Formworks v4
Web Service Guide

Contents

The Formworks Web Service

Introduction

The Fiddler application has been employed throughout this guide to demonstrate retrieving and posting data via the Formworks web service, in json and XML formats. In a production environment, bespoke software would probably be employed instead, but Fiddler easily demonstrates the requirements of the Formworks web service, and can be used to pre-populate forms.

Retrieving data from the web service

GET Command

To retrieve basic data and URL's for the web service, select the Composer tab, and enter the following URL into the Fiddler GET command line:

https://www.formworks.uk.com/api/v1/import.json

The .json extension can be changed to .xml to retrieve the web service's exported data in XML format. If you wish to retrieve the output in XML format, you do not need to change the .json value in the 'Request Headers' section to XML.

Request Headers

The Authorization: Bearer line should be set to the client specific value, which is located in the Formworks portal, on the Admin, Licences tab, following the "Your Web API Secret:" label. See Figure 1.

User-Agent: Fiddler
Host: dev.compsoft.co.uk
Authorization: Bearer f5457131-4b90-4077-98ac-6501f82dc815Accept: application/json

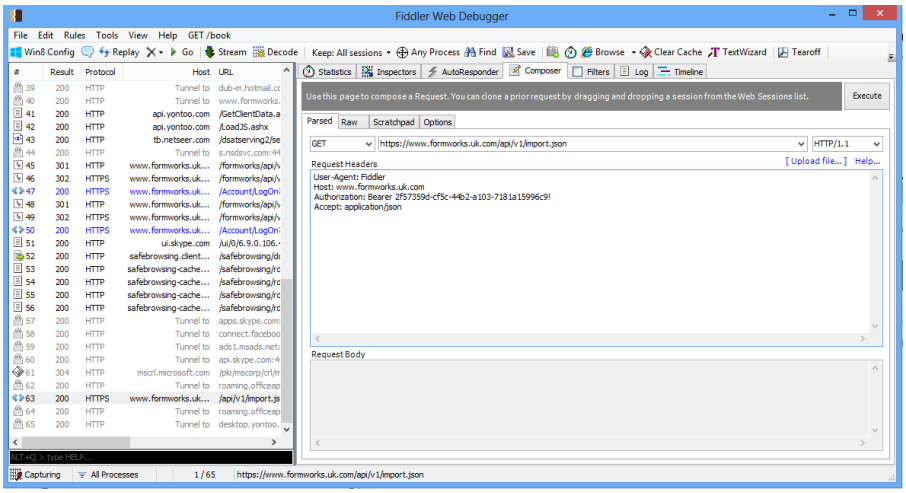


Figure 1: Fiddler Composer tab

To view the returned data, use the Inspectors tab, and select either the Raw, JSON or XML formats. XML data will only be available if you have specified an XML extension in the GET command line. See Figure 2.

