Ideagen

Q-Pulse Web Service APIs Developer Guide

Contents

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Introduction

Welcome to Q-Pulse Web Service APIs.

The web service APIs provides key Q-Pulse functionality in the form of XML or JSON based webservices.

Web service APIs provide an industry standards-based, cross platform, flexible mechanism for integrating Q-Pulse functionality into third-party applications and business processes.

They work by exposing "endpoints" on the network that receive messages destined for Q-Pulse and sending response messages from Q-Pulse in return.

Messages are sent using the standard formats and protocols used to build the World Wide Web, so provide great opportunities to build applications on any platform that supports these standards.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Q-Pulse Web Service API Extension Pack Contents

The API extension pack comprises the following items:

Web Service APIs Installer

Installs the Service APIs to an IIS virtual directory.

Q-Pulse Web Service API Developer Guide

This document.

Q-Pulse Web Service API Technical Reference Guide

Contains technical documentation for using the service APIs.

"Live" Documentation

Installed alongside the Services enabling direct access to WSDL/Schema files, Operation and Data Contracts used by the Q-Pulse Web Service APIs.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Concepts and Considerations

Required Skills

Developing applications to use Q-Pulse APIs requires an individual to have a solid understanding of:

- The relevant modules in Q-Pulse.
- The concepts involved in Web Service APIs:

  Message Exchange Patterns - http://en.wikipedia.org/wiki/Message_Exchange_Pattern

  XML (Extensible Markup Language) - http://en.wikipedia.org/wiki/XML

  HTTP (Hypertext Transfer Protocol) - http://en.wikipedia.org/wiki/HTTP

  SOAP (Simple Access Object Protocol) - http://en.wikipedia.org/wiki/Simple_Object_Access_Protocol

  REST - http://en.wikipedia.org/wiki/Representational_State_Transfer

  JSON – http://en.wikipedia.org/wiki/JSON

- An appropriate platform, toolkit or IDE for using web service APIs. i.e.

  Microsoft Visual Studio:

  Eclipse

  ColdFusion

  WCF

  Apache

  Axis

  JWSDP (Java Web Services Developer Pack) from Sun

  WebMethods

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Web Service API Concepts

Multiple Endpoint Interfaces

Q-Pulse Web Service APIs support SOAP, REST and JSON programming styles.

Your choice of which style to use is entirely up to you, however some platforms make working with a particular style easier.

For example PHP, Ruby and Python developers may prefer the REST style of programming. However, .NET and Java developers may prefer working with SOAP interfaces as the tools for these platforms provide simple point and click integration of web service APIs in their respective IDEs.

Contracts

The two ends of the wire involved in a call to a service ("service" and "gateway") communicate with each other using pre-agreed "Contracts" which describe the format of the data being sent and the operations that a service can perform using that data. These can be viewed in similar way to remote procedure calls and consist of a request and response message pair.

Messages agreeing to a specific "DataContract" can be sent to endpoint that provides an "OperationContract" that supports the given "DataContract".

Q-Pulse uses 3 types of Contracts to describe the messages passed around.

- **Data Contracts** – Sometimes referred to as "DataTypes" or "Schema Contracts" describe in XSD (XML Schema Definition) format, the shape of the data to be exchanged.
- **Service / Operation Contracts** – Sometimes referred to as "Action" Contracts, they describe in WSDL (Web Service Description Language) format the operations that the service endpoint can perform.
- **Fault Contracts** – if any of the services encounter a problem, they will return a predefined Fault Contract explaining the nature of the problem.

The technical documentation provides information on these contracts and sample messages sent to services.

NOTE: Some items within the schema for the service contracts contain annotations, for example:

```
<xs:annotation>
  <xs:appinfo>
        <Default Value EmitDefaultlue="false"
      xmlns="http://schemas.microsoft.com/2003/10/Serialization/"
      />
  </xs:appinfo>
</xs:annotation>
```


Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

This has been used as an optimization mechanism. Whilst not required to, endpoints can use this annotation to identify XML elements that may not be required to be sent over the wire. Microsoft's WCF for example uses this optimization automatically when generating a service proxy, but it is not a platform specific technique.

