MoneyWorks®
Seriously Good Accounting Software

Using FileMaker Pro/Go and the MoneyWorks REST APIs

The MoneyWorks REST APIs allow data to be displayed and submitted to MoneyWorks Datacentre using a standard URL. This document describes how to access the REST APIs from FileMaker Pro 12 (and FileMaker Go). The REST API documentation is available at

http://www.cognito.co.nz/support/?developer&art=rest

In FileMaker, it is easy to use the REST APIs to display data in the webviewer, or to extract data using the *Insert from URL* command. However it is not possible to submit data from FileMaker without a bit of a hand, specifically a server-based script to build a POST HTTP request to MoneyWorks instead of a GET.

We have prepared a simple php script (mwrest.php) that does this, essentially acting as a mediator between FileMaker (or similar systems) and the MoneyWorks REST system. This requires that some sort of webserver (such as Apache) and php must be running on your network (although not necessarily on the same machine as MoneyWorks Datacentre). Note that FileMaker Pro Server is not required for this implementation. Installation is outlined at the end of this document.

Note that, for better security, you should use https instead of http.

mwrest.php

To communicate with a MoneyWorks Datacentre file using mwrest.php, you need to specify as a minimum:

1. The IP address of the mwrest server (i.e. the webserver);
2. The internal network address of the MoneyWorks Datacentre (including the port number that it is configured for the REST APIs, which, by default, is 6710);
3. The username and password (if any) of the Datacentre;
4. The name of the MoneyWorks file with which you want to communicate;
5. The user and password of the MoneyWorks file;
6. The action (command and parameters) you wish to perform.

For the purposes of illustration, we will assume that the IP address of the mwrest server is 1.2.3.4, meaning that the URL to use will always start with http://1.2.3.4/mwrest.php/? with a list of parameters, separated by an ampersand (&), following the ?.

Available parameters (note they are case sensitive):

testLink: Used solely for testing that the mwrest.php file is accessible. So

```
http://1.2.3.4/mwrest.php/?testLink
```

should return:

```
MoneyWorks Rest php is active
```

ipp: The ip address and port number of your MoneyWorks Datacentre. For the purposes of illustration we will use 192.168.1.1:6710 for this. Note that, apart from when using the testLink parameter, the ipp must always be specified. To verify that your Datacentre is responding, use the following, which should return the version number of the Datacentre (or a Password required message, if your Datacentre is password protected):

```
http://1.2.3.4/mwrest.php/?ipp=192.168.1.1:6710&cmd=version
```

dcUser: The Datacentre username (omit if none)
dcPass: The Datacentre password (omit if none)
doc: The file path and name of your MoneyWorks document on the Datacentre (relative to the designated Datacentre documents folder). Files in a subdirectory need to be specified as "foldername/filename", but with the "/" percent encoded (i.e. as "foldername%2Ffilename"). You must percent encode slashes and ampersands, as otherwise these would interfere with the structure of the URL itself.
docUser: The MoneyWorks document username

docPass: The MoneyWorks document password
cmd: The type of REST request to run (e.g. import, evaluate, doreport). See below for a list of commands. Note that some commands will require additional parameters.

The order of the parameters is not important, with the exception that the data parameter (used for specifying the data to import) must be last. It is also important that the parameters do not contain ampersands or slashes (these can be specified as %26 and %2F respectively).

Note: There are a number of gotchas when passing data as part of a URL as we do here. One is that from the URL's perspective, a space and a + are sort of interchangeable (technically a space can be represented by a +). So, if you want to pass a "+" as part of some data or expression, pass it as "\+" (e.g, not "3 + 5" but "3 \+ 5"). mwrest will correctly substitute the "+" back in for MoneyWorks.