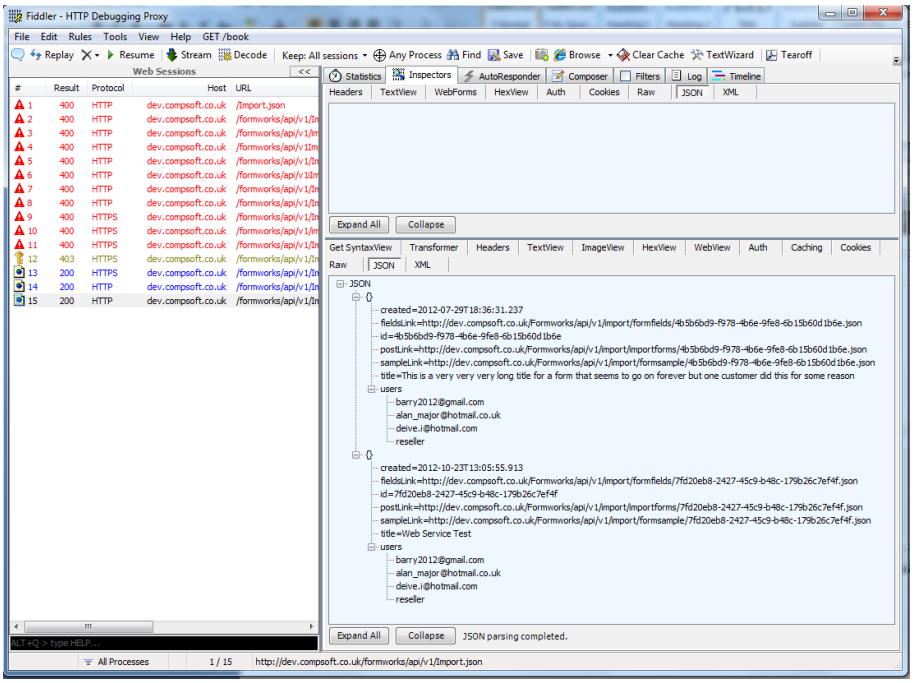


Figure 2: Inspectors JSON view

The exported data includes three links that can be used to both examine, and pre-populate the data fields contained in any available digital forms.

The fieldsLink URL provides full details of each field within the form, including its label, name, data type and sample data the field can contain. The fieldsLink URL is covered in detail in the fieldsLink chapter.

The postLink URL provides the option of pre-populating the data fields of a form within Formworks, and making it available for a specific Formworks iPad user. The postLink URL is covered in the postLink chapter.

The sampleLink URL provides a list of field names and samples of the data that the fields can contain. In addition, this URL is useful when posting pre-pop data to forms. The sampleLink URL is covered in detail in the sampleLink chapter.

fieldsLink

To retrieve full field data for the fields within a form, locate the form on the Inspectors screen, and copy the fieldsLink URL to the Clipboard. Return to the Composer screen and paste the link into the GET command line. Be careful to remove the characters, 'fieldsLink=', before Executing the command.See Figure 3.

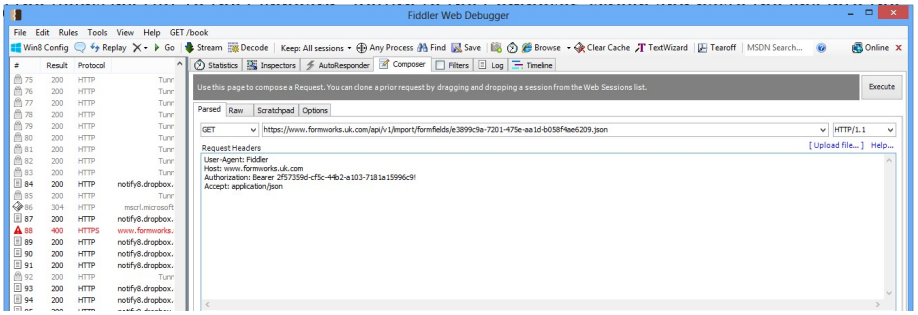


Figure 3: fieldsLink ULR

After executing the fieldsLink command, go to the Inspectors view. Depending on which extension (JSON or XML) was specified in the Fiddler Command line, select either the JSON or XML options inthe Inspectors view. See Figure 4.

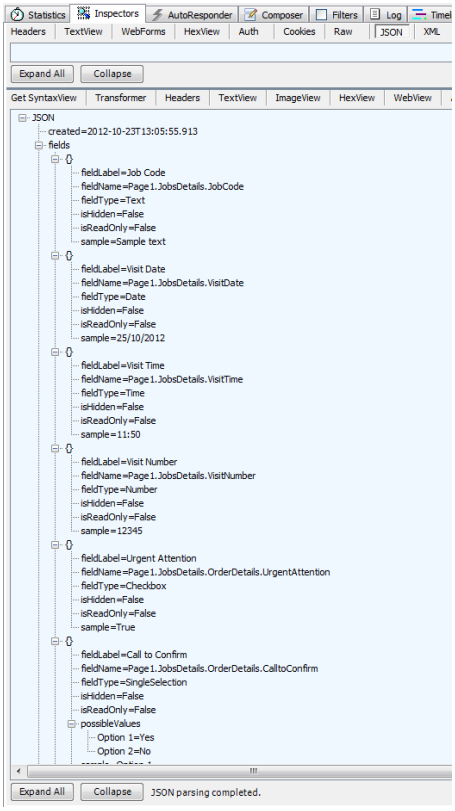


Figure 4: Inspectors view - fieldsLink URL option

sampleLink

To retrieve sample data for the fields within a form, locate the form on the Inspectors screen, and copy the sampleLink URL to the Clipboard.

