Confirm®

Document Management Web Application Interfacing Specification

v19.20c.AM
Open Source Attribution Notice

The Confirm suite of products contain the following open source software:

• Feature Data Objects v 3.5.0, which is licensed under GNU Lesser General Public License, Version 2.1, February 1999 with the unRAR restriction. The license can be downloaded from: http://fdo.osgeo.org/licenceAndGovernance.html. The source code for this software is available from http://fdo.osgeo.org/content/fdo-350-downloads

• MrSID software (specifically the mrsid32.dll) is used under license and is Copyright © 1995-2002, LizardTech, Inc., 1008 Western Ave., Suite 200, Seattle, WA 98104. All rights reserved. MrSID is protected by U.S. Patent No. 5,710,835. Foreign patents are pending. Unauthorized use or duplication prohibited.

Patented technology in the Software was developed in part through a project at the Los Alamos National Laboratory, funded by the U.S. Government and managed by the University of California. The U.S. Government has reserved rights in the technology, including a non-exclusive, nontransferable, irrevocable, paid-up license to practice or have practiced throughout the world, for or on behalf of the United States, inventions covered by the patent, and has other rights under 35 U.S.C. § 200-212 and applicable implementing regulations.

For further information, contact Lizardtech.

• NodaTime, version number 1.3.10, which is licensed under the Apache license, version number 2.0. The license can be downloaded from http://www.apache.org/licenses/LICENSE-2.0 . The source code for this software is available from http://nodatime.org/.

• Chromium Embedded Framework, version 3, which is licensed under the New BSD License. The license can be downloaded from http://opensource.org/licenses/BSD-3-Clause. The source code for this software is available from http://code.google.com/p/chromiumembedded/downloads/list.

• Xilium.CefGlue, version 3, which is licensed under the MIT License (with portions licensed under the New BSD License). The licenses can be downloaded from http://opensource.org/licenses/MIT and http://opensource.org/licenses/BSD-3-Clause. The source code for this software is available from http://xilium.bitbucket.org/cefglue/.

• D3 Data Driven Documentation, version 3.4.1, which is licensed under the New BSD License. The license can be downloaded from https://github.com/mbostock/d3/blob/master/LICENSE. The source code for this software is available from http://d3js.org/.

• OpenLayers, version 2.12, which is licensed under the Modified BSD License. The license can be downloaded from http://svn.openlayers.org/trunk/openlayers/license.txt. The source code for this software is available from http://trac.osgeo.org/openlayers/browser.

• OpenLayers, version 3, which is licensed under the BSD 2-Clause License. The license which can be downloaded from https://github.com/openlayers/ol3/blob/master/LICENSE.md. The source code for this software is available from https://github.com/openlayers/ol3.
• Proj4js, version 1+, which is licensed under the Apache License, Version 2, January 2004. The license can be downloaded from http://www.apache.org/licenses/LICENSE-2.0.html. The source code for this software is available from http://trac.osgeo.org/proj4js/.

• requireJS, version 2.1.2, which is licensed under the MIT License or the New BSD License. The license can be downloaded from https://github.com/jrburke/requirejs/blob/master/LICENSE. The source code for this software is available from http://requirejs.org/.

• Apache Cordova, version 8.1.2, which is licensed under the Apache License, Version 2, January 2004. The license can be downloaded from http://www.apache.org/licenses/LICENSE-2.0.html. The source code for this software is available from http://phonegap.com/download/.

July 16, 2019
Table of Contents

Specifications

Document Management Web Application Interfacing
Specification 6
GENERAL CONSIDERATIONS 6
Document Management Web Application Interface 6
Specifications

The following sections outline all the Specifications that exist within the Confirm functionality.

In this section

Document Management Web Application Interfacing Specification 6
Document Management Web Application Interfacing Specification

Introduction
This document describes the interaction between Confirm and Web Applications for document management. In this scenario the Document, and its related details, are sent from Confirm to an external Web Application, which in turn should reply with the appropriate Response.

The design is based on REST architecture, and both the request and response that Confirm expects the Document Management Web Application to follow are detailed. Implementation of the web application functionality itself has no specific requirements and the Client is free to choose how they achieve this.

GENERAL CONSIDERATIONS

Security
Access to the web application is achieved using the network credentials of the user currently logged onto the client PC. The document data is sent from Confirm to a web application that is totally independent of the Confirm application, so full Confirm credentials are not required. However, the Confirm Username and Database ID are sent as part of the request for information purposes.

If a proxy server is in use then settings for this are entered in the General System Settings Screen. These settings are shared so if other parts of Confirm are using a proxy server then the Document management system will also attempt to use it.

Extensibility
The purpose of using a web application to deal with the storage and management of documents linked to Confirm is to allow the clients to use any document management system without requiring changes to the Confirm application that can take time to develop and require extensive testing.

By sending the document and associated details to a web application in a predetermined format it becomes possible for clients to write their own web application, capable of storing the document in whatever document management system they choose, with no impact on the Confirm application itself provided the web application adheres to the interface requirements specified in this document.

Document Management Web Application Interface

General
The details for the HTTP protocol used for communication between the client and server is taken from W3C (www.w3c.org) - Hypertext Transfer Protocol -- HTTP/1.1 - http://www.w3.org/Protocols/rfc2616/rfc2616.html.

The Information related to the document being sent is passed using the HTTP requests headers. As such all information is passed in string format but where a header value represents a non string data type the specifications defined by w3c.org for simple XML schema data types are used. The definitions are given by W3C - XML Schema Part 2: Datatypes. - http://www.w3.org/TR/xmlschema-2/
Specifications

The Response Status Codes detailed here are based on the specifications provided by W3C – status code definitions (http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html).

