1. Introduction

About FileMaker & PHP

PHP is a great web scripting language because, like FileMaker Pro, you can actually wrap your head around it. With PHP you can build web applications that meet just about any clients needs. The web offers many PHP resources for help, and the PHP Documentation site is an excellent start. See http://www.php.net

The benefits for FileMaker Pro developers increase dramatically because PHP can talk to FileMaker Pro straight from a web page, allowing you to build dynamic web pages based on the content of your clients FileMaker Pro solution. Using PHP from your HTML web page you can dynamically display FileMaker Pro find results, customize your website layout interface and enable users to update, create and delete FileMaker Pro records.

The most tantalizing prospect of using PHP, is that although PHP scripts are inside your HTML document, they are processed before the HTML, CSS or JavaScript. Thus your HTML, CSS and JavaScript can be dynamically created and modified based on results of your FileMaker Pro database contents!

What about the learning curve?
PHP is easy to learn. There are tutorials online (google it) and every major book store will afford PHP manuals for the eager student. PHP can be used with a variety of skill levels: from simple page modifications to complex developers wanting a full web application. PHP is a lot like FileMaker scripting, yet it has all the familiar functions and controls of a procedural programming language.

How much is this going to cost?
PHP is free! It is best to install PHP as an Apache web server module. Mac OS X users have Apache pre-installed and can download a PHP package installer from Marc Lyanage at http://www.entropy.ch/software/macosx/welcome.html. Windows users can download the Apache, MySQL, PHP trio (WAMP) at http://www.cyborgspiders.com/wamp/wamp.php.

What about compatibility?
FileMaker Pro 7 files need to be shared via XML for custom web publishing on FileMaker Server 7 Advanced ($ ouch! $). An extended privilege set needs to be added to your file with the key word “fmxml”. The user account that will be used to access your file via the web will need to have fmxml privileges.

FileMaker Pro 6 or earlier files need to be shared through the web companion on FileMaker Pro client.

See FileMaker documentation on sharing databases via XML for more information.

2. Understanding PHP arrays

What's an array?

Simply put an array is a list of items, a lot like FMP repeating fields. Each item is indexed by a number and can contain a variety of different data types. For example, you could have an array with 3 items in it, 1 being a number, 2 being text and 3 being another array.

Here is a simple 4 element array...

```
$modes = ('Layout','Browse','Find','Preview');
```

In PHP the dollar sign is used to identify variables, single or double quotes contain strings and array values are separated by commas. When you create an array like this, the elements are assigned a numeric index (sometimes called the 'Key') starting with zero. So in the example above...

```
$modes[0] = Layout
$modes[1] = Browse
$modes[2] = Find
$modes[3] = Preview
```

In PHP you can also assign a name index as well as the internally assigned index, like so...

```
$modes['design'] = 'Layout';
$modes['data']   = 'Browse';
$modes['search'] = 'Find';
$modes['print']  = 'Preview';
```

This is called an associative array. So...

```
$modes['design'] =  $modes[0] = Layout
$modes['data']   =  $modes[1] = Browse
$modes['search'] =  $modes[2] = Find
$modes['print']  =  $modes[3] = Preview
```

This is an extremely brief introduction to arrays but all you need to know to start working with PHP and the FX Class. For more information please visit
http://www.php.net/manual/en/function.array.php

Understanding the FX Class

What is the FX Class?

More appropriately, what is a Class? In PHP you can write a little program that has it's own functions and it's own variables, it's called a Class. It's an object that you can instantiate and use whenever you want.

The actual FX object -or- Class, is written in a document called fx.php. That document is in the FX folder that you downloaded and is why you have to put it on your web server to 'install' FX.

How does FX talk to FileMaker Pro?

You could talk to FileMaker Pro yourself using any web browser by typing the URL that connects you to your host (http://www.yourfmpserverIP.com), then adding onto the URL specific FileMaker defined WPE syntax. The WPE syntax and how to use it can be found in the FileMaker Pro documentation on Custom Web Publishing with XML.

When you use WPE syntax in the URL, for example to perform a find, your web browser will connect to the server, authenticate and load the results. The returned data, however, is formatted in XML and contains all sorts of other meta data beyond the goals of your original query.

FX Class simplifies this. Giving you PHP functions very similar in name to their FileMaker Pro counterparts, it builds the URL query for you, sends it to the server and then parses the XML results into a beautiful PHP array.