Example:
http://dev.compsoft.co.uk/Formworks/api/v1/import/formsample/7fd20eb8-2427-45c9-b48c-179b26c7ef4f.json

Return to the Composer screen and paste the link into the GET command line. Be careful to remove the characters, 'sampleLink=', before Executing the command. The .json extension can be substituted with
.XML to extract data in XML format. See Figure 5.

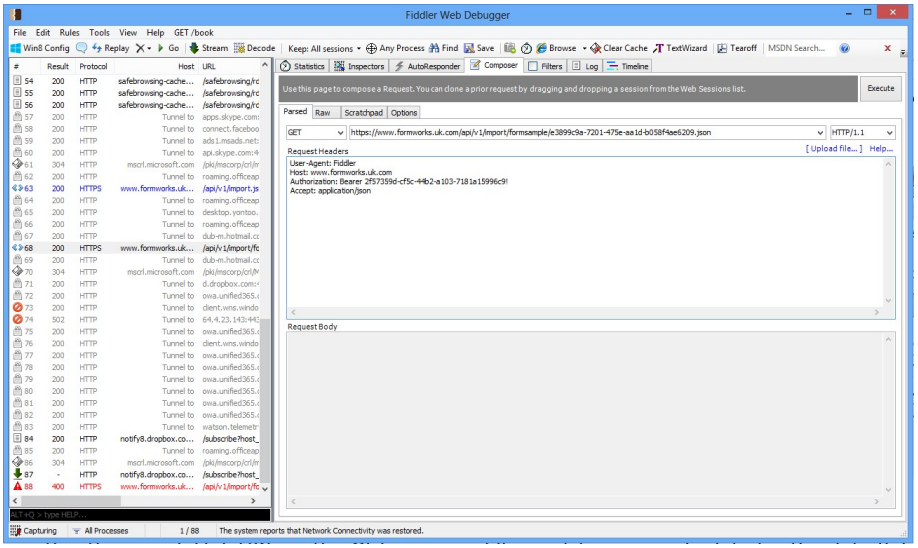


Figure 5: Retrieving form field sample data

Executing the sampleLink URL on the GET command line, retrieves sample data for the data fields in the second form, which is displayed in the Inspectors screen, in this example, on the JSON tab. The data includes the forms field names, and appropriate values for those fields. See Figure 6.

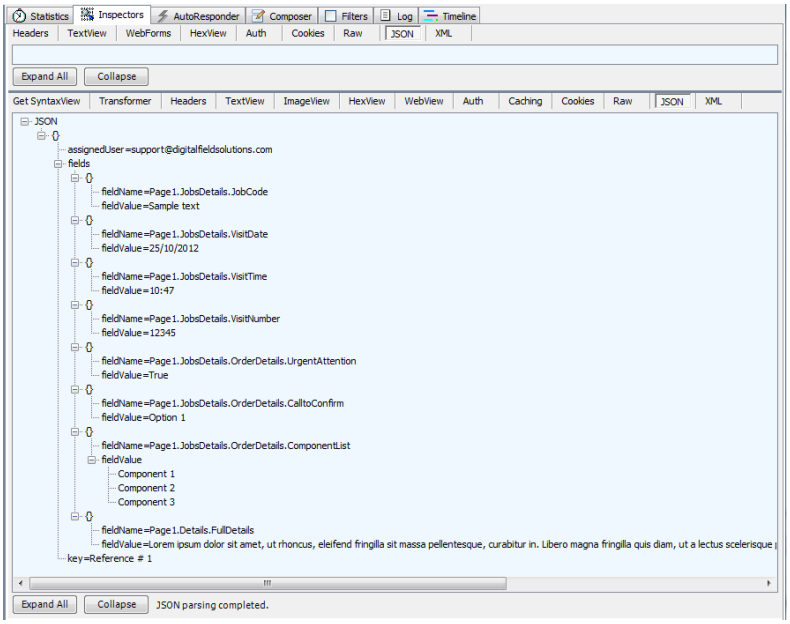


Figure 6: Inspectors screen – sampleLink URL

Raw tab

After executing the sample link, selecting Inspectors tab, followed by the Raw option, will display output similar to figure 7. This sample data can be modified and used to test the pre-populate process,as described in the postLink chapter.

