# An Atlas with the 4D Ajax Framework

By Jean-Yves Fock-Hoon

Technical Note 07-29

## Abstract

This Technical Note shows how to build an Atlas using 4D Ajax Framework (4DAF) objects and features. Specifically the Data Matrix (Image Browser) and Data Driven Tabs are explored. The resulting application is a "mash-up" that uses 4D data to leverage the Google Maps API in order to generate an interactive atlas.

A sample database is included.

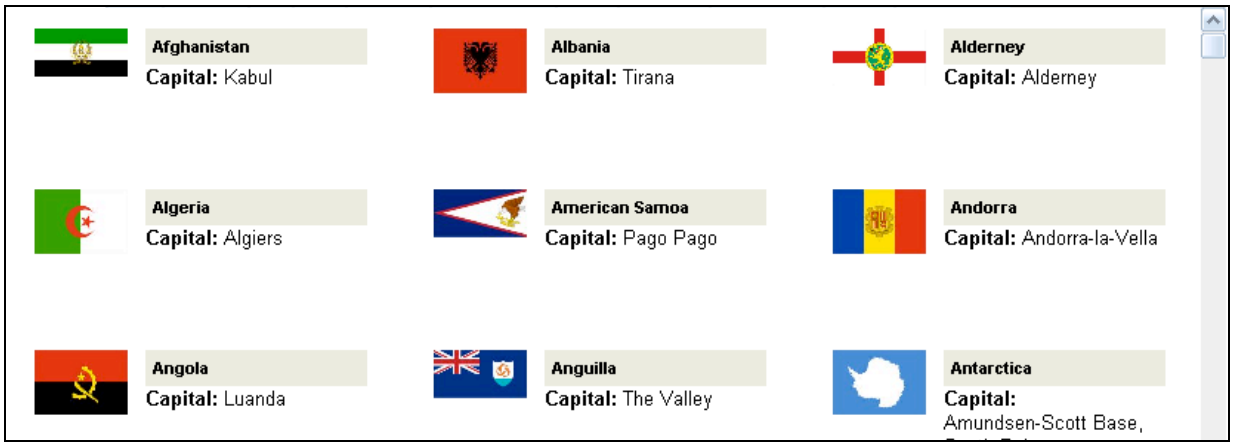
## Overview

4D Ajax framework 1.1 introduces two new features that we are going to illustrate into the same example by building an atlas based on the 4D Ajax Framework (4DAF). This Technical Note will show you how to use the Data Matrix (also called Image Browser) and Data Driven Tabs.

Using these two features the example database mixes 4D data (country names and capitals) with Google Maps, displayed inside a DDW, to build an interactive atlas.

### Data Matrix (Image Browser)

The Data Matrix is a new object introduced in 4DAF version 1.1. This object can display data as a grid, including Picture fields. This is the view that we will use to illustrate our atlas based on country flags. Here is an example:

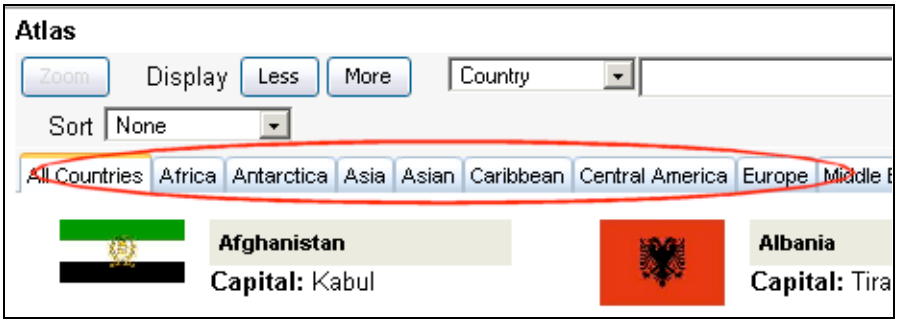



For more information on the Data Matrix refer to the Daxipedia entry:

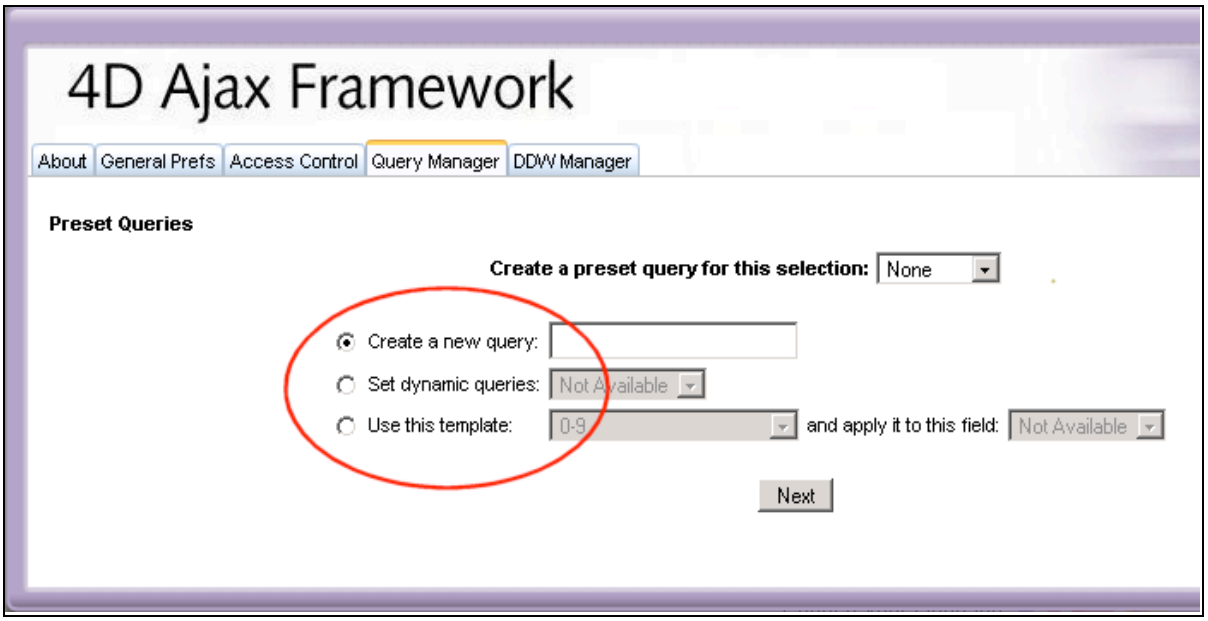
http://daxipedia.4d.com/index.php/Image_Browser

## Data Driven Tabs

Data Driven Tabs are tabs generated based on the values, e.g. from a specific field. Here is an example:




These tabs can be generated from "preset" queries. 4DAF version 1.1 offers three types of preset queries, as shown below:




Dynamic queries, based on the value of a field, will be used in this Technical Note. Refer to the "4D Ajax Framework Admin Reference" for more information on the different query types. You can download the reference here:

http://www.4d.com/support/documentation.html

## The Atlas Example

Note: This Technical Note includes the merged 4DAF Atlas application as well as the interpreted source database. Please note that the 4DAF is not installed in the source database, as only those who have purchased a Web 2.0 Pack license may use the 4DAF for development.

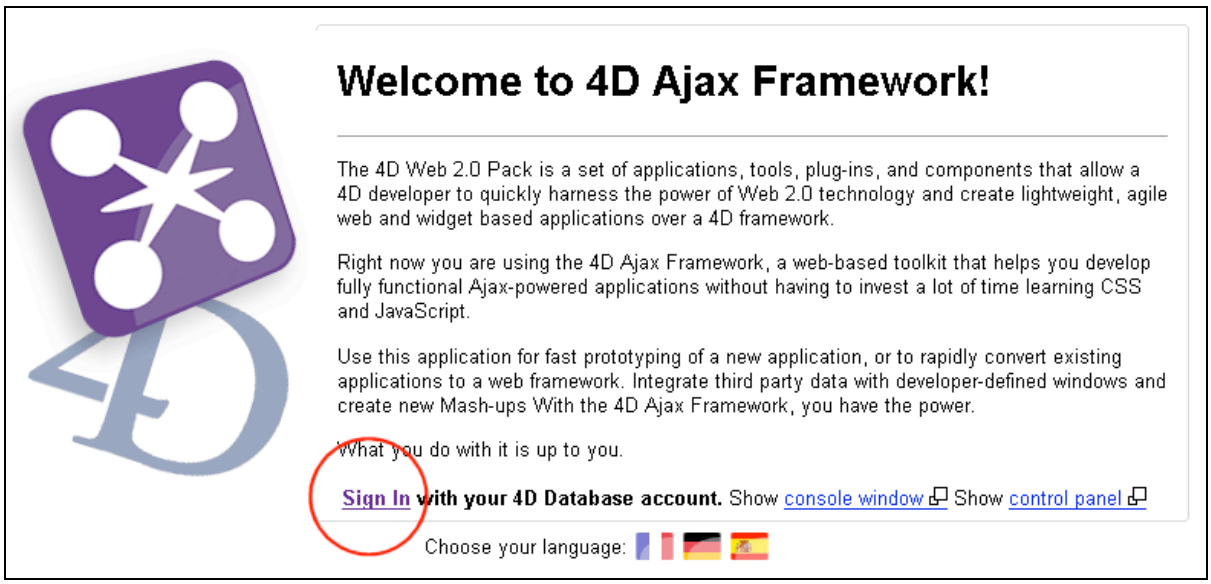
If you prefer to view the finished product, or you do not own a 4D Web 2.0 Pack license, you can use the merged 4DAF Atlas application and follow along, ignoring any steps that involve modifying the source database.