You don't have to know FileMaker URL syntax to use the FX Class. You don't have to know XML. You never have to leave the PHP environment.

How do I use the FX Class?

The rest of this document shows a simple PHP script that performs a FileMaker Pro find using the FX Parser. Use of the FX Class is explained briefly here, with enough details to get you started. For more detail see the FX Documentation inside the FX folder.

Using the FX Parser

What is FX Parser

The FX Parser is a quaint example of how to use the FX Class. It is a PHP document that will show you how to setup FX for your particular FileMaker Pro database. What FX Parser does after that, is give you a robust and interactive outline of the data array as it comes back from your FileMaker Server: it parses the returned array into a JavaScript driven HTML viewer.

Installing the FX Class and FX Parser

The FX Parser is distributed with the FX Class from http://www.iviking.org/. Make sure you have downloaded the latest version of this distribution.

The FX Class does not have an installer, it is merely a folder called “FX”. To install the FX class put it’s folder at the root level of your site. Your site should now have the path **www.yourdomain.com/FX**.

Inside the FX folder are developer documents, examples, tutorials and of course the FX Parser. The FX Parser resides in it’s own folder called “fxparser” at the root level of the FX folder. With your favorite text editor open up the document “index.php” inside the “fxparser” folder and you’ll be ready to go!

Re-cap of what you are about to do

When you become familiar with PHP and the FX Class you will be typing your own PHP code, querying against a FileMaker Pro databases and building dynamic web pages.

But right now, we’re going to show you how that’s done using the FX Parser. We will edit some PHP code in the “index.php” document and then load the page to see the results.

Editing the FX Parser “index.php”

The FX Parser is a document call “index.php” inside the “fxparser” folder at the root level of the “FX” folder. Open “index.php” in your favorite text editor.

We will be editing the lines of code near the top between the lines marked

```
#       BEGINNING OF USER CODE HERE
        ....
#       END OF USER CODE HERE
```

The code you need to modify is well documented. These lines of code are good examples of how you will use FX in your own web solutions. What follows are step by step instructions of how to modify these lines of code so that FX Parser works and you can see how to use the FX Class.

Instantiating and Configuring the FX Object
The first line of code needs to tell PHP where the FX object code is located. You do this by using the “include” or “require” statement(s).

```
require_once($_SERVER['DOCUMENT_ROOT'] . "/FX/FX.php");
```

With this line, we’ve loaded the FX object so if PHP receives any calls to FX functions they can be executed.

> **NOTE:** DOCUMENT_ROOT does not work on Windows. Please see Chris Hansen’s modifications at the beginning of “index.php” for the work around.

Now we need to get FX going. This is done be creating an FX object. Once we create an FX object then we can start talking to it and telling it what to do.

```
$query = new FX(your_serverIP, 80, 'FMPro7');
```

No tricks here. You are saying that the variable $query will be and FX object. When creating the FX object called $query, you need to give it the IP address of your FileMaker Server, the port number in which Custom Web Publishing is running on, and the version of FileMaker Pro you are using. More information on this is found in the FX documentation.

Now that we have an ‘instance’ of FX, we can work with it and eventually tell it to give us the results of a FileMaker Pro find.

Two important lines come next, one deals with authentication and the other with the database. First, authentication:

```
$query->SetDBPassword(your_password, your_username);
```

Of course your web available database should be protected with a username and password. Notice the syntax here:

```
$query->SetDBPassword ...
```

To use the functions in the FX Class object we first type the objects name, $query, then we use the symbols ->, then we type the function. This syntax for calling object functions and setting object parameters is consistent throughout the object oriented programming of PHP and will be used when working with the FX Class.

Next, we need to tell $query, which database file to use and which layout to work from.

```
$query->SetDBData(your_fm7File, your_layout, 3);
```

The SetDBData() function is used here. The first two parameters are your file name and layout name respectively. They are strings so make sure you surround them in quotes. Ex:

```
$query->SetDBData("accounting", "general_ledger", 3);
```

The last parameter "3" is optional. It specifies how many records to display in the result. Keep in mind that FX and FileMaker will work through **all** the records even though you get just the last 3 in the found set. So if you have a lot of records it's not worth using this parameter and you will be better off performing a find to get the 3 records or the exact set you want.

Review of our New FX Object

So far we have told PHP where to find the FX related code by using a require statement. We then created an FX object called $query, passing the IP address, Port number and Version of FileMaker Pro as parameters.