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Example</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>CCYY-MM-DD</td>
<td>2002-01-31</td>
<td></td>
</tr>
<tr>
<td>dateTime</td>
<td>CCYY-MM-DDThh:mm:ss</td>
<td>2002-01-31T14:45:34</td>
<td></td>
</tr>
<tr>
<td>boolean</td>
<td>true, false, 1, 0</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>decimal</td>
<td>p,s</td>
<td>a 6,2 decimal could be up to 9999.99</td>
<td>Positive and negative numbers with decimal point if appropriate. Size shown in the form p,s where p = precision and s = scale</td>
</tr>
<tr>
<td>string</td>
<td>character value</td>
<td>High Street</td>
<td>Max size shows maximum length</td>
</tr>
</tbody>
</table>

HTTP Request

The HTTP request contains all of the required information for uploading a document, including the document data itself. It is initiated from within Confirm so there will be little or no opportunity to change the details. There will, however, be a setting in Confirm to allow the send method to be specified (PUT or POST). Any web application written to receive these requests should be aware of certain header values that are provided.

<table>
<thead>
<tr>
<th>Header</th>
<th>Value</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-Control</td>
<td>no-cache</td>
<td>Confirm requires that all responses are current.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>application/octet-stream</td>
<td>This is the MIME type being sent as part of the PUT/POST method. The file may be any type so this is used to apply to all types.</td>
</tr>
<tr>
<td>Proxy-Authorization</td>
<td></td>
<td>This will only have details if they have been entered into Confirm General System Settings.</td>
</tr>
</tbody>
</table>

In addition to the standard headers there are additional custom headers sent containing the document related details. These details are always provided by Confirm but can be ignored by the web application if they are not needed.

<table>
<thead>
<tr>
<th>Header</th>
<th>Max Size</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-confirm-FileName</td>
<td>50</td>
<td>This was the name of the file being sent.</td>
</tr>
<tr>
<td>x-confirm-Date</td>
<td>20</td>
<td>The Document date time value from the document link record.</td>
</tr>
<tr>
<td>x-confirm-Description</td>
<td>100</td>
<td>The Description from the document link record.</td>
</tr>
<tr>
<td>x-confirm-EntityType</td>
<td>10</td>
<td>The type of Confirm record the document link is actually related to.</td>
</tr>
<tr>
<td>x-confirm-EntityKey</td>
<td>40</td>
<td>Information used to identify the record that the link relates to.</td>
</tr>
</tbody>
</table>
An alternative way of sending the document information is to include it as part of a query string on the URL in the Document Management System Settings. The query string is added onto the 'Web Service URL' using whatever syntax and field names are required by the receiving web application. The values that should be inserted into the query are specified by using tags consisting of the header names (minus the “x-confirm-” part) specified in the table above, but enclosed in $ symbols e.g. $EntityType$, $DatabaseId$. An example of a URL passing values in this way is:

http://company.com/DocumentManagement/?DocumentName=$FileName$&CreatedDate=$Date$

In the example the query fields 'DocumentName' and 'CreatedDate' are examples of query argument names that the receiving web application would be configured to receive and understand, and '$FileName$' and '$Date$' would be replaced by actual values from the linked document being uploaded. If a document called “TestResults.doc” that had a document date (taken from the document link record) of 2-March-2009 then the URL above would be sent to the web server as:

http://company.com/DocumentManagement/?DocumentName=TestResults.doc&CreatedDate=2009-03-02

Note: Sending a document over HTTP means that the request can be very large. This must be catered for when setting up the receiving web application, as they will probably have maximum transfer sizes and timeout values that prevent large files from being sent.

HTTP Response

The HTTP response has fewer details to return to the client, but relevant header details are detailed here.

If the web application successfully adds the document to the Document Management System then the URL to the newly created resource should be returned in the ‘Location’ header and the response should have a status code of 201 (Created), although all codes in the 300 range (redirect codes) and
all others in the 200 range (success) will be accepted by Confirm. For codes in the 200 range that may not have a Location header value the URL that the document was sent to will be used as the URL to the resource so it is important that this is correct if using a Document Management system that does not redirect after creating a new resource.

The redirect location returned will not be followed by Confirm but simply used to indicate the location of the newly created resource. For this reason it is important that access to any Document Management Web Applications from Confirm is direct and there are no other redirections along the way as Confirm will treat the redirect address as the document URL.

**Error Handling**

Exceptions that occur in the web application must be communicated back in a platform-independent manner. To accomplish this, exceptions raised from the Web application should be returned using the HTTP response status information.

Custom status codes will be used to convey the nature of the problem based on the W3C status code definitions (http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html). This defines status codes beginning with a 4 as being client errors and those beginning with a 5 as server errors.

The status codes specified here should be used to allow Confirm to display the error message to the user correctly, but the status description can be changed to describe the true nature of the problem. This description will be displayed in a popup error message window within Confirm when an error is detected so should be formatted for this.

When an error is returned the message “Failed to upload - the following error was returned by the Web Service:”, followed by the error message that is returned in the status description.

Other types of error will be detected and displayed in the same popup error message window but their descriptions may have less meaning to a user than a custom message.

**Note:** Other types of error will be detected and displayed in the same popup error message window but their descriptions may have less meaning to a user than a custom message.

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>472</td>
<td>Error indicating a problem initiated from the client side i.e. invalid character in the filename header or some required information is missing from the request.</td>
</tr>
<tr>
<td>572</td>
<td>Error initiated from the server side i.e. unable to connect to external document management system.</td>
</tr>
</tbody>
</table>