In order to use the interpreted source database you will need to install the 4DAF (as per the installation instructions) and also make the following modifications:

Edit the ***DAX_DevHook_DDW_Install*** Component method and insert a call to the *m_Assign_DDW* Project method at the end. Then execute the Component method ***DAX_Dev_Reset*** in order to rebuild the XML files.

**Note** *Calling "DAX_Dev_Reset" is generally faster than quitting and re-launching the database.*

After setting up the database, you should now connect to the 4DAF Client (by default at http://localhost:8080/) with the Administrator login (no password) and open the Control Panel dialog. Be sure to log into the 4DAF Client and not just the Control Panel:

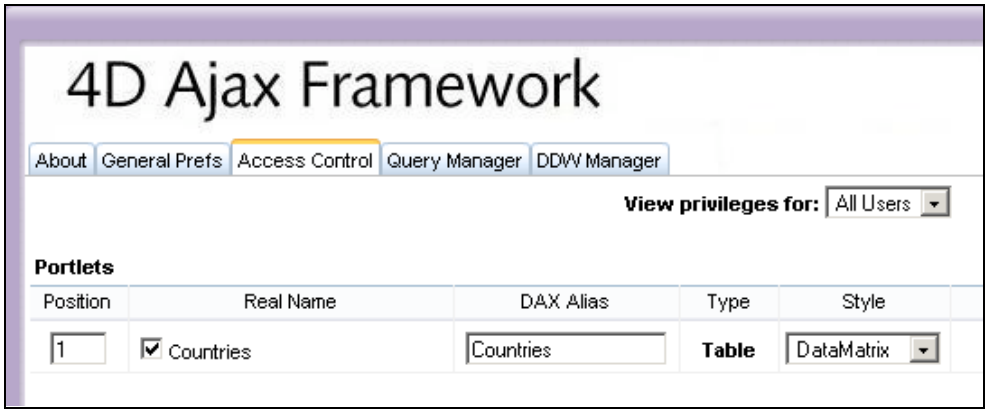



To complete the example the following steps will be taken (and discussed in detail below):

- Make the necessary configuration settings in the 4DAF Control Panel.
- Explore the Data Matrix from within the 4DAF Client.
- Make the necessary changes to a custom HTML file in order to present the Data Matrix.