Once the FX object was created we called the 2 basic and essential functions SetDBPassword() and SetDBData(). PHP now knows about FX and can execute it's code. FX now knows about your FileMaker Server, Port, Version, Username, Password, File Name and Layout Name.

Don't forget to make sure the account you use for this job has access to the File and Layout that was prescribed.

We are at exactly the same point in a FileMaker database that a user would be if they just opened up a file, logged in and navigated to a specific layout. What we're going to do now is perform a find using that layout.

Performing the Find

In FileMaker, you would perform a find by entering find mode first and then typing the criteria in the field(s). With FX it's even easier because you don't have to enter find mode, you just start entering find criteria right into the fields. This is done with the AddDBParam() function. Only one AddDBParam() function per field.

```
$query->AddDBParam(your_fieldname, your_find_criteria);
```

Again, your parameters in this functions are strings, so you will need to enclose them in quotes. To search on criteria in more than one field you just keep adding AddDBParam() lines.

```
$query->AddDBParam(your_fieldname2, your_find_criteria2);
$query->AddDBParam(your_fieldname3, your_find_criteria3);
```

In FileMaker, the logical OR can be added to a find by creating new find requests. To do this with FX you add another AddDBParam() line with two special parameters:

```
$query->AddDBParam('-lop', 'or');
```

This AddDBParam() line with the special '-lop','or' params creates a New Find Request. Just like being in FileMaker Find Mode and then selecting 'New Request' from the menu.

You can now populate the fields in this request by using the AddDBParam() function, one line per field, just like you did on the first request. When you are satisfied with the criteria setup for your find you ready to "Perform" the find.

Performing the find is the easiest part. This is done with the function FMFind() with once **important** difference. You need to use a new variable for the last function to work: a new variable that will hold the results of your search. If you just type

```
$query->FMFind();
```

it will indeed perform a find, but it's also wants to **give you an array** of the results. You need to define a variable to hold that array.

So, a proper find is executed with code like this,

```
$queryResult = $query->FMFind();
```

and that is exactly what you see in the FX Parser.

Please remember that the code

```
$queryResult = $query->FMFind();
```

must not be changed in the FX Parser. The Parser will work with an array call $queryResult and if you change the name of that variable.... oops .... it breaks.

Displaying Your Results
If you have filled all the parameters of the functions above correctly, and your PHP and FileMaker pro installation is configured correctly, this page is ready to be displayed.

Know this, however, rarely does the first time prove successful. Learning to setup and configure FileMaker Pro for XML sharing is cumbersome in it's own right. Adding to that the possible configuration challenges of PHP and Apache (or your own web server) can complicate things.

Upload this index.php file to your web server and direct your browser to it. If you have blatant PHP errors they will be displayed immediately. If you see the basic FX Parser page displayed then you are close to success.

If your communication with FileMaker Pro was unsuccessful, there could be a plethora of reasons: sharing configuration and account privileges are most likely the issue. Either way, you will receive a resulting array from the FX class.

FX Parser will loop through this array and show you all the elements of the first level in red. If any of the elements in this first level contain arrays, FX Parser will give you a link with the following icon:

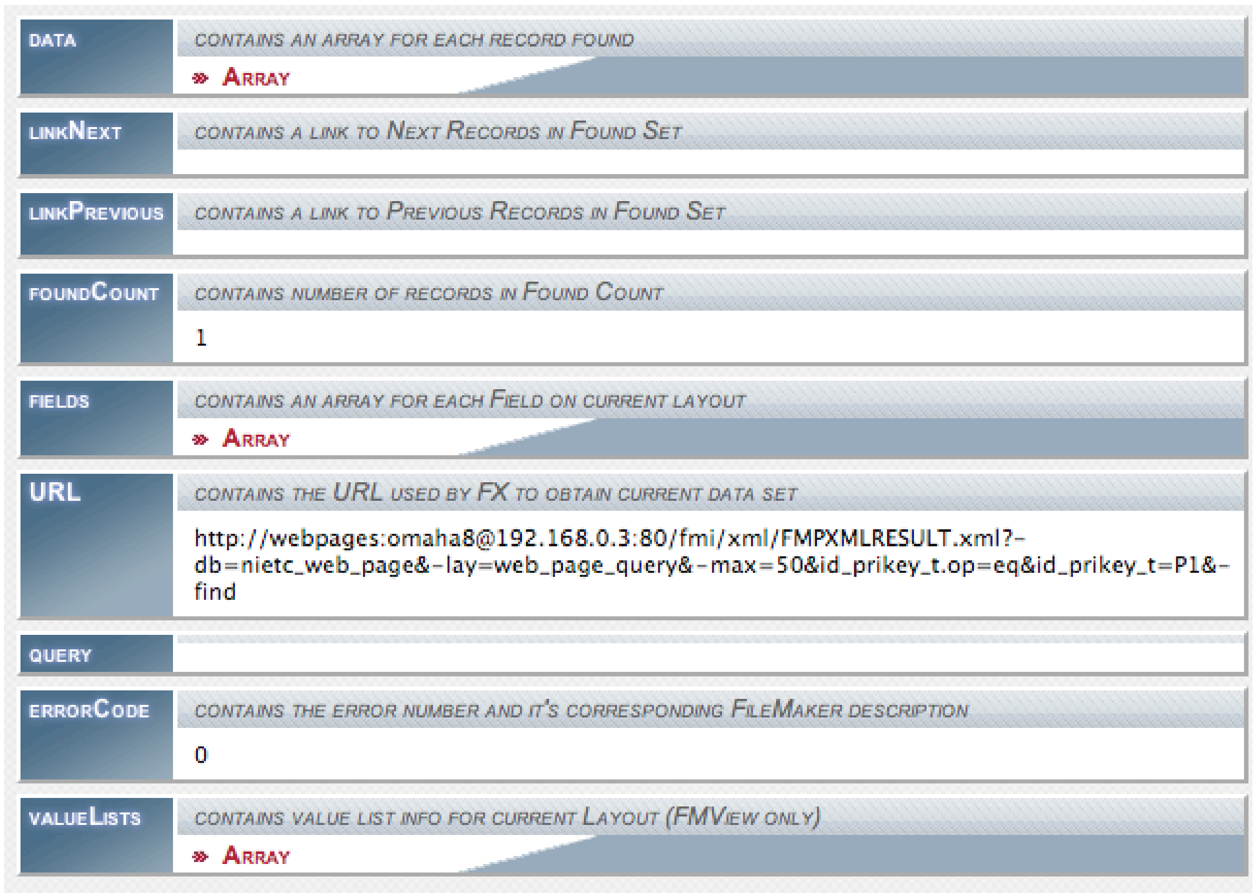
» ARRAY

Any array icon can be clicked on to display it's contents. If you had a successful query against your FileMaker Pro database, you will see an element in the first level of the result call 'data'. Click on the array icon and you will be able to drill down into the sub level arrays to see your resulting data.

If there were errors, you will see elements in the first level array that describe the errors to help you troubleshoot the problem.

The FX Data Set Displayed
If you received a successful data set from your qeury you will receive and array with 9 elements. The 'key' or index of those elements are displayed at the left in white.

data,linkNext,linkPrevious,foundCount,fields,URL,query,errorCode,valueLists



DATA	CONTAINS AN ARRAY FOR EACH RECORD FOUND » ARRAY
LINKNEXT	CONTAINS A LINK TO NEXT RECORDS IN FOUND SET
LINKPREVIOUS	CONTAINS A LINK TO PREVIOUS RECORDS IN FOUND SET
FOUNDCOUNT	CONTAINS NUMBER OF RECORDS IN FOUND COUNT 1
FIELDS	CONTAINS AN ARRAY FOR EACH FIELD ON CURRENT LAYOUT » ARRAY
URL	CONTAINS THE URL USED BY FX TO OBTAIN CURRENT DATA SET http://webpages:omaha8@192.168.0.3:80/fmi/xml/FMPXMLRESULT.xml?-db=nietc_web_page&-lay=web_page_query&-max=50&id_prikey_t.op=eq&id_prikey_t=P1&-find
QUERY	
ERRORCODE	CONTAINS THE ERROR NUMBER AND IT'S CORRESPONDING FILEMAKER DESCRIPTION 0
VALUELISTS	CONTAINS VALUE LIST INFO FOR CURRENT LAYOUT (FMVIEW ONLY) » ARRAY

So the array of our $queryResult has 9 elements. FX Parser shows us those elements and the data that each element contains:

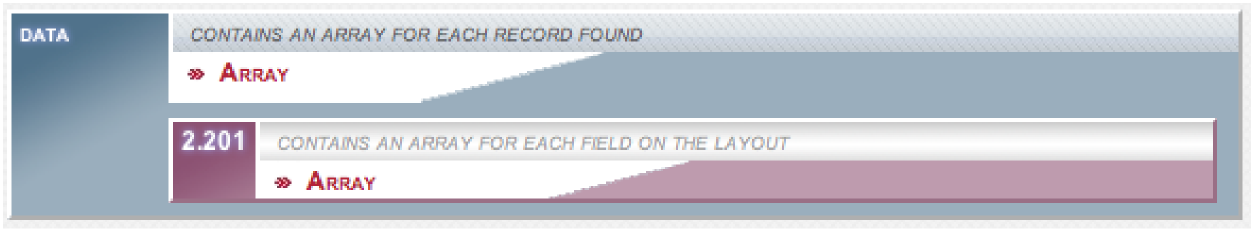
```
$queryResult['data'] = array
$queryResult['linkNext'] =
$queryResult['linkPrevous'] =
$queryResult['foundCount'] = 1
$queryResult['fields'] = array
$queryResult['URL'] = // the URL created to make the XML request to FileMaker
$queryResult['query'] =
$queryResult['errorCode'] = 0
$queryResult['valueLists'] = array
```

An interesting element here is the 'foundCount'. If you wanted to check for results and display a "Sorry, no records were found" message, you would simply use the following code:

```
if($queryResult['foundCount']==0)
    echo "Sorry, no records were found.";
```

Typically in a web page you want to display the data. Therefore, most of your time is going to be spent working with the contents of the ['data'] element.

Click on the >>Array icon in your web browser to see the contents of $queryResult['data'].


DATA
CONTAINS AN ARRAY FOR EACH RECORD FOUND
» ARRAY
2.201
CONTAINS AN ARRAY FOR EACH FIELD ON THE LAYOUT
» ARRAY

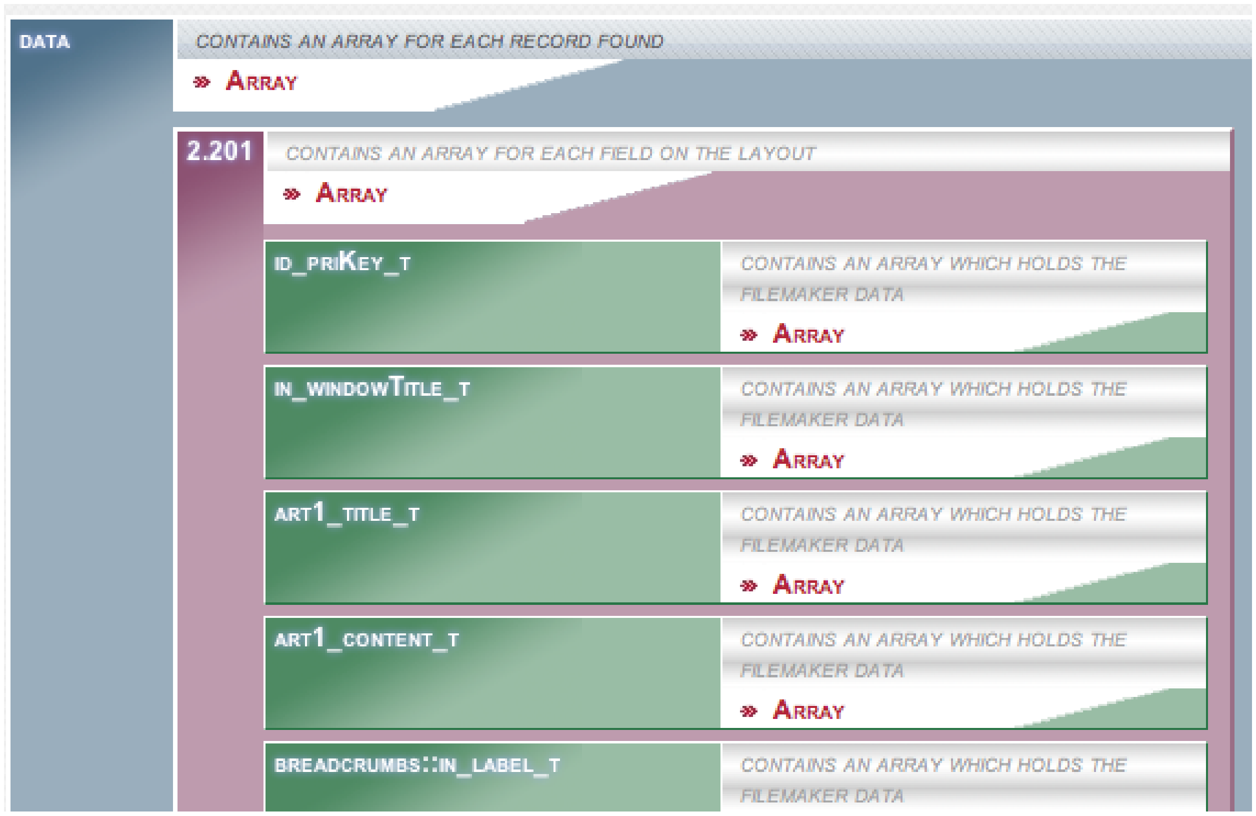
In my example, there is only on found record, thus when I click on 'Array' I get one item in red. You will have one red item in this display for every record in your found result.

Notice the key or index for this record: **2.201** That is the internal record ID and modification ID from FileMaker, concatenated together with a period. This becomes important when you use FX to edit and delete records because you need to pass the record ID as a parameter.

Inside the data element is an array with an element for each found record, in my case just one. The key for that one element is 2.201 and the contents (or value) of that element is .... *another array*!

Again, click on the >>Array icon in red, and open it up to see the contents. Remember we are now looking at the contents of $queryResult['data'][2.201].

My example:



DATA

CONTAINS AN ARRAY FOR EACH RECORD FOUND

» ARRAY

2.201

CONTAINS AN ARRAY FOR EACH FIELD ON THE LAYOUT

» ARRAY

ID_PRIKEY_T	CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY
IN_WINDOWTITLE_T	CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY
ART1_TITLE_T	CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY
ART1_CONTENT_T	CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY
BREADCRUMBS::IN_LABEL_T	CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA

The contents of the 2.201 element are now visible. Remember, this is another array. Inside the 2.201 element is an array shown in green and it has elements for every field on the layout you specified. My layout has more fields on it than I care to display here, but you can see the first 5 elements and their keys (or indexes).

The keys for this array are the names of the fields. Remember, each element in and array has a key and a value and in this instance the keys are the field names, the values are .... *more arrays*!   (coffee break)

Let’s reflect on where we. We are looking at the variable $queryResult. This is the variable that you said is equal to $query->FMFind()... remember:

```
        $queryResult = $query->FMFind();
```

Inside the $queryResult variable is an array with a ‘data’ element.

