apps for DELMIAWORKS using the tools you desire
- Customize the software with solutions that help you address your specific challenges
- Browse the extensive help file system in the HTML Help 1.x format
- Take advantage of the training and development assistance you have come to expect from DELMIAWORKS

What's Included?

The WebAPI Software Development Kit provides these key elements:

- **Example Applications**
  Multiple sample applications come with the WebAPI Software Development Kit. Built with Visual Studio 2010 using the C# .Net programming language, the samples provide fully functional examples of interactivity with the WebAPI.
- **Web Services/Routines**
  A specific list of interfaces is included with the WebAPI Software Development Kit that defines the inputs and outputs that your application must adhere to in order to communicate with DELMIAWORKS.
- **Documentation**
  The WebAPI Software Development Kit comes with a complete help file system in the HTML Help 1.x format, similar to Microsoft's MSDN format for ease of use and application success.
- **WebAPI Training and Development Assistance**
  DELMIAWORKS offers WebAPI training and development assistance two ways: The first is one-on-one personalized collaboration with the DELMIAWORKS Automation Department on an hourly basis. Support includes questions, issues and development surrounding the WebAPI and is billed up front. The second is through the online discussion forum on MyDELMIAWORKS, where customers can source best practice advice and tips from other DELMIAWORKS power users.
- **Versions and Updates**
  If you are working with DELMIAWORKS to develop new applications, you will be notified by email any time an update has been published. Additionally, DELMIAWORKS maintains the compatibility of the WebAPI Software Development Kit by releasing a corresponding version with each release of the DELMIAWORKS software.

GETTING STARTED WITH THE DELMIAWORKS API FRAMEWORK

Requirements

- Oracle Processor based licensing which allows unlimited user connections to the database. The standard licensing provided with EnterpriseIQ is a Named User license which is based on the number of users.
- Internet Information Services (IIS) must be activated on the WebAPI server (**please reference the IIS installation TechNote for more information**)
- Microsoft Framework .Net 4.0 must be installed on the IIS server.
- Oracle 11gR2 is required for the client (**please reference the Oracle installation TechNote for Oracle 11gR2**).
- Installed on the server (or PC) which the Android device(s) will be connecting to (**please reference the IIS installation TechNote for instructions**).

Configuration

There are application settings in the Web.config (located in the root of the WebAPI installation folder) that can be set based on the needs of the developer.

- Debug Mode - Enabling debug mode allows the developer to access the API calls bypassing authentication. A debug user must also be set.
- Data Row Limit - Changing the data row limit allows the developer to set the maximum number of rows to be returned for any one API call. **Default**: 500.

```
< appSettings>
< addkey= “debugUser” value=“iqms”/>
< addkey= “DebugModeEnabled” value=“false” />
< addkey= “SystemHardRowLimit” value=“500” />
</ appSettings>
```

CORS is enabled by default through the Allow-Control-* entries within the Web.config. Referring hosts may be restrcited by setting the ‘Allowed Hosts’ setting on the API settings page.

- **Allowed Hosts**
  Comma-delimited list of hosts permitted to consume the API. The request header must contain a ‘Referer’ url to determine the origin of the request. When ‘Allowed Hosts’ contains entries, requests without a ‘Referer’ or not found in the defined list will be rejected with a 404. If left blank, any origin will be allowed to the API.

Connection Pooling

If you are experiencing connection timeout errors, they may be occurring because you are hitting the maximum number of active memory pool connections.

To view the current number of active memory pool connections, run the following script:

select program, count(*) as active_conn_count from v$session group by program order by active_conn_ count desc;

Doing so will display the number of active memory pool connections – if there is a high number of active connections (100 or close to 100 connections), you may benefit from increasing the number of memory pool connections from the default number.

To increase the number of memory pool connections, do the following:

1. Create a backup copy of the “web.config” file.
2. Wait for off hours to make the change – changing and saving the “web.config” file will bounce sessions.
3. Make the following change to the “connectionString” parameter within the “web.config” file: Change the parameter from:

```
< addname= “OracleInjectedConnection String”
connectionString=“Data Source=IQORA;User
Id={0};Password={1};”/>
```

To:

```
< addname= “OracleInjectedConnection String”
connectionString=“Data Source=IQORA;User
Id={0};Password={1};pooling=true;Max Pool
Size=200;”/>
```

HELPER LIBRARIES

IQMS.Entities.dll

- Class library of business objects that map to logical and/or database entities. These are the objects (parsed t o/from JSON) that the WebAPI consumes and returns.