## Configure the Control Panel

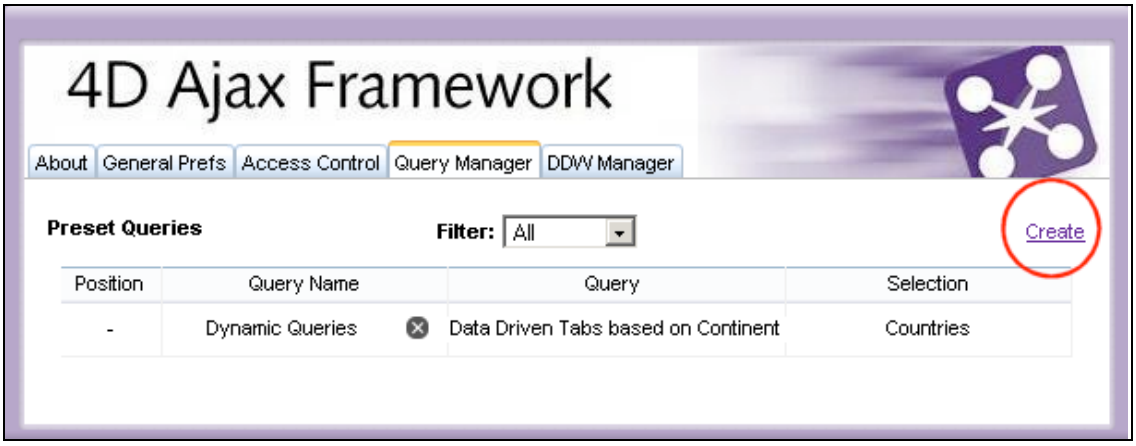
In the "Access Control" tab, we can see our only table, "Countries":




We can change the current "Style" pop-up menu from "Grid" to "DataMatrix". This View will be now displayed as a Data Matrix.

The database also contains a Developer Defined Window (a.k.a. DDW) named "Map it", which was created by the *m_Assign_DDW* Project method that we previously called. This DDW method will execute the *mMapIt* Project method. The purpose of this DDW is to open an extra window with a call to the Google Maps API and the DDW will display the map of the country within a separate window.

We can now click on the "Query manager" tab in order to create our Data Driven Tabs, based on continents:




Click on the "Create" link on the top-right of the window in order to create a preset query. Select "Countries" for the selection and select the "Set dynamic queries" radio button. In the pop-up menu field, select "Continent", so our tabs will be generated based on the continents stored inside the data file. Click on next to save the preset query.

In the "Query manager" tab you can view all available preset queries. You can select a table in order to display all preset queries for the selected table. If you select "Countries", we can still see our unique preset query but we can also see a new checkbox, "Display the All records Tab". This checkbox will display the "All records" tab as first tab control in our window. We will keep it checked in order to see all countries.  Note that you need to set the "Filter" to "Countries" in order to see only the preset queries for the Countries selection.

## Explore the Data Matrix Window

We can now close our Control Panel dialog. As we can see, there is a new icon (Portlet) in the Portal labeled "Countries". Click on the Countries Portlet to open then new Data Matrix:


Our records are now displayed as a picture gallery with the flag pictures on the left side, and other fields on the right side. You can select any of these records and double-click them to edit them. Of course, you can edit any of these records and change their values. You can also select any record and delete it, or create a new one.

We can also see our Data Driven Tabs labeled with Continents. Select any of them to narrow down the selection to the appropriate continent. If you select Europe, you can see all countries that belong to the Europe continent. If you modify or create any record, and add a new value for the tab controls, the new tab control will be updated with the new value with the next automatic refresh (by default, after up to 3 minutes). If all query tabs cannot be displayed in the current window, extra icons will be displayed on the window in order to allow you to browse them.

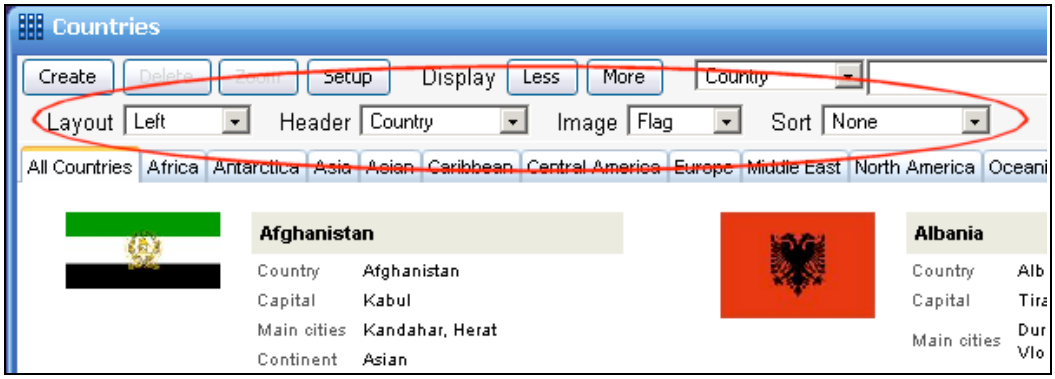
Like in other windows, you can still narrow down the selection by querying a value. We can select "Capital" in the pop-up menu and type P. The query type-ahead will narrow down the selection with all records with Capital's value starting with P. We

should see 2 countries, Czech Republic (Prague) and France (Paris). We can select "All countries". Since we still have P in the query area, our selection would be all countries with Capital's value starting with P.