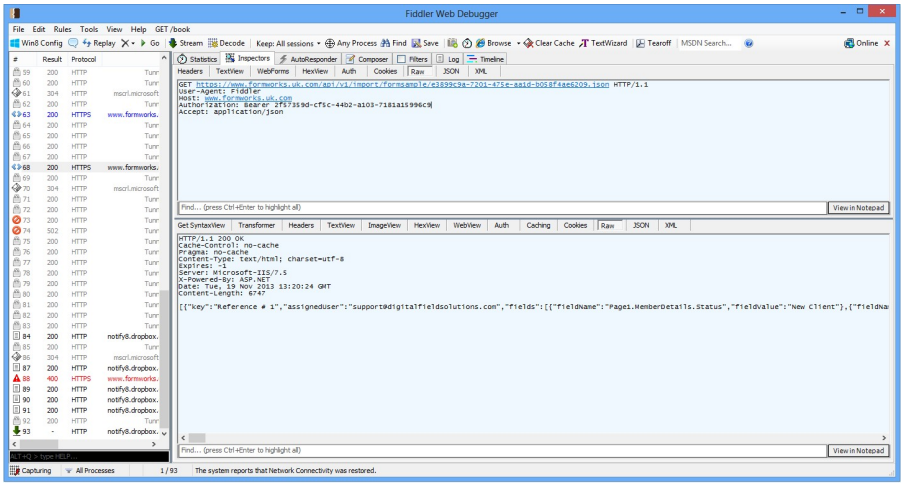


Figure 7: Raw tab - sample data

Posting data to the web service

postLink

The postLink URL is used to upload pre-pop data to forms. To pre-populate data fields, copy the postLink URL from the Inspectors screen to the Clipboard. On the Composer screen, change the command type option to POST. Paste the postLink URL into the command line. Be careful to remove the characters, 'postLink='.

postLink example:
http://dev.compsoft.co.uk/Formworks/api/v1/import/importforms/7fd20eb8-2427-45c9-b48c-179b26c7ef4f.json

Request Headers

The only change to the Request Headers entries is to substitute the Accept: application/json instruction with the instruction: content-type: application/json as below, or: content-type: text/xml if XML data is being used topre-populate a form.:

User-Agent: Fiddler
Host: dev.compsoft.co.uk
Authorization: Bearer f5457131-4b90-4077-98ac-6501f82dc815content-type: application/json
Content-Length: 683

Request Body

The json format sample data captured in the previous chapter can be pasted into the Request Body section, and the sample values amended to reflect valid input. Alternatively, you can create a text file of the sample data, and amend this. To create a text file, select File, Save, Response, Entire Reponse. See Figure 8. A complete example using json is included below, and Figure 9 shows how Fiddler should appear at this stage.

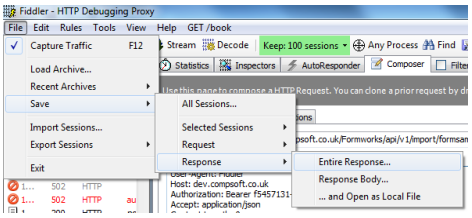


Figure 8: Saving output to a file

Example Request Body

[{"key":"Reference #
3","assignedUser":"alan_major@hotmail.co.uk","fields":[{"fieldName":"Page1.JobsDetails.JobCode","fieldValue":"ABC1
234"},{"fieldName
":"Page1.JobsDetails.VisitDate","fieldValue":"25/10/2012"},{"fieldName":"Page1.JobsDetails.VisitTime","fieldValue":"12:
22"},{"fieldName"
:"Page1.JobsDetails.VisitNumber","fieldValue":1},{"fieldName":"Page1.JobsDetails.OrderDetails.UrgentAttention","fieldVa
lue":true},{"field Name":"Page1.JobsDetails.OrderDetails.CalltoConfirm","fieldValue":"Option
2"},{"fieldName":"Page1.JobsDetails.OrderDetails.ComponentList","fieldValue":["Component 1","Component
3"]},{"fieldName":"Page1.Details.FullDetails","fieldValue":"test,test,test."}]}]

Figure 9: Post instruction

If the post instruction is successful, Fiddler will display the Inspectors screen, with a Successfullyprocessed message. See Figure 10.