IQMS.WebAPIClient.dll

- Library of helper methods to perform common actions, such as login, building a request, and mapping a response to a business object.

IQMS Entities

IQMS.Entities.dll

Class library of business objects that map to logical and/or database entities. These are the objects (parsed to/from JSON) that the WebAPI consumes and returns.

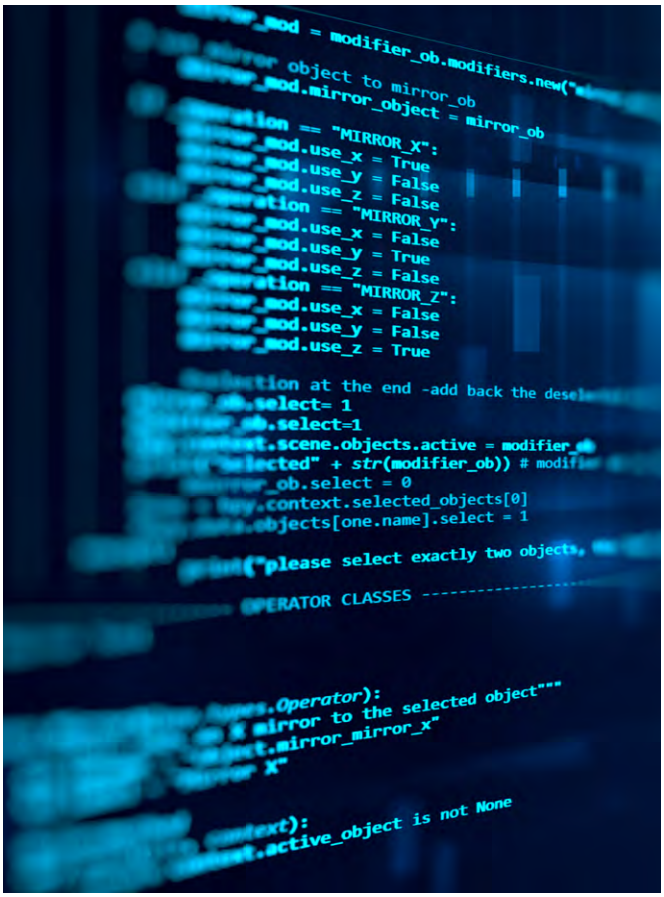
IQMS Service Client

IQMS.WebAPIClient.dll

Library of helper methods to perform common actions, such as login, building a request, and mapping a response to a business object.

Example:

Display a list of Customers, select a Customer from that list, update the city and state for the selected Customer, and post that update.




```
string BASE_URL = “http://myCompanyWebServer.com:8080/WebAPI”; // base URL of the WebAPI
// initialize an IQMSServiceClient object (using IQMS.WebServiceClient.dll)
IQMSServiceClient webServiceClient =
new IQMSServiceClient(BASE_URL, “username”, “password” , “password” , “database”);
// returns true if authentication succeeds
webServiceClient.Authenticate ();
// API URL to get a list of customers
string URL_CUSTOMER_LIST = “/CRM/CustomerCentral/Customers/” ;
List <Customer> customers = webServiceClient.GetItems<Customer>(
webServiceClient.NewRequest(URL_CUSTOMER_LIST).SetHttpMethod(“GET”));
// select a customer from the list (Customer from IQMS.Entities.dll)
Customer customerToUpdate = customers[0];
// update the city for the selected customer
customerToUpdate.City = “Paso Robles”;
// update the state for the selected customer
customerToUpdate.State = “CA”;
// API URL to update a customer
string URL_UPDATE_CUSTOMER = “/CRM/CustomerCentral/UpdateCustomer/”;
// post the update
Customer updatedCustomer =
webServiceClient.GetItem<Customer>(
webServiceClient.NewRequest(URL_UPDATE_CUSTOMER)
.SetHttpMethod(“POST”).SetDataObject(customerToUpdate));
```

AUTHENTICATION

Overview

Authentication for WebAPI is an integral piece of any application leveraging the WebAPI. Authentication is required by default, though may be toggled by enabling debug mode in the Web.config (see Getting Started > Configuration Options). When debug mode is turned off WebAPI will require an authentication token attached to requests' headers. If no token is found or the token is invalid the request will be rejected.