The number of records displayed in this window depends on its size. You can display more or less by increasing or decreasing its size. If you do not want to change its current size, you can still use the "More" and "Less" buttons. These buttons will act like a zoom buttons, to display more or less records on the screen. Click on the Less button in order to reduce the width and height of the display used by one record. Click on the "More" button to increase the display used for that record. By clicking on the "Less" button, you can display up to a specific number of records per row. The number depends on the width of the window. By clicking on the "More" button, you can manage to display one record, one row in that window. If the display size is too small, you cannot see all field values.

Select any record and click on the "Zoom" button. This will display the selected record on the whole screen without editing it, so, you can have a quick look at its contents. Click again on the "Zoom" button to come back to the regular gallery.

Click on the "Setup" button if you want to customize your display. A new toolbar will be displayed, between the current toolbar and the tab controls:




The "Layout" pop-up menu allows you to define the position of the text or title as follows:

- Top: The picture will be displayed on the top of the header and no other fields will be displayed, unless if you edit them.
- Bottom: Same for Top, except that the picture will be under the header field.
- Left: Header and preview on the current fields will be displayed on the left side of the picture field.
- Right: Header and preview on the current fields will be displayed on the right side of the picture field.

The "Header" pop-up controls header of the record, i.e., the title of the record. If you select "None", the record will have no title. Only the picture field will be displayed if you select the "None" and "Top" values.

The Image pop-up menu lets you select which picture (e.g. from Picture field) will be displayed inside the grid.  If you select "None" for the picture and for the title, with "Top" for the Layout, a generic picture and an empty title will be displayed for all records.

The "Sort" pop-up menu is the sorting order. You can choose any of the selected fields and the selection will be sorted in ascending based on the type of the field.

At this time, the default interface in the 4DAF Client is not customizable. You cannot hide any of those features, such as hiding the Delete button or preventing to edit any record, for example. The only way to lock some of these features would be to define your own HTML page and use the 4D Ajax framework objects. The next section will cover this part.

## Configure the HTML

From a Web browser, we can request the "pictrix.html" file (e.g. http://localhost:8080/pictrix.html). As we can see, we have an embedded Data Matrix inside our html page with some of the features:


If we look at the source code for "pictrix.html", we have our grid named "myDataMatrix1" in the JavaScript. The script automatically logs in (as Designer), then the grid itself is created with the line "myDataMatrix1 = new dataMatrix". Here is the syntax of the dataMatrix constructor.

```
myDataMatrix = new dataMatrix(  targetNode, selectionName, header,
                                imageField, contents, zoomContents, layout,
                                margin, zoomLevel, scrollMode, forceRows,
                                forceColumns, allowCellMouseInteraction);
```

Here is a description of each parameter:

1. **targetNode** - in which DOM element would you like insert the image browser. If set to null, the browser will pop up in a new window. In our example, we will provide our current name with the following syntax: $('dax_matrix1')
2. **selectionName** -Selection name or Selection ID where data resides. In our example, we are providing "1" as our first table.
3. **header** - Header string, where [selection]field syntax will be replaced with actual data from corresponding field. Use auto to automatically set first available field as header. In our example, we are using the country name ([Countries]Country) as record header.
4. **imageField** - Name or id of image field. Use auto to automatically select the first available Picture field. In our example, we are defining the "Flag" picture field, [Countries]Flag.
5. **contents** – This is the information displayed next to the picture field. You can display all fields (default setting) or define a string to be displayed. In our example, we are defining the vDetail variable that will contain some fields to be displayed. The content of this variable has been defined before creating the grid, a mix of static texts and fields.
6. **zoomContents** - Same as "contents", but displayed when cell is in zoomed-in state. We use the same concept, with the vDetail variable.
7. **layout** - Location of the image in image mode: top, bottom, left, or right. Top and bottom display only the header, left and right display header and contents. In our example, we chose "Top".
8. **margin** – This is the cell margin size between records. Our default value is 10 pixels.
9. **zoomLevel** - 0 (smallest possible cell size) and above. Adapts cell size to object based on this number.
10. **scrollMode** – Define a vertical or horizontal scrollbar. Horizontal is recommended for one row object. In our example, we chose "ver" for the vertical scrollbar.
11. **forceRows** - Forces number of rows to be displayed in the grid. This must be accompanied by forceColumns. Using this will disable change zoom level toolbar buttons ("more" and "less"). In our example, we do not use it. Our current values are "null".
12. **forceColumns** - Forces number of columns to be displayed in the grid. This must be accompanied by forceRows. Using this will disable change zoom level toolbar buttons ("more" and "less"). In our example, we do not use it. Our values are "null".
13. **allowCellMouseInteraction** - Allows for the cell content to be clickable. Our current value is "False".