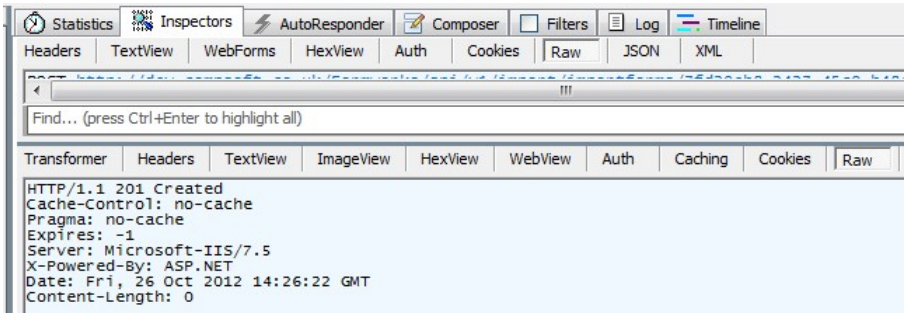


Figure 10. Fiddler indicating that the Post

instruction workedYou can view the pre-populated imported data in

Formworks. See Figure 11.

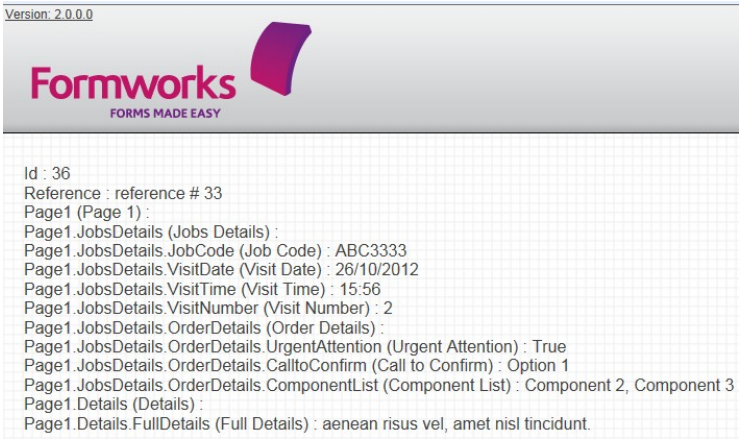


Figure 11: Formworks displaying prepopulated data

Importing XML data to the web service

The above example used the json format to import data, but it is also possible to use the XMLformat. If XML is used, the POST screen Composer screen should look similar to Figure 12.