Examples:

The following uses the list command to return an XML list of available files in the Datacentre for Datacentre user *bill* whose password is *dog*

```
http://192.168.1.217:5710/mwrest.php/?
ipp=192.168.1.219:6700&dcUser=bill&dcPass=dog&cmd=list
```

In the following two examples, we use the evaluate function to return the name of the designated MoneyWorks file:

The MoneyWorks file "Acme Widgets.mwd6" for user "fred" whose password is "secret".

```
http://1.2.3.4/mwrest.php/?ipp=192.168.1.1:6710&doc=Acme
%20Widgets.mwd6&docUser=fred&docPass=secret&cmd=evaluate&expr=name
```

The file "Acme3.mwd6" in the "Test1" Datacentre subdirectory (which has no password). Note that the full path name (percent encoded) is passed as the document name, and is case sensitive.

```
http://192.168.1.217:5710/mwrest.php/?
ipp=192.168.1.219:6700&dcUser=Test1&doc=Test1%2facme3.mwd6&docUser=fred&docPass=secret&cmd=
evaluate&expr=name
```

For the remainder of this document, we'll just use <base URL> to represent the logon credentials, so for example the above URL will be represented as

```
<base URL>&cmd=evaluate&expr=name
```

&cmd=list

Returns a list of available reports or forms for the document.

```
<base URL>&cmd=list&folder=reports
```

Additional parameters can be used to specify whether you want standard or custom plug-ins, and the type of form. For example the following will retrieve a list of custom statements:

```
<base URL>&cmd=list&folder=forms&filter=user&type=stmt
```

Although it is possible to retrieve a list of controls for a report using &name=reportname, it is not clear how these can be implemented in FileMaker.

When &cmd=list used with no document specified (just the Datacentre credentials), ra list of available MoneyWorks files is returned.

&cmd=evaluate

Evaluates the MoneyWorks expression passed in using the required &expr= parameter:

```
<base URL>&cmd=evaluate&expr=name
```

Returns the document name

```
<base URL>&cmd=evaluate&expr=lookup("SMITH", "name.dbalance")
```

Returns the balance owed by debtor SMITH

```
<base URL>&cmd=evaluate&expr=sumselection("transaction.gross", "type=`DII` and status =
`P`")
```

returns the total amount of all incomplete and posted debtor invoices

```
<base URL>&cmd=evaluate&expr=find("product.code", "category1=`BRONZE`", 100, ",",
"sellprice-")
```

returns a comma separated list of up to 100 product codes whose category1 field is BRONZE, sorted in reverse order of sellprice (i.e. most expensive first).

&cmd=export

Returns a set of export reports for the specified table. See Appendix A of the MoneyWorks manual for a list of table and field definitions.

```
<base URL>&cmd=export&table=name
```

Returns all the data stored in the name table as tab-delimited text

```
<base URL>&cmd=export&table=name&format=xml
```

The same as above, except in XML (although note that FileMaker seems to translate the < > to &gt; and &lt; respectively).

```
<base URL>&cmd=export&table=name&format=xml&search=dbalance>1000
```

The same as above, but limited to customers who owe more than $1,000.

```
<base URL>&cmd=export&table=name&format=[code]\t[name]\t[dbalance]\r&search=dbalance>1000
```

A tab delimited list of code, name and balance for those customers that owe more than $1,000

&cmd=doreport

Returns the named report. The standard and custom controls for the report can be specified as parameters. By default the report will be tab delimited text, but the format parameter also allows html and pdf.

```
<base URL>&cmd=doreport&report=Balance Sheet
```

Returns the Balance Sheet for the current period as tab delimited text

```
<base URL>&cmd=doreport&report=Balance Sheet&format=pdf
```

Same as previous, but report is returned as a pdf. In FileMaker you would need to put this into a container field to be able to view it.

```
<base URL>&cmd=doreport&report=Balance Sheet&format=pdf&include_unposted=1
```

As above, but this time with the "Include Unposted" on.

```
<base URL>&cmd=doreport&report=Balance Sheet&format=pdf&from=20101130
```

Returns a pdf of the Balance Sheet as at the period ending 30 Nov 2010. Note that dates must be specified as YYYYMMDD.

```
<base URL>&cmd=doreport&report=Bank%20Register&Bank=1000: Main BankAccount&List
Transactions=0&format=pdf
```

Returns a pdf of the Bank Register for account 1000 (Main Bank Account). Because "List Transactions" is off, it returns just the chart (so we cannot use format=text, as it cannot display graphics). Note that Bank and List Transactions are custom controls on the report.