After creating the object, we can define other attributes with the "customize" function. Here is the syntax:

```
myDataMatrix1.customize( showToolbar, showAuxToolbar, useEditor,
                         createBtn, deleteBtn, zoomBtn, setupBtn,
                         displayBtn, layoutBtn, searchBtn,
                         headerDropdown, ImageDropdown, sortDropdown);
```

Parameters defined in this command are Boolean variables defined just before the call. The True value will enable/display the feature, while the False value will disable/hide that feature.

1. **showToolbar** - Display the main toolbar.
2. **showAuxToolbar** - Display the Setup toolbar when displaying the dialog for the first time. If you do not want to display that toolbar at all, you also need to hide the Setup button.
3. **useEditor** - Enable or disable the ability to edit records.
4. **createBtn** - Display the "Create" record button (main toolbar).
5. **deleteBtn** - Display the "Delete" record button (main toolbar).
6. **zoomBtn** - Display the "Zoom" button (main toolbar).
7. **setupBtn** - Display the "Setup" toolbar
8. **displayBtn** - Display both "more" and "less" buttons (main toolbar).
9. **searchBtn** - Display the "Search" box (main toolbar).
10. **layoutBtn** - Display the select image position pop-up (Setup toolbar).
11. **headerDropdown** - Display the "choose header" field pop-up (Setup toolbar).
12. **imageDropdown** - Display the "choose image" field pop-up (Setup toolbar).
13. **sortDropdown** - Display the "choose sorting" field pop-up (Setup toolbar).

Unfortunately, Auto-resize is not implemented for Data Matrix in version 1.1. This means that your Data Matrix objects have a static size in your HTML pages. The "pictrix.html" page has a work-around that allows you to make it a little bit better. Upon resize, a JavaScript code will redefine the window size and call the refresh. The refresh allows you to redefine the size of the window and to display more or less records in your Data Matrix. Unfortunately, the refresh does not work well with Data Driven Tabs. If you have too many of them, the navigation tabs will not get displayed. A future version should fix this issue and should also make that JavaScript work-around obsolete.

Here is the code defined in the Body element to implement resizing:

```
onresize="myDataMatrix1.window.size($('dax_matrix1').offsetWidth,$('dax_matrix1
').offsetHeight);myDataMatrix1.refresh();"
```

## In summary

Data Matrixes are very useful when pictures need to be displayed. However, pictures fields are not always required and you can use that view to display data in a gallery. They are easy to implement and very powerful, combined with Data Driven Tabs, in the provided 4DAF Client or embedded in your own HTML pages.

## A Note about 4D Web 2.0 Pack

The products in 4D Web 2.0 Pack are a departure from most other 4D products. As 4D Web 2.0 Pack is a subscription-based product it is expected that incremental releases will be made. Thus please note that this Technical Note is based on 4DAF

version 1.1. As new features are implemented this Technical Note may become obsolete (faster than most other 4D products).

## Related Resources

Refer to Technical Note 07-17, "4D Ajax Framework: Preset Queries" for more information on the Preset Query feature.

For the latest information on the 4DAF consult the latest documentation and also check the 4D Web 2.0 Pack Wiki:

http://daxipedia.4d.com

For the latest news about 4D Web 2.0 Pack or to find out how to purchase it please see:

http://www.4d.com/products/4dweb20pack.html