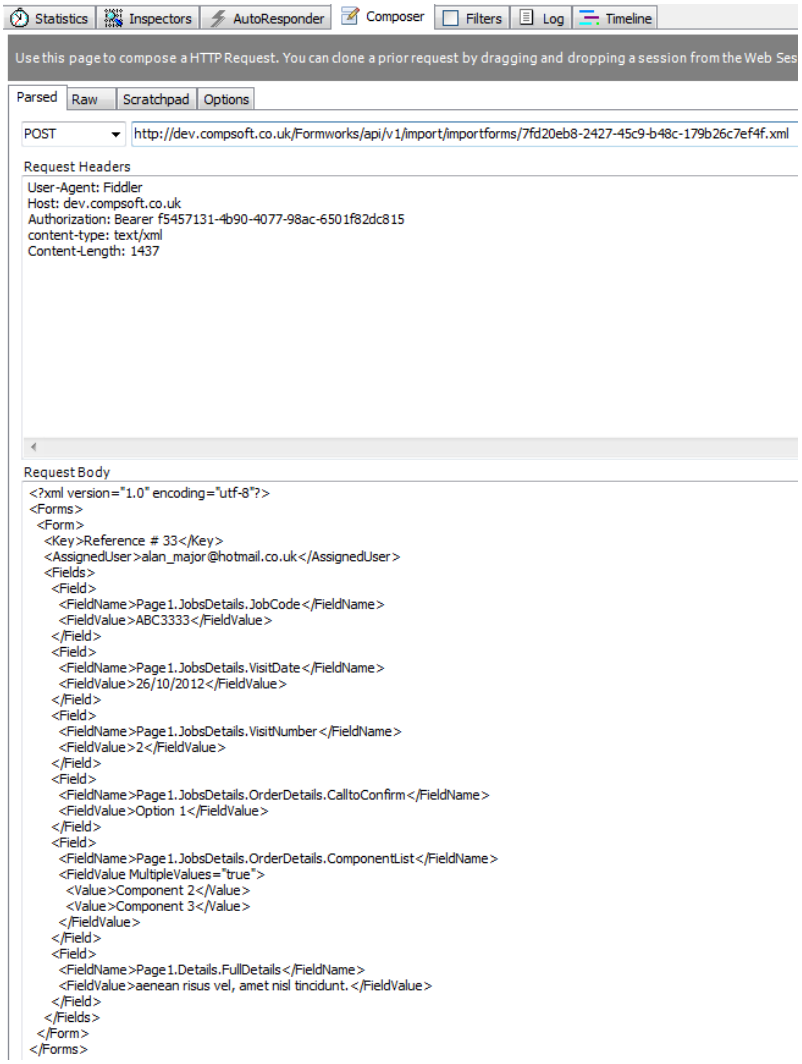


Figure 12: Composer screen using XML format

Invalid Postings

If you attempt to pre-populate form fields with inappropriate values, i.e., incorrect data types areused, Fiddler will provide examples of the type of error message you will receive.

Posting to Date fields

Entering a valid time, like 15:23 against a date field will be accepted, and today's date will berecorded in the pre-pop data.

Entering a value that cannot be interpreted as a date will generate an error similar to the followingexample:

[{"key":"Reference # 40","message":"The value for 'Page1.JobsDetails.VisitDate' cannot be interpreted as the required 'Date'type.\r\n\r\n"}]

Posting Date values to Time fields

Unlike posting a time value to a date field, posting a date value to a time field will generate an error similar to the example below, indicating that the value cannot be interpreted as a time. Entering any value not formatted as HH:MM will generate an error, including an incorrect hour, such as 30:15.

HTTP/1.1 400 Bad
RequestCache-
Control: no-cache
Pragma: no-cache
Content-Type: text/html;
charset=utf-8Expires: -1
Server: Microsoft-
IIS/7.5X-Powered-
By: ASP.NET
Date: Tue, 30 Oct 2012 16:24:04 GMT
Content-Length: 141

[{"key":"Reference # 36","message":"The value for 'Page1.JobsDetails.VisitTime' cannot be interpreted as the required 'Time'type.\r\n\r\n"}]

Posting incorrect values to check box fields

Entering anything other than 'true' or 'false' into a check box field, including 'yes' or 'no', will generatean error of the type shown below.

{"message":"An error has occurred."}

Incorrect assignedUser email address

Formworks uses the assignedUser field to allocate pre-populated forms to the correct user. If the email address supplied does not exist for a valid iPad user account, the message below will be displayed:

[{"key":"Reference # 37","message":"Import unsuccessful, this user cannot be found.\r\n"}]

Posting incorrect values to a digit field

Text fields with a Data Type of Number will not accept any alpha text as input. Valid entries would be, 33 or 33.35 for example. Entering an inappropriate value will generate the error message below:

{"message":"An error has occurred."}

Posting values to a multiple-selection listing

When pre-populating multiple-selection lists, Formworks does not check that the values being uploaded match the values available within the multiple-selection list, as per the Formworks designer. So for example, if the list return values were, "Component 1, Component 2, Component 3", the value "Ok" could be used without the pre-pop upload being rejected.