Below are examples of viable approaches to WebAPI authentication.

IQMS Service Client Library

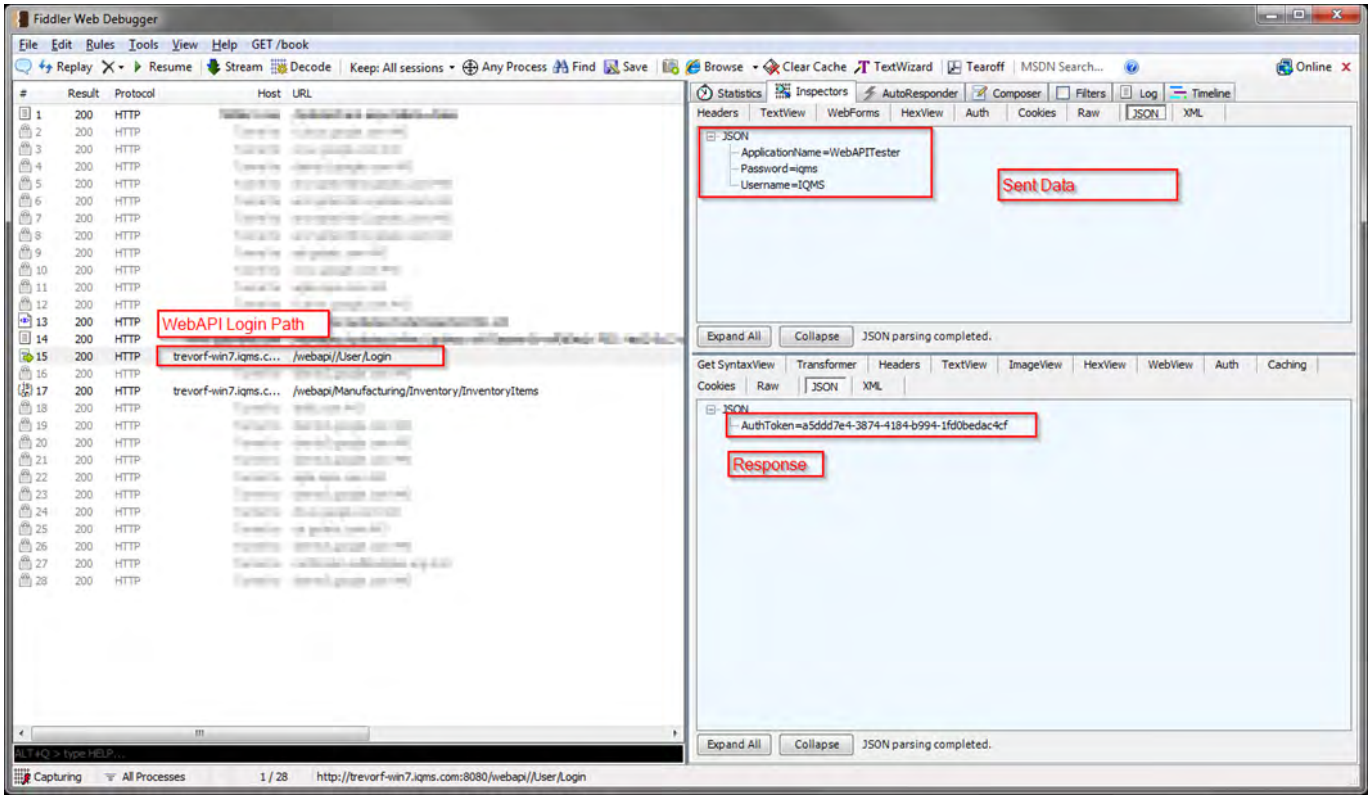
If using the IQMSServiceClient, authentication is performed by passing user credentials, application name, and database alias name in the service instantation. From this point authentication is handled automatically. An example is included below.

```
IQMSServiceClient webServiceClient
= new IQMSServiceClient(BASE_URL, "username", "password" , "myApplication", "database");
webServiceClient.Authenticate ();// returns true if authentication succeeds
```