Versioning and Product Dependency Considerations

One of the aims of the Q-Pulse Web Service API's is to make them available from any platform that supports Web Services. As a result, all the types exposed in the Q-Pulse Data Service contracts are XSD compatible types. There are no platform specific types exposed (such as AutoSerialized .NET complex types) that may limit platform availability.

Contracts are versioned as a group, so all the web service contracts have the same version specified at any particular release. Clients can validate the current version of the supported schema at runtime by, for example, hard coding a supported version in their application. Alternatively, messages may be posted to the service without validation.

As the versioned schema is only used to construct messages (and not transported with the message itself) any attributes set as either Null/Nil or empty strings are ignored. This provides a degree of forward compatibility as APIs can be added to without breaking a previous message contract.

The actual implementation of the API's must match the current version of business objects used within Q-Pulse. As a result, with each new release of Q-Pulse, you will be required to install the latest version of the API's. However, this does not necessarily mean that the contracts published by the API's will change also. These may remain the same providing a degree of decoupling between clients and Q-Pulse itself.

As contracts change over time, we will endeavour to provide backwards compatibility wherever possible, and where we cannot, to attempt to identify any breaking changes in the release notes.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Getting Started

Deploying Q-Pulse Web Service API's

To deploy Q-Pulse Web Service API's you can run the installer and follow the on-screen instructions.

Alternatively, if you wish to install the Web Service API's manually:

It is relatively simple to set up as it runs as a standard ASP.NET/IIS Application.

1. You must have IIS 6.0 or 7.0 and ASP.NET 4.0 or above (include .NET Framework 2.0 and .NET Framework 3.5) installed. For details on how to install and configure IIS 7.0 on your chosen operating system please refer to the Microsoft documentation located at http://technet.microsoft.com/en-us/library/cc753433.aspx
2. You should have a working Q-Pulse setup. (See the Q-Pulse install manual for details).
3. Unzip the web services folder into a directory under your IIS directory (Normally C:\inetpub\wwwroot\)
4. Mark the selected folder as an "Application" in the IIS Management Console
5. You will need to configure the Web Service APIs using the tags within the web.config file.

Within the <appSettings> section, the following fields should be edited to match your existing Q-Pulse installation. The current settings may be viewed in the web.config file located in your Q-Pulse web installation directory.

```
<add key="Authentication" value="CSLA"/>

<add key="Provider" value="BacchusSql"/>

<add key="DBType" value="Sql"/>

<add key="Location" value="Server"/>

<add key="UseService" value="true"/>

<add key="ServiceEndpoint" value="<Q-Pulse Server IP Address>:747"/>

<add key="MessageTriggersEnabled" value="false"/>

<add key="MessageServiceEndPoint" value="<Q-Pulse Server IP
Address>:747"/>

<add key="UseMessageService" value="true"/>

<!--Used by attachment management-->

<add key="ShowDirectAccessWarning" value="false" />

<add key="WindowsClientMetrics" value="true" />

<add key="SystemUser" value="<Q-Pulse Username>"/>

<add key="SystemPW" value="<Q-Pulse Passowrd>"/>
```

In addition to the aforementioned keys, you can find a "PublicTokenKey" within the file.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

You should change this to be an 8-character (64 bit) string. This key is used to validate user session tokens. If you suspect that a token has been misappropriated, you can invalidate previously issued keys by changing this value.

NB: If you are running the APIs in a load-balanced environment you should set this key to be the same on all machines within the cluster.

Security Considerations

Authentication and Licensing

You must have the appropriate licensed features enabled to use certain services within the Q-Pulse Web Service API's. If you do not have an appropriate license invalid responses will be returned.

The Q-Pulse Web Service API's themselves use 4 types of Authentication scheme to control access to their exposed functionality.

Anonymous

The ICore Available Databases service requires no user credentials. You can request the available Q-Pulse Database list without providing any credentials.

Q-Pulse Authentication

In this case, the Application passes users Q-Pulse credentials to the service to generate an authentication token for use with the service APIs.

Windows Authentication

Applications can be built to use an end-user's Windows Identity to perform login, so an end user does not need to enter Q-Pulse Authentication credentials. In this case, Q-Pulse matches a given Windows identity to a Q-Pulse Identity and uses the credentials of the delegated Q-Pulse identity to decide access rights.

Windows Prompted Authentication

Applications can be built to prompt to use an end-user's Windows Identity and Password to perform login, instead of a Q-Pulse Authentication credentials. Inthis case, Q-Pulse matches a given Windows identity to a Q-Pulse Identity and uses the credentials of the delegated Q-Pulse identity to decide access rights.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

To enable Windows or Windows Prompted authentication for the service APIs you need to configure this functionality within Q-Pulse.

In Q-Pulse follow the following steps:

1. Log in as a user with access to the Administration module.
2. Open the Administration Module
3. In the Admin Console go to "Security" > "Defaults and Settings"
4. In the" Authentication" section select "Edit..." This opens the "Setting and Defaults" window.
5. If not selected, select the "Authentication" tab
6. Select "Use Windows Authentication (automatic login)" or "Use Windows Authentication (prompt for credentials)" and click OK.
7. If required, you can use the "Import" function under the "Security – People" section to import your users from exchange or Active Directory.

IMPORTANT NOTE: The authentication tokens used by Q-Pulse Web Service APIs are generated using the "PublicTokenKey" entry in the web service APIs web.config file. If you change the value of this Key. Previously generated authentication tokens will be invalidated.

Digital Signatures

The Acknowledge Document API method supports the use of Digital Signatures. A user can pass their username and password as parameters to use for digital signing. The username and password correlates to the Authentication method enabled. For Q-Pulse Authentication the user must pass their Q-Pulse username and password. For Windows authentication the user must pass their linked windows username and password.

To enable Digital Signatures for the service APIs you need to configure this functionality within Q-Pulse.

In Q-Pulse follow the following steps:

1. Log in as a user with access to the Administration module.
2. Open the Administration Module
3. In the Admin Console go to "Security" > "Signature Management"
4. Select "Document Acknowledge" from list and click Edit.
5. Enable 'Signature Point is active' option and click OK.

Hosting Environment (IIS Security Considerations)

Q-Pulse Web Service APIs are hosted in IIS, and you should take some precautions to secure your installation.

XML and JSON are a plain text format so messages can be read as they travel between machines. You should secure access to your service APIs using industry standard SSL (Secure Sockets Layer) certificates.

You should also restrict access to the Services Server to only those machines that require access to it.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

This can be configured in IIS 6.0 using the "Directory Security" tab on the Application and in the configuration file in IIS 7.0 (see http://learn.iis.net/page.aspx/110/changes-between-iis6-and-iis7-security/ for details).

```
<system.webServer>
        <security>
                <ipSecurity allowUnlisted="false">
                    <add ipAddress="127.0.0.1" allowed="true" />
                </ipSecurity>
        </security>
</system.webServer>
```

Microsoft's documentation on Securing IIS is available here. https://docs.microsoft.com/en-us/previous-versions/iis/6.0-sdk/ms525331(v=vs.90)

General Application Design Considerations

You should NEVER save any authentication information or security tokens to an insecure disk location. You should keep any user credentials and keys secure. If you must save the details, you should store these in a secure location with minimum possible required access rights.

You should always carry out a full security audit / risk assessment as part of your application development process.

If you are developing a distributed application on top of the Web Service APIs, any client clocks must be synchronized with the main Q-Pulse Application server as whilst Q-Pulse Web Service APIs support UTC format, they do not currently support live Time Zone conversions.

Ideagen Plc. T: +44 1629 699 100 e: info@ideagen.com w: www.ideagen.com

Ideagen

Mere Way

Ruddington Fields Business Park

Ruddington

Nottinghamshire

NG11 6JS

UK

e: info@ideagen.com

w: www.ideagen.com