Posting data to a duplicate Reference value

Pre-pop data can continue to be uploaded against a specific reference, until the iPad user accesses the form and Submits it. At this point the form is locked, and any attempt to upload any additional pre-pop data will fail. Until the iPad user Submits the form, uploading additional pre-pop data will overwrite any previous data held within Formworks. Data Submitted by an iPad user will overwrite any pre-pop data previously uploaded. For examples see below:

Scenario 1

1. Pre-pop data uploaded to Formworks.
2. Pre-pop form is Viewed on the iPad, but not amended.
3. Re-uploaded pre-pop data. This will overwrite the previous upload, but will also update thedata on the iPad.

Scenario 2

1. Pre-pop data uploaded to Formworks.
2. iPad user retrieves pre-pop form, changes data and issues the Save and Close instruction.
3. Pre-pop data re-uploaded to Formworks. Formworks will overwrite the original pre-pop data with the latest upload. The 'Saved' data held on the iPad is not affected by the secondpre-pop upload, because the Save and Close instruction was issued.
4. The iPad user issues the Submit Form instruction. This will overwrite any data held inFormworks and lock the form. This prevents any subsequent pre-pop uploads.

Scenario 3

1. Pre-pop data uploaded to Formworks.
2. iPad user changes data and issues the Submit Form instruction. The form is now locked.
3. Attempt to upload new pre-pop data to Formworks.
4. Formworks data not overwritten, and error message generated. Screen Figure 13.

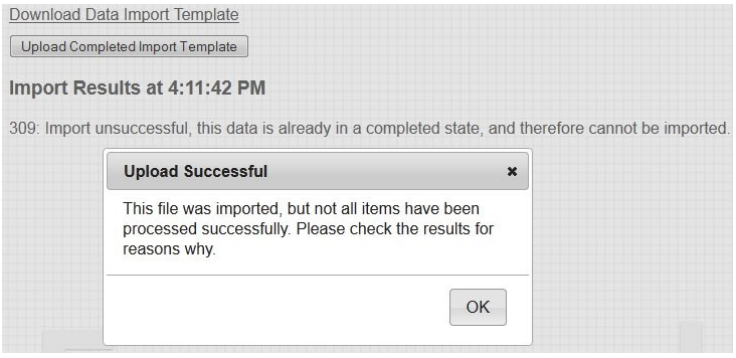
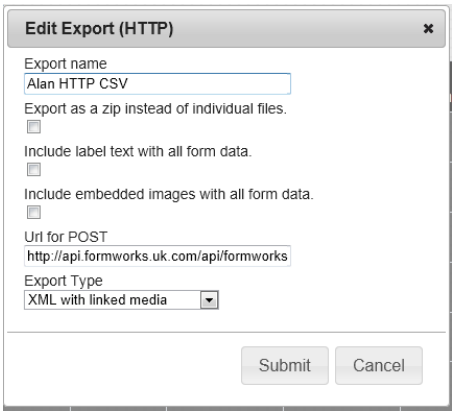


Figure 13: Attempting to upload pre-pop data to a locked form

Web Service Export

This was tested by uploading a document from an iPad, and using the Formworks Export (HTTP) option. HTTP posts, including PDF, CSV and XML data were made. These were captured by a webservice, and the files emailed on.

There was no difference between the HTTP option and standard email of files. However, a difference was found between the fields captured within the CSV and XML files. This is outside ofthe HTTP routine, and is a general issue.



HTTP post option.

Id : 409
Reference :
Page1 (Page 1) :
Page1.Section1 :
Page1.Section1.lblTest (Label text of doom) :
Page1.Section1.Group1 :
Page1.Section1.Group1.Photo1 : Page1.Section1.Group1.Photo1.jpg
Page1.Section1.Group1.Photo2 :
Page1.Section2 :
Page1.Section2.TextTest (Text Test) : Test
Page1.Section2.DigitTest (Digit Test) : 12345
Page1.Section2.DateTest (Date Test) : 11/12/2012
Page1.Section2.TimeTest (Time Test) : 12:41
Page1.Section2.CheckBox (Check Box) : True
Page1.Section2.SingleSelection (Single Selection) : Option 1
Page1.Section2.MultiSelection (Multi Selection) : Option 1, Option 2
Page1.Section2.SignatureTest (Signature Test) : Page1.Section2.SignatureTest.jpg
Page1.Section2.PhotoTest (Photo Test) : Page1.Section2.PhotoTest.jpg
Page1.Section2.SketchTest (Sketch Test) : Page1.Section2.SketchTest.jpg
startLat : 51.43213
startLong : 0.3972859
Page1.Section2.SignatureTest_datetime : 2012-12-11 12:41:59 +0000

The PDF, CSV and XML match with the data uploaded.