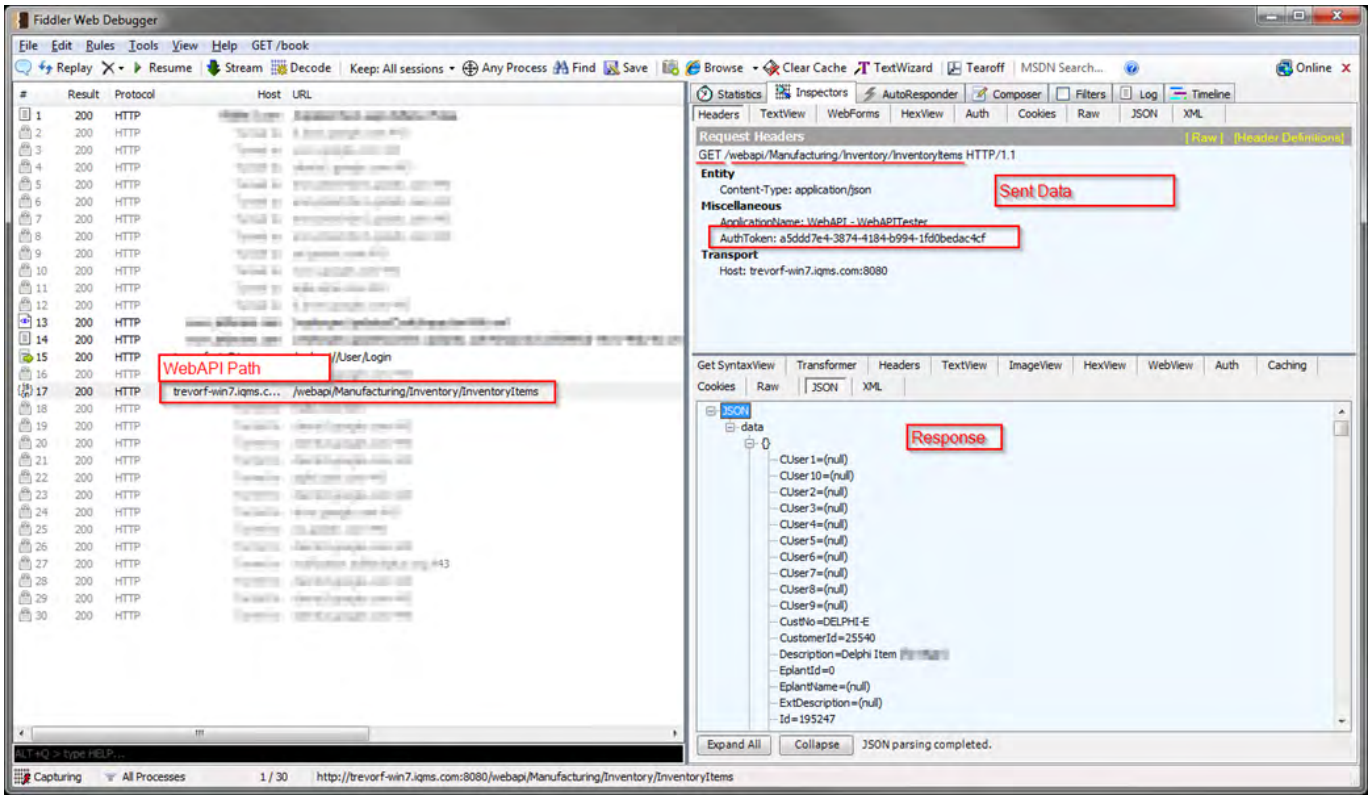
Manual

When not using the IQMSServiceClient library, authentication may be performed manually by providing valid credentials to the route /User/Login as a POST request and attaching the returned token to subsequent request headers. The following is a detailed example of this process using a no-cost Web Debugger tool called Fiddler:

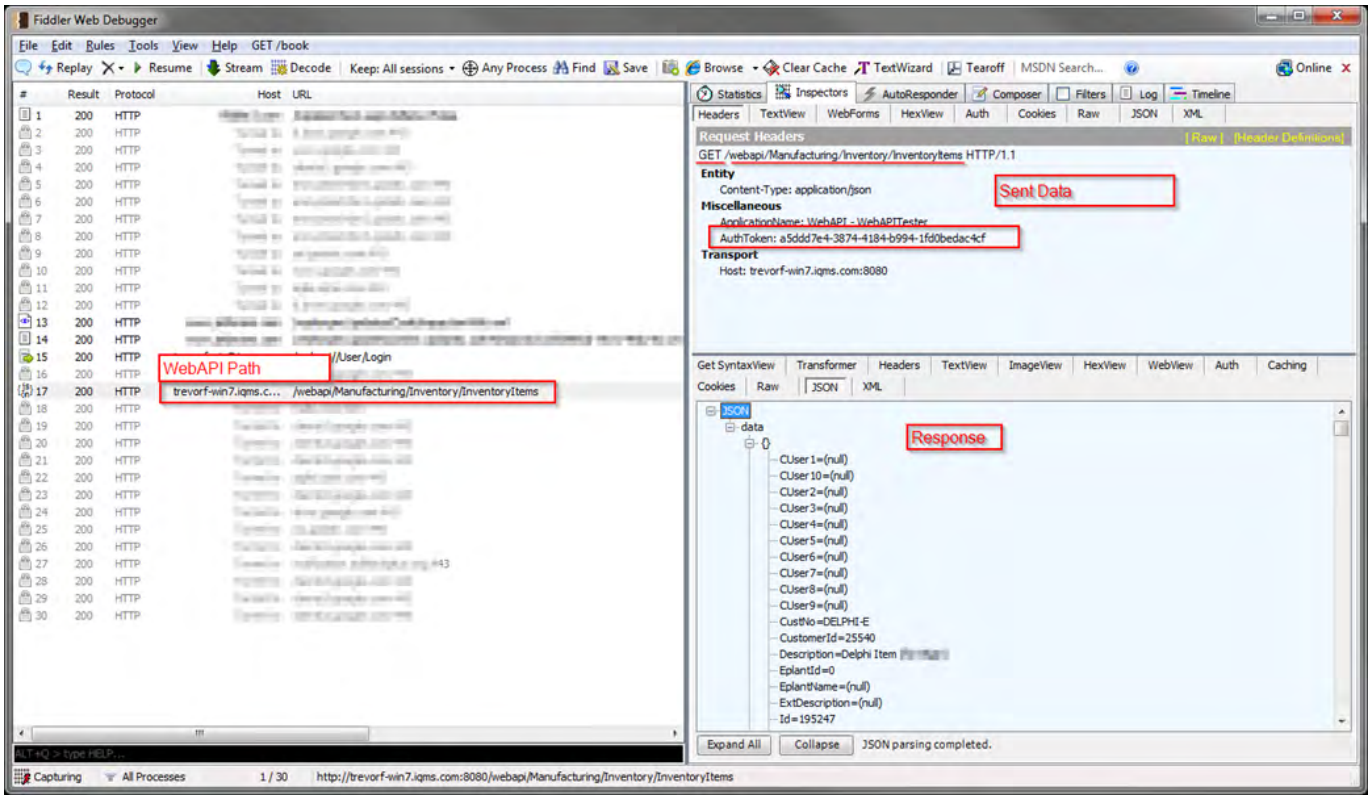
1. Obtain a valid authentication token (AuthToken) by passing the application name, username, and password to the /User/Login action as a POST request. If the login is successful, we receive a response containing the AuthToken that we will use in the subsequent API calls. If authentication fails you will receive a 500 error.



2. This next screenshot shows more information regarding the same login call. you'll notice we have not passed the AuthToken value as we do not have it at this time. AuthToken does not need to be defined; it is only shown for example.



3. Now that we have an AuthToken value, we can access the various WebAPI methods by passing that value in the header of the API call. If the authorization token is missing or invalid requests will receive a 403 Forbidden error.



CONCLUSION

The DELMIAWORKS API framework is comprised of 23 unique API classes. These include Accounting, AccountsPayable, AccountsReceivable, AssemblyData, AssetManagement, Country, CRM, Estimating, ExpenseReports, ExternalPartners, Inventory, Labels, Manufacturing, MRO, Payroll, PendingApprovals, POReceiving, Quality, SalesDistribution, TimeAttendance, UserDefined, WMS and Workforce. Taken together the DELMIAWORKS API framework is designed to support a single development vision for implementing an ERP, MES and manufacturing system that can scale across an entire organization. The DELMIAWORKS API framework supports having transactions updated instantly throughout the entire system, in real-time. This results in a lower overall cost of investment, lower maintenance costs and quicker Return on Investment (ROI).

For more information, please visit www.iqms.com or call 1.866.367.3772