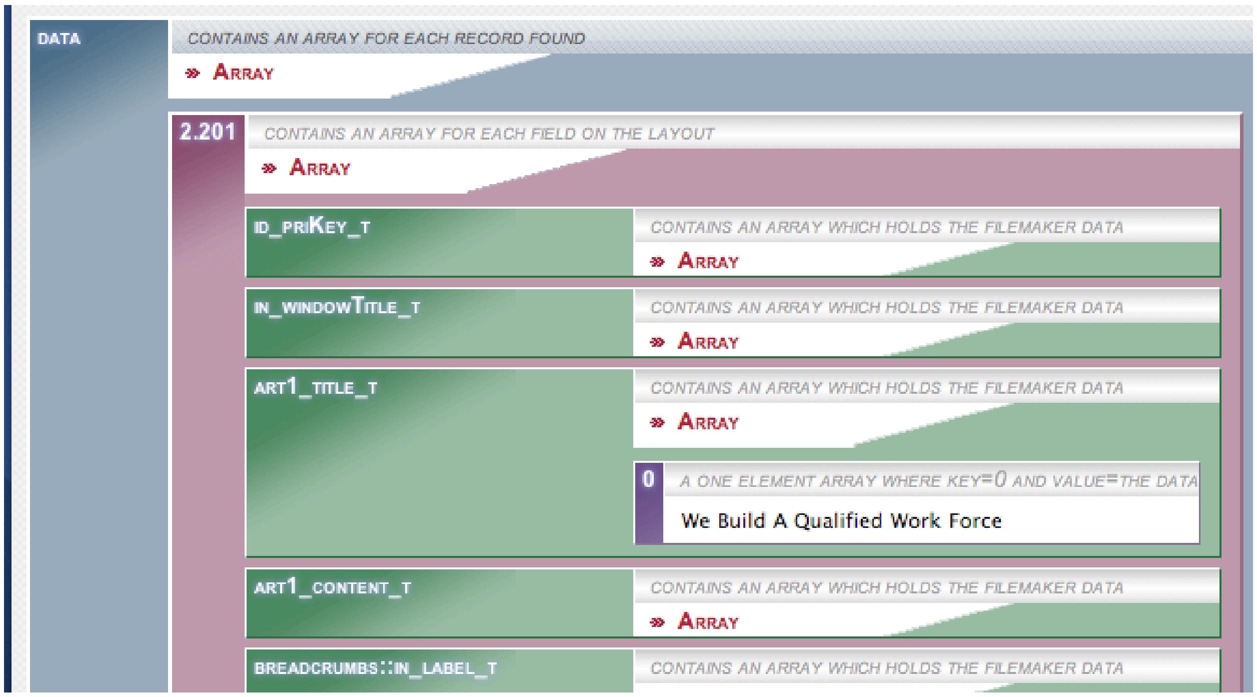
```
        $queryResult[‘data’];
```

Inside the ‘data’ element is an array with an element for each record in the found count. We selected the first record with a key of the recID.modID of 2.201.

```
        $queryResult[‘data’][2.201];
```

Inside the 2.201 element is an array with an element for every field on the specified layout. These elements contain keys and values. The key is the name of the field, the value is another array.

Click on the >>Array icon to display the contents of one of the field elements.



DATA — CONTAINS AN ARRAY FOR EACH RECORD FOUND » ARRAY

2.201 — CONTAINS AN ARRAY FOR EACH FIELD ON THE LAYOUT » ARRAY

ID_PRIKEY_T — CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY

IN_WINDOWTITLE_T — CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY

ART1_TITLE_T — CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY

0 — A ONE ELEMENT ARRAY WHERE KEY=0 AND VALUE=THE DATA

We Build A Qualified Work Force

ART1_CONTENT_T — CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA » ARRAY

BREADCRUMBS::IN_LABEL_T — CONTAINS AN ARRAY WHICH HOLDS THE FILEMAKER DATA

I clicked on the field “art1_title_t”. It’s content is another array with only one element. That element is the **holy grail** of all the the array elements: it is always going have a key of zero, and its value will be the field contents.

So we have taken the variable $queryResult, gone into the ‘data’ element, then the 2.201 element, then the art1_title_t element and finally the 0 element. How do you output this to your web page. Like so:

```
echo $queryResult[‘data’’][2.201][‘art1_title_t’][0];
// prints to the broswer,
// We Build A Qualified Work Force
```

The FX Parser shows you visually each array and it’s elements. You can drill down by clicking on the >>Array links. You can also see a straight text display of the whole result set, parsed out in mono font, with colored syntax, by click on the TEXT ONLY link at top.

This is only one example how to use the FX Class. See the FX Functions document to learn how FX can be used to it’s fullest capacity. No matter what the FX function, you can use the FX Parser to display the results and help understand the array structures.

Final Comments

Using the FX Parser From Here On

You may find that no matter how familiar you become with PHP, FX and the FileMaker XML syntax, there are times when troubleshooting is difficult and complicated. The FX Parser is a good tool to keep at hand because it is abstracted from the HTML and control logic of your project. Pasting just your FX code into the Parser gives you a quick and dirty look at PHP/FileMaker behavior, and you can quickly see if the problem is the FX / FileMaker combo, or somewhere else.

Because the FX Parse utilizes the FMErrors.php file, you can still be benefited by viewing an error result set. You can drill down to see the back trace or view the error code and text to help troubleshoot the problem.

The FX Parser is Open

The FX Parser is free and is distributed under the same artistic license as the FX Class. Any suggestions for modifications should be forwarded so that the tool can be of more use to the community. Please submit your changes or suggestions to:

lhallberg@nietc.org

Thanks to Chris Hansen and the FX Team

Kudos to Chris Hansen **and all** who have made FX a working solution. PHP in combination with FileMaker Pro is an outstanding and marketable asset. If you are new to dynamic web development you have found a worthy deployment method.

The synergy of FileMaker Pro and PHP will be ongoing. The tools and knowledge developed are most welcomed in the open source community. Please share your thoughts, ideas and code with the rest of us at the FX form:

http://fxdialog.comitas.no/

Cheers,

-Lance Hallberg