&cmd=doform

Returns a pdf of the nominated form (invoice, statement etc) printed for the nominated record(s). The records are those that meet the search criteria, as specified using the &search parameter. Searching on the sequencenumber is the best way to uniquely identify a record (and can be returned as part of the importing process). In particular care should be taken with the search expression to avoid pdf-ing a large number of records. In FileMaker you will need to display the result in a container field.

```
<base URL>&cmd=doform&form=Maple%20Ruled%20Invoice&search=sequencenumber=1305
```

Returns a pdf of transaction with sequencenumber 1305 printed using the form "Maple Ruled Invoice".

```
<base URL>&cmd=doform&form=Purchase%20Order&search=ourref=186
```

Returns a pdf printed as a purchase order of all transactions with ourref (order number) of 186.

```
<base URL> &cmd=doform&form=Plain%20Ruled%20Invoice&search=namecode="SMITH" and type="DI@"
```

Returns a pdf of all invoices sent to customer SMITH (which might be quite a lot, and quite large to download). Each invoice starts on a new page of the pdf.

```
<base URL>&cmd=doform&form=Product%20Labels&search=code="BA100"
```

Returns a pdf of the Product Label for item whose code "BA100"

```
<base URL>&cmd=doform&form=Product%20Labels&search=[transaction:type=`SO@` and transdate =
today()][product]
```

Returns a set of product labels for all items sold (on sales orders) today.

```
<base URL>&cmd=doform&form=Job%20Work%20Statement&search=code="100"
```

Returns the Job Work Statement for job 100.

&cmd=import

Imports the data specified by the &data= parameter. The data must be in xml, and the &data parameter must be the last in the URL. Slashes and ampersands in the xml will need to be percent encoded. There is one additional optional parameter, return_seq. Set this to true to receive as a reply the sequencenumber of the last record imported (this can be stored in FileMaker to indicate that the record has been imported, and, for transactions, can also be used to print a invoice/form or to post it). If return_seq is not set, the response will be of the form *created n; updated m.*

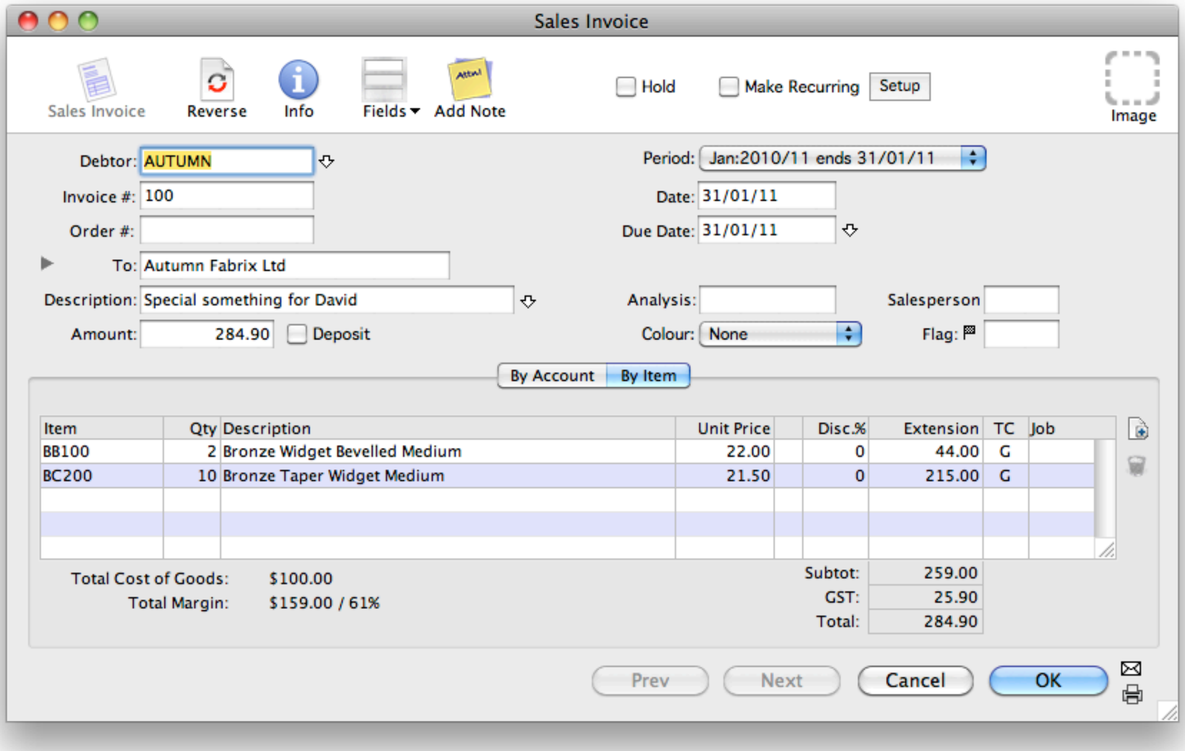
The following will import a new job *100GC* with description *A great opportunity* for client *SMITH*.

```
<base URL>&cmd=import&return_seq=true&data=<table name="job"><job><code>100GC</
code><description>A great opportunity</description><client>SMITH</client></job></table>
```

The following imports a two line invoice for customer AUTUMN:

```
<base URL>&cmd=&return_seq=true&data=<table
name="transaction"><transaction><namecode>AUTUMN</namecode><ourref>100</ourref><type>DI</
type><transdate work-it-out="true" /><tofrom work-it-out="true" /><contra work-it-
out="true" /><description>Special something for David</description><subfile
name="detail"><detail><detail.stockcode>BB100</detail.stockcode><detail.stockqty>2</
detail.stockqty><detail.unitprice>22</detail.unitprice><detail.account work-it-out="true" /
><detail.description work-it-out="true" /><detail.net work-it-out="true" /><detail.TaxCode
work-it-out="true" /><detail.Tax work-it-out="true" /><detail.gross work-it-out="true" /></
detail><detail><detail.stockcode>BC200</detail.stockcode><detail.stockqty>10</
detail.stockqty><detail.unitprice>21.5</detail.unitprice><detail.account work-it-
out="true" /><detail.description work-it-out="true" /><detail.net work-it-out="true" /
```

```
><detail.TaxCode work-it-out="true" /><detail.Tax work-it-out="true" /><detail.gross work-
it-out="true" /></detail></subfile><gross work-it-out="true" /></transaction></table>
```



☐ Hold ☐ Make Recurring

Debtor: AUTUMN
Invoice #: 100
Order #:
To: Autumn Fabrix Ltd
Description: Special something for David
Amount: 284.90 ☐ Deposit
Period: Jan:2010/11 ends 31/01/11
Date: 31/01/11
Due Date: 31/01/11
Analysis:
Salesperson
Colour: None
Flag:

By Account | By Item

Item	Qty	Description	Unit Price	Disc.%	Extension	TC	Job
BB100	2	Bronze Widget Bevelled Medium	22.00	0	44.00	G	
BC200	10	Bronze Taper Widget Medium	21.50	0	215.00	G	

Total Cost of Goods: $100.00
Total Margin: $159.00 / 61%

Subtot: 259.00
GST: 25.90
Total: 284.90

In more readable form, the xml looks like:

```
<?xml version="1.0"?> <table name="transaction">
        <transaction>
              <namecode>AUTUMN</namecode>
              <ourref>100</ourref>
              <type>DI</type>
              <transdate work-it-out="true" />
              <tofrom work-it-out="true" />
              <contra work-it-out="true" />
              <description>Special something for David</description>
              <subfile name="detail">
                      <detail>
                              <detail.stockcode>BB100</detail.stockcode>
                              <detail.stockqty>2</detail.stockqty>
                              <detail.unitprice>22</detail.unitprice>
                              <detail.account work-it-out="true" />
                              <detail.description work-it-out="true" />
                              <detail.net work-it-out="true" />
                              <detail.TaxCode work-it-out="true" />
                              <detail.Tax work-it-out="true" />
                              <detail.gross work-it-out="true" />
                      </detail>
                      <detail>
                              <detail.stockcode>BC200</detail.stockcode>
                              <detail.stockqty>10</detail.stockqty>
                              <detail.unitprice>21.5</detail.unitprice>
                              <detail.account work-it-out="true" />
                              <detail.description work-it-out="true" />
                              <detail.net work-it-out="true" />
                              <detail.TaxCode work-it-out="true" />
                              <detail.Tax work-it-out="true" />
                              <detail.gross work-it-out="true" />
                      </detail>
              </subfile>
```

```
                <gross work-it-out="true" />
        </transaction>
</table>
```

Note that all XML imports to MoneyWorks are required to start with <?xml version="1.0"?>. This is added automatically by mwrest. If you want to copy the above xml examples to paste into the appropriate MoneyWorks list, ensure that you add the <?xml version="1.0"?>.

&cmd=post

Posts the transaction with the specified sequencenumber.

```
        <base URL>&cmd=post&seqnum=1731
```

Posts transaction whose sequencenumber is 1731.

&cmd=image

Retrieves (or uploads) the image (if any) for the product (identified by product code), or the transaction (identified by sequencenumber). This would need to go into a container field in FileMaker.

```
        <base URL>&cmd=image&product=BA100
```

Retrieve the image for product BA100.

```
        <base URL>&cmd=image&transaction=1731
```

Retrieve the image for transaction whose sequencenumber is 1731.

The MoneyWorks REST APIs also allow an image to be uploaded to the MoneyWorks. However we haven't figured how to do this from FileMaker Go (it may not be possible).

mwrest Extensions

Although not part of the REST API, the following parameters have been added to mwrest to allow emailing direct from the (correctly configured) machine hosting the mwrest.php.

&email=emailaddress

Will email a pdf returned by the REST request to the specified email address, with the optional subject and body specified by &subject= and &body= parameters. Note that the mwrest script must have write access to its local directory (temporary PDFs will be written before they are emailed, they are automatically deleted), and will need to be altered to include the sender's email details.

Note that it is relatively trivial (and probably preferable) to use the inbuilt FileMaker email script steps to email directly from the FileMaker client.

Refer to http://cognito.co.nz/support/?developer&art=rest for more REST documentation.

Installing Apache and php

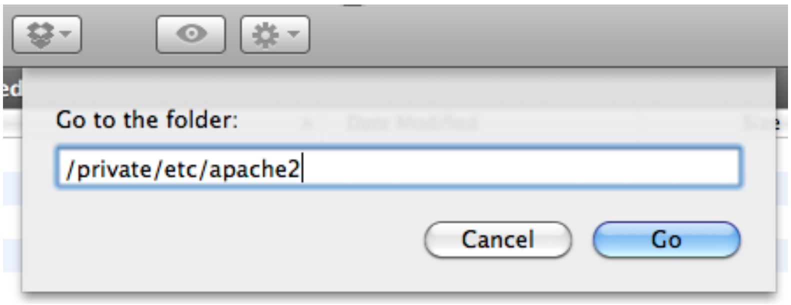
On a Mac:

To turn on Apache, open your System Preferences, go to Sharing and turn on Web Sharing (turn the checkbox on). The IP address of the computer will be displayed (note that this might change unless you have your Network preferences to have a manual IT address.

To enable php:

You need to edit the file /private/etc/apache2/httpd.conf. The easiest way to do this is to use a text editor such as the (excellent and free) textwrangler (from http://www.barebones.com/products/TextWrangler/).

You can locate the file in the Finder by using Go to Folder in the Go menu. The folder you want to go it is /private/etc/apache2, so enter as shown below:



Go to the folder: /private/etc/apache2

Right-click on the http.conf file and choose Open With>Other... and then select TextWrangler from your applications folder. You are looking for the following line (about line 116 of the file):

```
#LoadModule php5_module        libexec/apache2/libphp5.so
```

Remove the # to uncomment the line and save the file.

You also need to create a php.ini file, which should be in the /private/etc directory. Navigate to this directory (use the Go to folder if necessary). You should see a file called php.ini.default. Duplicate this and rename it php.ini.

Finally, put the php file (mwrest.php) into the /Library/WebServer/Documents directory and restart Apache.

On Windows:

Get an IT person.