TABLES WEB API

Authentication

Each API request needs to have the Authorization header set with the customer API key. Forexample, Authorization : Bearer 748BFBD0-8098-49EE-B09D-0679C085F239

Upload

URL

Live

https://www.formworks.uk.com/api/tables/upload/table_name

Test

https://uat.formworks.uk.com/api/tables/upload/table_name

Content-Type

This header should always be set to text/csv

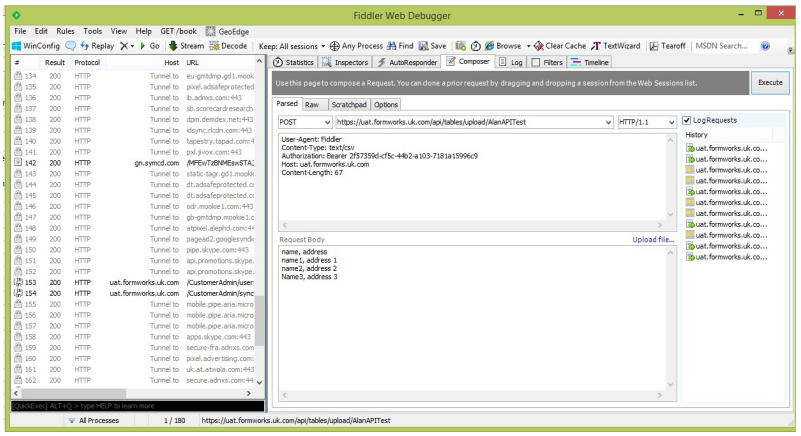
The table_name parameter in the url is the actual name of the table that you are uploading. This name is the unique identifier for tables in the customer account. Calling this method will either create or update the records associated with this table name. The request content of this methodwill be the core csv data.

POST https://uat.formworks.uk.com/api/tables/upload/APITest HTTP/1.1Content-Type: text/csv
User-Agent: Fiddler
Host: uat.formworks.uk.com
Content-Length: 14024
Authorization: Bearer 2F57359D-CF5C-44B2-A103-7181A15996C9

Estate_Name,Property_Address,Block,Road,Postcode
Marsh,44 Middle barton Road,Block 1,Middle Barton Road,OX25 6JN

HTTP/1.1 202 Accepted
Cache-Control: no-cache
Pragma: no-cache Expires: -1
Server: Microsoft-IIS/8.0X-
Powered-By: ASP.NET
Date: Wed, 13 Apr 2016 12:09:48 GMT
Content-Length: 0

Using Fiddler, you upload screen should look similar to the capture.



Download

URL

Live

https://www.formworks.uk.com/api/tables/download/table_name

Test

https://uat.formworks.uk.com/api/tables/download/table_name

This method will simply stream down the csv file.

GET https://uat.formworks.uk.com/api/tables/download/APITest HTTP/1.1Content-
Type: text/csv
User-Agent: Fiddler
Host: uat.formworks.uk.com
Content-Length: 14024
Authorization: Bearer 2F57359D-CF5C-44B2-A103-7181A15996C9

HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache Content-
Length: 14025 Content-Type:
text/csv Expires: -1
Server: Microsoft-IIS/8.0
Set-Cookie: ASP.NET_SessionId=bfpwiwghtbdmnnot1u5sg4ku; path=/; HttpOnlyX-
Powered-By: ASP.NET
Date: Wed, 13 Apr 2016 12:09:59 GMT

Estate_Name,Property_Address,Block,Road,Postcode
Marsh,44 Middle barton Road,Block 1,Middle Barton Road,OX25 6JN

Using Fiddler, your screen should appear similar to the capture: